



TUGAS AKHIR - TE 145561

**RANCANG BANGUN ALAT KONTROL TEKANAN
UDARA PADA SISTEM *AIR PRESSURE BOOSTER*
UNTUK Pengereman *DRAWWORK* DALAM PROSES
PENGEBORAN**

Rifki Riyadani
NRP 10311500000052

Pembimbing
Ir. Djoko Suprajitno R., MT.
Ciptian Weried P., S.ST., MT.

DEPARTEMEN TEKNIK ELEKTRO OTOMASI
Fakultas Vokasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018



ITS
Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR - TE 145561

**RANCANG BANGUN ALAT KONTROL TEKANAN
UDARA PADA SISTEM AIR *PRESSURE BOOSTER*
UNTUK Pengereman *DRAWWORK* DALAM PROSES
Pengeboran**

Rifki Riyadani
NRP 1031150000052

Pembimbing
Ir. Djoko Suprajitno R., MT.
Ciptian Weried P., S.ST., MT.

DEPARTEMEN TEKNIK ELEKTRO OTOMASI
Fakultas Vokasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018



ITS
Institut
Teknologi
Sepuluh Nopember

FINAL PROJECT - TE 145561

***DESIGN AN AIR PRESSURE CONTROL DEVICE ON
AIR PRESSURE BOOSTER SYSTEM FOR
DRAWWORK BRAKING IN DRILLING PROCESSES***

Rifki Riyadani
NRP 10311500000052

Supervisor
Ir. Djoko Suprajitno R., MT.
Ciptian Weried P., S.ST., MT.

DEPARTEMENT OF ELECTRICAL AUTOMATION ENGINEERING
Faculty of Vocations
Sepuluh Nopember Institute of Technology
Surabaya 2018

PERNYATAAN KEASLIAN

TUGAS AKHIR

Dengan ini penulis menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir dengan judul "**Rancang Bangun Alat Kontrol Tekanan pada Sistem Air Pressure Booster untuk Pengereman Drawwork dalam Proses Pengeboran**" adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang penulis akui sebagai karya sendiri.

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

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

Surabaya, 30 Juli 2018



Rifki Riyadani
NRP 1031150000052

-----Halaman ini sengaja dikosongkan-----

RANCANG BANGUN ALAT KONTROL TEKANAN PADA SISTEM AIR PRESSURE BOOSTER UNTUK PENGGERAMAN DRAWWORK DALAM PROSES PENGEBORAN

TUGAS AKHIR

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

Menyetujui :

Dosen Pembimbing I

Dosen Pembimbing II



Ir. Djoko Suprajitno R. M.T. Ciptian Weried P. S.ST., M.T.
NIP. 195506221987011001 NIP. 1990201711060

**SURABAYA
JULI, 2018**

-----Halaman ini sengaja dikosongkan-----

**RANCANG BANGUN ALAT KONTROL TEKANAN PADA
SISTEM *AIR PRESSURE BOOSTER* UNTUK Pengereman
DRAWWORK DALAM PROSES PENGEBORAN**

Nama : Rifki Riyadani
Pembimbing I : Ir. Djoko Suprajitno Rahardjo, M.T.
Pembimbing II : Ciptian Weried Priananda, S.ST., M.T.
Pembimbing III : Sopar Simanullang, S.T.

ABSTRAK

Pada sistem *air pressure booster* terdapat 3 pompa yang digunakan untuk menaikkan tekanan udara. Pompa menaikkan tekanan dengan melakukan *stroke*. Pada proses pengeboran di rig, pompa yang terdapat pada sistem *air pressure booster* diatur secara manual. Pompa pada *air pressure booster* terus bekerja saat *drawwork* digunakan karena tidak terdapat *stop* pompa otomatis ketika tekanan udara sudah cukup untuk *disbrake drawwork*. Dari data yang diperoleh, pompa akan melakukan lebih dari 2000 *stroke* ketika *drawwork* bekerja dalam waktu 1 jam.

Sistem *air pressure booster* yang ditambah alat kontrol akan dapat menggunakan pompa secara efisien. Alat kontrol juga dapat digunakan untuk monitoring jumlah kerja pompa. Monitoring jumlah kerja pompa berguna untuk acuan *maintenance*.

Alat dapat mengatur tekanan udara pada tangki dengan mengubah persentase buka *control valve*. *Control valve* akan terbuka 100% saat arus keluaran dari *converter* sebesar 20mA. Monitoring jumlah kerja pompa dapat dilakukan menggunakan HMI.

Kata Kunci : Kontrol Tekanan, *Air Pressure Booster*, *Drawwork*.

-----Halaman ini sengaja dikosongkan-----

***DESIGN AN AIR PRESSURE CONTROL DEVICE ON AIR
PRESSURE BOOSTER SYSTEM FOR DRAWWORK BRAKING
IN DRILLING PROCESSES***

Name : Rifki Riyadani
Supervisor I : Ir. Djoko Suprajitno Rahardjo, M.T.
Supervisor II : Ciptian Weried Priyananda, S.ST., M.T.
Supervisor III : Sopar Simanullang, S.T.

ABSTRACT

On an air pressure booster system there is 3 pumps used to increase air pressure. Pump increase the air pressure by doing a stroke. The drilling proces in rig, pump on air pressure booster system set by manual. Pumps of air pressure booster will be continues to work when the drawwork is used because there is no automatic stop if air pressure is enough to disbrake drawwork. From the data obtained, the pump will do more than 2000 stroke when drawwork is used within 1 hour.

Air pressure booster system that can be efficiently by equip control device. With the equip control device it's possible to reduce the number of pump is used and can monitoring how many pump do stroke. Monitoring the amount of pump is use for maintenance reerence.

The device can control air pressure of tank with set the percentage open of control valve. Control valve will open with percentage 100% if current output of converter is 20mA. Monitoring pump work can do by use HMI.

Keyword : Pressure Control, Air Pressure Booster, Drawwork.

-----Halaman ini sengaja dikosongkan-----

KATA PENGANTAR

Maha Suci Allah SWT atas segala kasih sayang dan cobaannya, sehingga penulis dapat menyelesaikan laporan tugas akhir ini dengan judul:

RANCANG BANGUN ALAT KONTROL TEKANAN PADA SISTEM AIR PRESSURE BOOSTER UNTUK Pengereman DRAWWORK DALAM PROSES PENGEBORAN

Tugas akhir ini mempunyai beban 4 SKS (Satuan Kredit Semester) dan disusun guna memenuhi persyaratan untuk mencapai Gelar Ahli Madya pada Departemen Teknik Elektro Otomasi, Fakultas Vokasi, Institut Teknologi Sepuluh Nopember (ITS) Surabaya.

Tidak ada gading yang tak retak, begitu pun dengan laporan tugas akhir ini yang masih jauh dari sempurna. Baik dalam penyusunan, maupun dalam pembahasan masalah, karena keterbatasan pengetahuan penulis. Untuk itu, dalam kesempatan ini pula penulis mohon maaf atas segala ketidaksempurnaan yang ada. Penulis mengharapkan kritik dan saran membangun dari semua pihak agar dapat lebih baik dimasa yang akan datang.

Pada kesempatan ini disampaikan terima kasih kepada pada Orang tua yang selalu memberikan dukungan dalam bentuk apapun dengan tulus. Tidak lupa kepada Bapak Djoko Suprajitno Bapak Ciptian Weried, dan Bapak Sopar selaku pembimbing yang sudah memberi bimbingan untuk terwujudnya tugas akhir ini. Selanjutnya terima kasih kepada Diptiya Regita yang sudah memberikan semangat agar segera menyelesaikan studi. Untuk terakhir kali, terima kasih untuk berbagai pihak yang turut membantu dalam pengerjaan tugas akhir ini baik yang secara langsung maupun tidak langsung, yang penulis tidak dapat menyebutkannya satu persatu.

Penulis Berharap buku Tugas Akhir ini dapat memberikan informasi dan manfaat bagi pembaca pada umumnya dan mahasiswa Departemen Teknik Elektro Otomasi pada khususnya.

Surabaya, Juli 2018

Penulis

-----Halaman ini sengaja dikosongkan-----

DAFTAR ISI

HALAMAN JUDUL	i
HALAMAN JUDUL.....	iii
PERNYATAAN KEASLIAN TUGAS AKHIR Error! Bookmark not defined.	
HALAMAN PENGESAHAN INSTITUT.....	vii
ABSTRAK.....	ix
<i>ABSTRACT</i>	xi
KATA PENGANTAR.....	xiii
DAFTAR ISI.....	xv
DAFTAR GAMBAR.....	xvii
DAFTAR TABEL.....	xix
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang.....	1
1.2 Permasalahan.....	2
1.3 Batasan Masalah.....	2
1.4 Tujuan.....	2
1.5 Metodologi Penelitian.....	2
1.6 Sistematika Laporan.....	3
1.7 Relevansi.....	4
BAB II MODIFIKASI SISTEM <i>AIR PRESSURE BOOSTER</i>	5
2.1 <i>Air Pressure Booster</i>	5
2.2 Metode <i>Least Square</i>	5
2.3 Sensor Tekanan Udara.....	6
2.4 <i>Control Valve</i>	7
2.5 <i>Voltage to Current</i>	8
2.6 Mikrokontroler AVR ATmega16.....	8

2.7 LabVIEW.....	10
BAB III PERANCANGAN MODIFIKASI SISTEM.....	15
3.1 Perancangan Modifikasi Sistem <i>Air Pressure Booster</i>	15
3.1.1 Algoritma Kontrol Tekanan.....	17
3.1.2 Algoritma <i>Input Setpoint</i>	18
3.1.3 Algoritma Komunikasi Data.....	18
3.2 Perancangan <i>Hardware</i>	19
3.2.1 Perancangan <i>Panel Box</i>	20
3.2.2 Perancangan Modul Kontroler.....	20
3.3 Perancangan HMI (<i>Human Machine Interface</i>).....	24
3.3.1 Algoritma HMI.....	24
3.3.2 Perancangan Visual HMI.....	24
3.4 Identifikasi <i>Plant</i> (Tangki Udara).....	26
BAB IV PENGUJIAN KONTROLER, HMI, DAN TANGKI.....	27
4.1 Pengujian Rangkaian Modul Kontroler.....	27
4.1.1 Pengujian Modul ATmega16.....	27
4.1.2 Pengujian <i>Converter V-I</i>	29
4.2 Pengujian HMI(<i>Human Machine Interface</i>).....	30
4.3 Pengujian Tangki Udara (<i>Plant</i>).....	30
BAB V PENUTUP.....	35
DAFTAR PUSTAKA.....	37
LAMPIRAN.....	39
RIWAYAT HIDUP PENULIS.....	73

DAFTAR GAMBAR

Gambar 2.1 Sensor Tekanan Udara.....	7
Gambar 2.3 <i>Positioner Control Valve</i> YT – 1000 RDn531.....	7
Gambar 2.4 Rangkaian RC <i>Filter</i>	8
Gambar 2.5 Rangkaian Penguat <i>Non-inverting</i>	8
Gambar 2.10 Susunan Pin ATmega16.....	9
Gambar 2.11 <i>Front Panel</i>	11
Gambar 2.12 Blok Diagram dari Vi.....	12
Gambar 2.13 <i>Control Palette</i>	12
Gambar 2.14 <i>Functions pallete</i>	13
Gambar 3.1 Modifikasi Sistem <i>Air Pressure Booster</i>	15
Gambar 3.2 Ilustrasi Modifikasi Sistem <i>Air Pressure Booster</i>	16
Gambar 3.3 Skema Sistem Kontrol Tekanan Udara.....	16
Gambar 3.4 <i>Flowchart</i> Kontrol Tekanan Udara.....	17
Gambar 3.5 <i>Flowchart Input Setpoint</i>	18
Gambar 3.6 <i>Flowchart</i> Komunikasi Data.....	19
Gambar 3.7 <i>Panel Box</i>	20
Gambar 3.8 Skematik Modul ATmega16.....	21
Gambar 3.9 Rangkaian RC <i>Filter</i>	22
Gambar 3.10 Rangkaian Penguat <i>Non-inverting</i>	23
Gambar 3.11 HMI LabVIEW.....	25
Gambar 3.12 <i>Flowchart</i> Program HMI.....	26
Gambar 3.13 Diagram Identifikasi <i>Plant</i>	26
Gambar 3.14 Tangki Udara.....	26
Gambar 4.1 Modul ATmega16.....	28
Gambar 4.2 Susunan <i>Keypad</i>	28

Gambar 4.3 Tampilan Utama Awal.....	28
Gambar 4.4 Tampilan Mengubah <i>Setpoint</i>	29
Gambar 4.5 Tampilan Utama.....	29
Gambar 4.6 Diagram Instalasi Pengujian <i>Converter V-L</i>	29
Gambar 4.7 Pengujian HMI Alat Kontrol.....	30
Gambar 4.8 Grafik Respon Sistem (Percobaan 1).....	31
Gambar 4.9 Grafik Respon Sistem (Percobaan 2).....	32
Gambar 4.10 Grafik Respon Sistem (Percobaan 3).....	32
Gambar 4.11 Grafik Respon Sistem (Percobaan 4).....	33
Gambar 4.12 Grafik Respon Sistem (Percobaan 5).....	34

DAFTAR TABEL

Tabel 2.1 Data Kerja Pompa <i>Air Pressure Booster</i>	5
Tabel 4.1 Data Pengujian <i>Converter V-I</i>	29

-----Halaman ini sengaja dikosongkan-----

BAB I

PENDAHULUAN

1.1 Latar Belakang

PDSI merupakan anak perusahaan dari PT. Pertamina (Persero) yang telah beroperasi selama lebih dari sepuluh tahun. Perusahaan ini bergerak dalam bidang eksplorasi dan eksploitasi pengeboran minyak dan gas bumi, serta panas bumi. PDSI juga memberikan pelayanan service pengeboran yang terintegrasi. PDSI berhubungan erat dengan pengeboran minyak gas dan bisnis lain yang berkaitan dengan masalah pengeboran dan bekerjasama dalam pemenuhan atas kebutuhan yang berkaitan dengan eksplorasi dan eksploitasi minyak, gas dan panas bumi di segala area, pengeboran di darat maupun lepas pantai. sejak tahun 2016 PDSI telah mengembangkan tidak kurang dari 24 unit usaha baru di luar bisnis rig, yang masih terkait dengan kegiatan pengeboran. Pada proses produksi minyak dan gas bumi terdapat usaha yang dinamakan pengeboran. Tujuan pengeboran adalah mencari cadangan-cadangan hidrokarbon untuk meningkatkan produksi minyak dan gas bumi. Terdapat dua jenis pengeboran yaitu eksplorasi dan eksploitasi. Pengeboran eksploitasi adalah pengeboran yang dilakukan pada lokasi tertentu berdasarkan data-data yang didapatkan dari pengeboran eksplorasi. Pengeboran eksplorasi adalah pengeboran yang dilakukan pada lokasi tertentu tanpa adanya data-data [1].

Proses pengeboran minyak dilakukan dengan menggunakan sekumpulan peralatan yang disebut rig. Rig mempunyai lima sistem utama yaitu sistem pengangkat (*hoisting system*), sistem pemutar (*rotating system*), sistem sirkulasi (*circulating system*), sistem tenaga (*power system*), dan sistem pencegah semburan liar (*BOP system*) [1].

Sistem pengangkatan (*hoisting system*) berfungsi untuk menaikkan dan menurunkan rangkaian pipa bor, *casing* serta peralatan-peralatan lainnya. Terdapat unit mesin penting pada sistem pengangkat yaitu *drawwork*. *Drawwork* berfungsi sebagai katrol pada sistem ini. *Drawwork* membutuhkan suplai udara sebagai pengatur membuka dan menutup spring pada *disbrake-nya*. Ketika tekanan udara mencapai 175 psi, maka *spring* akan terbuka. Sebaliknya, jika tekanan udara kurang dari 165 psi, maka *spring* akan menutup. Proses suplai udara ke *drawwork* terdapat pada sistem udara (*air system*).

Dalam proses suplai udara untuk *drawwork*, udara melewati beberapa tahapan mulai dari *air compressor*, *air dryer*, *air receiver*

tank, *air pressure booster*, dan *brake drawwork*. Pada sistem air ini terdapat kekurangan pada *air pressure booster*. *Air pressure booster* adalah sistem yang bertujuan menaikkan tekanan udara. Kekurangan pada sistem *air pressure booster* yaitu pompa bekerja terus-menerus selama *drawwork* digunakan. Permasalahan ini timbul dikarenakan tidak adanya alat pengontrol tekanan udara pada sistem *air pressure booster*.

Penelitian ini bertujuan untuk menyelesaikan permasalahan yang terdapat pada sistem *air pressure booster* dengan membuat suatu alat. Alat ini nantinya dapat mengatur pergantian penggunaan pompa *air pressure booster*, kontrol tekanan udara dan monitoring tekanan pada *tank reservoir*.

1.2 Permasalahan

Pompa pada sistem *air pressure booster* masih bekerja secara manual dengan mengatur *ball valve* yang terdapat pada tiap pompa booster. Pada kondisi lapangan yang ada, *ball valve* dibuka terus dikarenakan area pompa ini termasuk berbahaya. Pada sistem *air pressure booster* terdapat *relief valve* yang akan membuat tekanan udara apabila melebihi batas yang sudah ditentukan. Akibatnya, pompa bekerja terus menerus melakukan siklus *charge* dan *discharge* yang membuat kerusakan pada pompa.

1.3 Batasan Masalah

Batasan masalah dalam tugas akhir ini yaitu Mengatur kerja dari 3 pompa *booster* menggunakan *control valve* dengan alat berbasis mikrokontroler.

1.4 Tujuan

Penelitian bertujuan untuk mengatur tekanan, sehingga tidak terjadi pemborosan proses *charge* dan *discharge* pada *tank reservoir*. Maintenance pada pompa akibat kerusakan akan berkurang karena proses *charge* dan *discharge* akan lebih efisien. Pengawasan tekanan udara pada *tank reservoir* akan lebih mudah karena terdapat sensor tekanan.

1.5 Metodologi Penelitian

Dalam proses Rancang bangun alat kontrol tekanan pada *Air Pressure Booster* untuk Pengereman *Drawwork* dalam Proses

Pengeboran terdapat beberapa tahap. Tahapan yang akan dilakukan antara lain pengamatan permasalahan, studi literatur, konsultasi atau diskusi dengan dosen pembimbing, perancangan alat, uji coba dan analisis data, dan penyusunan laporan.

Tahapan Pengamatan dilakukan untuk memperoleh data mengenai pada sistem kerja *drawwork*. Dalam hal ini, kami telah melakukan pengambilan data mengenai *air pressure booster* pada *drawwork* pada saat kerja praktek di *PT. Pertamina Drilling Services Indonesia* tepatnya di lokasi SKR-001 RIG PDSI #42.3/N1500-E, Muara Sungai, Prabumulih, Sumatera Selatan. Selanjutnya tahap pencarian literatur yang relevan dan dapat dijadikan landasan mengenai alat kontrol tekanan. Studi literatur meliputi materi mengenai *air supply system* pada *drawwork*, *control valve*, kontroler, dan mikrokontroler. Beberapa sumber literatur kami dapatkan melalui *manual book rig* terkait ataupun rig secara umum pada *PT. Pertamina Drilling Services Indonesia*, internet, dan juga beberapa jurnal-jurnal mengenai *control valve*, dan mikrokontroler. Tahapan perancangan dilakukan setelah mendapat informasi dari referensi dan arahan dari dosen pembimbing. Setelah diselesaikannya perancangan alat, dilakukanlah uji coba terhadap alat serta membandingkan dengan kondisi *real* di lapangan. Hal ini bertujuan untuk memperoleh beberapa data yang nantinya dianalisis untuk mencari data yang benar-benar *valid* serta dapat mengetahui kekurangan dari alat yang telah dibuat. Yang terakhir penyusunan laporan dilakukan setelah didapat data mengenai alat yang telah dibuat. Laporan ini sebagai bukti pertanggung jawaban atas penelitian yang telah dilakukan.

1.6 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, metodologi penelitian, sistematika laporan dan relevansi.

Bab II Teori Dasar

Bab ini menjelaskan tentang tinjauan pustaka tentang teori-teori pendukung, konsep dasar kontroler, *control valve*, dan pengenalan perangkat lunak.

Bab III Perancangan dan Implementasi

Bab ini membahas tentang instalasi dan implementasi kontroler pada pengaturan *control valve*. Pengaturan *control valve* bertujuan untuk mengatur tekanan udara pada *tank reservoir*.

Bab IV Pengujian dan Analisis

Bab ini memuat hasil pengukuran dan analisa program *keypad*, *converter V/I* dan tangki udara.

Bab V Penutup

Bab ini berisi kesimpulan dan saran dari hasil pembahasan yang telah diperoleh.

1.7 Relevansi

Hasil yang diperoleh dari Tugas Akhir ini dapat menyelesaikan permasalahan *air pressure booster* pada *air supply system* untuk *drawwork* dalam proses pengeboran. Alat ini diharapkan dapat membuat kerja pompa pada sistem *air pressure booster* menjadi lebih efisien dan mengurangi *maintenance* dari pompa akibat kerusakan.

BAB II

MODIFIKASI SISTEM *AIR PRESSURE BOOSTER*

2.1 *Air Pressure Booster*

Dalam sistem *air pressure booster* untuk pengereman *drawwork* pada proses pengeboran terdapat pompa yang berfungsi sebagai penaik tekanan. Pompa yang terdapat pada sistem *air pressure booster* akan bekerja saat mendapat masukkan udara bertekanan. Tekanan udara yang masuk dapat diatur dengan menggunakan *control valve*. Sehingga secara tidak langsung *control valve* dapat mengatur tekanan pada *tank reservoir*. Pada Lampiran C sistem *air pressure booster* yang terdapat pada manual book rig PDSI #42.3/N1500-E [1].

Pada penelitian kali ini, akan dilakukan perancangan alat kontrol yang dapat mengatur kerja dari kontrol sehingga juga dapat mengatur tekanan secara otomatis. Alat ini dapat digunakan untuk mengatur tekanan udara otomatis dan mengatur perpindahan kerja dari pompa.

Karena pada penelitian ini tidak dapat mempraktikan pada sistem *air pressure booster* secara langsung, maka akan dilakukan demo dengan mengacu berdasarkan data yang sudah didapat (tabel 2.1). Data tersebut diambil dari pengisian 165-175 PSI. Beberapa yang dibutuhkan untuk melakukan demo diantaranya alat kontrol, tangki, code vision, dan LabVIEW.

Tabel 2.1 Data Kerja Pompa *Air Pressure Booster*

<i>No</i>	<i>Ball Valve</i>	<i>Time Charge</i>	<i>Stroke</i>	<i>Discharge with pump</i>	<i>Discharge</i>	<i>Time/ Stroke</i>	<i>Time Charge & Discharge</i>	<i>Stroke/ Cycling</i>
1	50	51	28	75	28	1.8	126	1.8
2	100	33	25	102		1.3	135	1.3

2.2 **Metode Identifikasi Sistem**

Identifikasi sistem pada penelitian ini menggunakan metode *least square* sebagai metode mendapatkan model dari sistem dan metode evaluasi *root mean squared error* sebagai evaluasi akurasi dari hasil model *least square*.

- Metode *Least Square*

Metode *Least Square* sering digunakan untuk pemodelan regresi dari titik-titik data diskrit. Metode ini termasuk dalam golongan metode pendekatan berdasarkan distribusi eror yang didapat bersama dengan interval keseluruhan data. Metode least square digunakan untuk mendapat persamaan matematis [2].

Metode least square termasuk dalam metode pendekatan eksperimen. Metode eksperimen adalah metode pembentukan model matematis dari sebuah sistem fisik berdasarkan observasi dari sistem. Metode ini digunakan dengan mencatat hubungan dari masukan dan keluaran dari sistem [2].

- Metode *Root Mean Squared Error* [3]

Akar kesalahan kuadrat rata-rata (root mean square error) adalah metode untuk evaluasi teknik *regresi* yang digunakan untuk mengukur tingkat akurasi hasil model. *RMSE* merupakan nilai rata-rata dari jumlah kuadrat kesalahan. *RMSE* dapat menyatakan besarnya kesalahan yang dihasilkan oleh suatu model. Nilai *RMSE* rendah menunjukkan bahwa model prakiraan mendekati model sebenarnya. [3]

2.3 Sensor Tekanan Udara

Sensor adalah transduser yang digunakan untuk mengubah nilai ukur suatu keadaan fisik menjadi besaran listrik. Sensor sebagai alat ukur menjadikannya salah satu komponen penting dalam sistem pengendali. Sensor seolah memberikan indra pada kontroller sehingga dapat mengukur suatu keadaan [4].

Pada penelitian ini, variabel proses yang dikontrol adalah tekanan udara yang terdapat dalam suatu tangki udara. Sensor yang digunakan pada penelitian ini terlihat pada Gambar 2.1, yang memiliki sebagai berikut :

- Rentang temperatur kerja : -40 C ~ 125 C
- Rentang tekanan kerja : 0~174 PSI
- *Response time* : < 2ms
- Suplay tegangan : 4.75V~5.25V
- Rentang keluaran : 0V~5V
- Maksimum tekanan : 300 PSI
- Maksimum tegangan : 8.75V



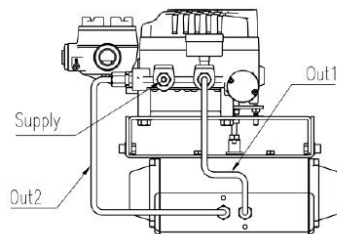
Gambar 2.1 Sensor Tekanan Udara

2.4 Control Valve

Control valve adalah alat yang digunakan untuk mengatur aliran fluida (gas, air, uap panas (*steam*), dan bahan - bahan kimia) sehingga controller dapat menjaga variabel proses tetap berada pada set point. *Control valve* mengatur aliran fluida dengan merubah persentase buka valve. *Control valve* terdiri atas dua bagian dasar yang meliputi aktuator dan valve dapat dilihat pada Gambar 2.3. *Valve* adalah komponen mekanis yang menentukan besarnya aliran fluida sedangkan Aktuator adalah bagian yang mengerjakan gerak buka tutup *valve* [5].

Pada penelitian ini, jenis aktuator mengacu pada *positioner control valve* YT – 1000 RDn531. *Positioner control valve* jenis ini memiliki *input* untuk mengatur persentase buka valve dengan arus DC. Spesifikasi *positioner control valve* YT – 1000 RDn531 sebagai berikut :

- Rentang temperatur kerja : - 40 ~ 70c.
- Tekanan suplay : 1,4-7 bar.
- Sinyal input : 4-20 mA
- Arus maksimum : 120 mA.
- Suplay Tegangan : 9~28 VDC



Gambar 2.3 Positioner Control Valve YT – 1000 RDn531

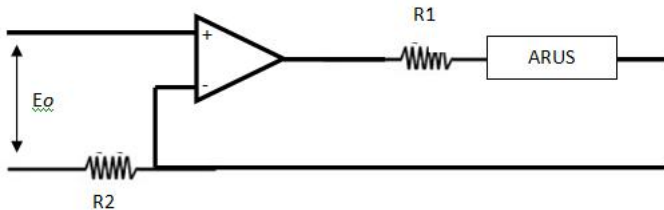
2.5 Voltage to Current

Voltage to Current adalah converter yang digunakan untuk mengubah Tegangan ke Arus. *Pulse Width Modulation* (PWM) adalah salah satu teknik mengubah lebar pulsa dengan frekuensi dan amplitudo yang tetap. Arus adalah banyaknya muatan listrik yang mengalir melalui suatu titik dalam sirkuit listrik tiap satuan waktu.

Pada penelitian ini, *converter* tegangan-arus dibuat agar sinyal kontrol dari kontroler dapat mengatur persentase buka *valve*. *Converter* ini terdiri dari 2 rangkaian yaitu, rangkaian *RC Filter* (Gambar 2.4) dan rangkaian penguat *non-inverting* (Gambar 2.5).



Gambar 2.4 Rangkaian RC Filter



Gambar 2.5 Rangkaian Penguat Non-inverting

Tujuan dari dua rangkaian pada Gambar 2.4 dan Gambar 2.5 adalah merubah keluaran dari mikrokontroler menjadi arus. Rangkaian pada Gambar 2.4 berfungsi untuk menstabilkan tegangan keluaran dari pin PWM mikrokontroler. Rangkaian pada Gambar 2.5 berfungsi untuk mengubah nilai besaran tegangan 0V-5V menjadi besaran arus 0mA-20mA.

2.6 Mikrokontroler AVR ATmega16

Mikrokontroler adalah suatu chip (IC) yang digunakan sebagai pengontrol utama sistem elektronika. Dalam mikrokontroler sudah terdapat unit pemroses, memory ROM (*Read Only Memory*), RAM (*Random Access Memory*), *Input - Output*, dan fasilitas pendukung

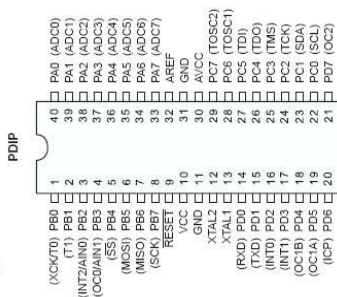
lainnya. Pada mikrokontroler AVR juga sudah terdapat ADC internal [6].

AVR dapat dikelompokkan menjadi empat kelas, yaitu keluarga ATtiny, keluarga AT90Sxx, keluarga ATmega, dan keluarga AT86RFxx. Pada dasarnya yang membedakan masing – masing kelas adalah memory, peripheral dan fungsinya.

Mikrokontroler AVR (*Alf and Vegard's Risc processor*) merupakan pengontrolan utama standar industri dan riset saat ini. Hal ini dikarenakan berbagai kelebihan yang dimilikinya dibandingkan mikroprosesor antara lain lebih murah, dukungan *software* dan dokumentasi yang memadai dan memerlukan komponen pendukung yang sangat sedikit. Salah satu tipe mikrokontroler AVR untuk aplikasi standar yang memiliki fitur memuaskan ialah ATmega16.

AVR merupakan seri dari mikrokontroler CMOS 8-bit buatan Atmel, berbasis arsitektur RISC (Reduced Instruction Set Computer). Hampir semua instruksi dieksekusi dalam satu siklus clock. AVR mempunyai 32 register general-purpose, timer/counter fleksibel dengan mode compare, interrupt (internal dan eksternal), serial UART, ADC dan PWM internal. AVR juga mempunyai InSystem Programmable Flash on-chip yang memungkinkan memori program untuk diprogram ulang dalam sistem menggunakan hubungan serial SPI.

Pada penelitian ini, mikrokontroler yang digunakan adalah ATmega16. Penggunaan mikrokontroler ATmega 16 karena sistem yang akan dibuat membutuhkan 3 output analog, 1 input analog, 8 input digital, dan komunikasi serial. Deskripsi *Pinout* pada mikrokontroler AVR ATmega16 dengan kemasan 40-Pin DIP (*Dual In line Package*). Susunan *port* dan fungsi dari tiap pin ATmega16 dapat dilihat pada Gambar 2.10.



Gambar 2.10 Susunan Pin ATmega16

Berikut penjelasan tiap pin ATmega16:

- VCC (*Digital power supply*) merupakan pin masukan positif catu daya.
- GND (*ground*) sebagai pin *Ground*.
- Port A (PA7..PA0)
Port A berfungsi sebagai *input* analog pada A/D Konverter. *Port A* juga berfungsi sebagai suatu Port I/O 8-bit dua arah, jika A/D Konverter tidak digunakan. Pin - pin *Port* dapat menyediakan resistor *internal pull-up* (yang dipilih untuk masing-masing bit).
- Port B (PB7..PB0)
Port B adalah suatu *Port I/O* 8-bit dua arah dengan resistor *internal pull-up* (yang dipilih untuk beberapa bit).
- Port C (PC7..PC0)
Port C adalah suatu *Port I/O* 8-bit dua arah dengan resistor *internal pull-up* (yang dipilih untuk beberapa bit).
- Port D (PD7..PD0)
Port D adalah suatu *Port I/O* 8-bit dua arah dengan resistor *internal pull-up* (yang dipilih untuk beberapa bit). Pada *Port D* juga terdapat fasilitas komunikasi serial dan PWM yang disediakan oleh mikrokontroler ATmega16.
- RESET (*Reset input*) merupakan pin yang digunakan untuk me-*reset* mikrokontoler.
- XTAL1 (*Input Oscillator*) dan XTAL2 (*Output Oscillator*) sebagai pin masukan *clock* eksternal.
- AVCC adalah pin penyedia tegangan untuk *port A* dan A/D Konverter (ADC)
- AREF adalah pin masukan tegangan referensi analog untuk A/D konverter (ADC) pada *Port A*.

2.7 LabVIEW

LabVIEW (*Laboratory Virtual Instrument Engineering Workbench*) adalah bahasa pemrograman grafis atau blok yang menggunakan ikon bukan baris teks untuk membuat aplikasi. *Software*

ini digunakan khusus untuk pemrosesan serta visualisasi data dalam bidang akuisisi data, kendali, instrumentasi, dan otomatisasi industri. Berbeda dengan bahasa pemrograman berbasis teks, di mana petunjuk menentukan eksekusi program [7].

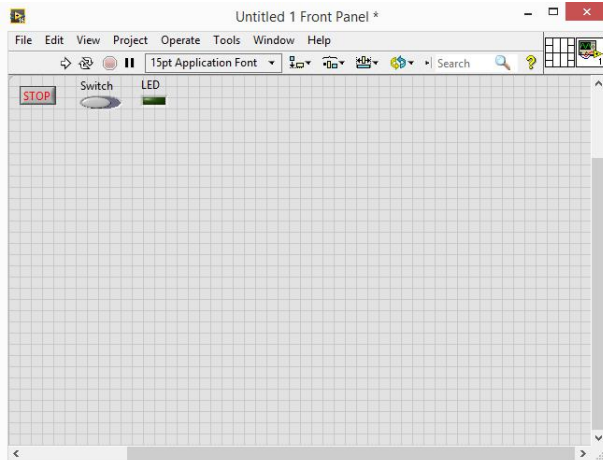
LabVIEW menggunakan *dataflow* pemrograman, di mana aliran data menentukan eksekusi. LabVIEW dapat digunakan untuk mengontrol aliran data antara komputer dan instrumen, memiliki banyak fitur penting untuk pemrosesan sinyal dan analisis.

Pada labVIEW, *user* pertama membuat *user interface* atau *front panel* dengan menggunakan komponen *control* dan indikator. Kontrol pada labVIEW adalah *knobs*, *push buttons*, *dials* dan peralatan input lainnya, sedangkan indikator adalah *graphs*, LEDs dan peralatan *display* lainnya. Setelah *user interface* tersusun, *user* menyusun blok diagram yang berisi kode-kode VIs untuk mengontrol *front panel*.

Software LabVIEW terdiri dari tiga komponen utama, yaitu :

2.7.1 Front Panel

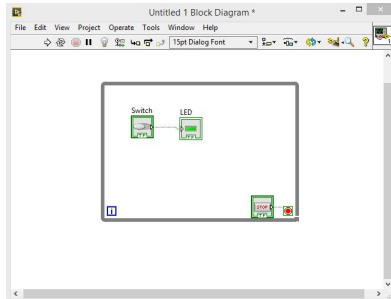
Front panel adalah bagian *window* yang berlatar belakang abu-abu serta mengandung *control* dan indikator. *front panel* digunakan untuk membangun sebuah VI, menjalankan program dan *men-debug* program. Tampilan dari *front panel* dapat di lihat pada Gambar 2.11



Gambar 2.11 Front Panel

2.7.2 Blok diagram dari Vi

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 2.12.



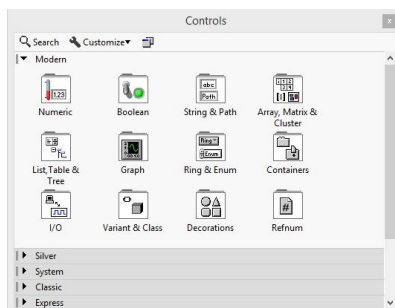
Gambar 2.12 Blok Diagram dari Vi

2.7.3 Control dan Functions Palette

Control dan *Functions Palette* digunakan untuk membangun sebuah Vi.

2.7.3.1 Control Palette

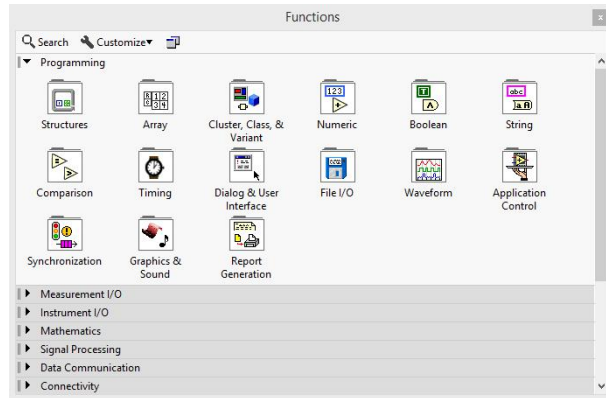
Control Palette merupakan tempat beberapa *control* dan indikator pada *front panel*, *control palette* hanya tersedia di *front panel*, untuk menampilkan *control palette* dapat dilakukan dengan mengklik *windows >> show control palette* atau klik kanan pada *front panel*. Contoh *control palette* ditunjukkan pada Gambar 2.13.



Gambar 2.13 Control Palette

2.7.3.2 Functions Palette

Functions Palette digunakan untuk membangun sebuah blok diagram, *functions palette* hanya tersedia pada blok diagram, untuk menampilkannya dapat dilakukan dengan meng-klik **windows >> show control palette** atau klik kanan pada lembar kerja blok diagram. Contoh dari *functions palette* ditunjukkan pada Gambar 2.14.



Gambar 2.14 *Functions palette*

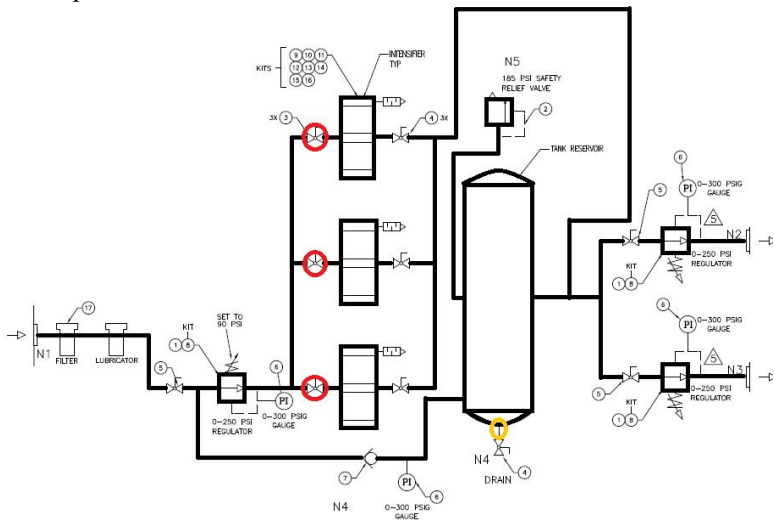
-----Halaman ini sengaja dikosongkan-----

BAB III

PERANCANGAN MODIFIKASI SISTEM

3.1 Perancangan Modifikasi Sistem *Air Pressure Booster*

Pada penelitian ini, bertujuan memodifikasi sistem *air pressure booster* agar menjadi lebih baik dalam prosesnya. Modifikasi dilakukan dengan mengganti komponen penyusun sistem dan menambah beberapa komponen. Modifikasi yang dilakukan dapat dilihat pada Gambar 3.1.

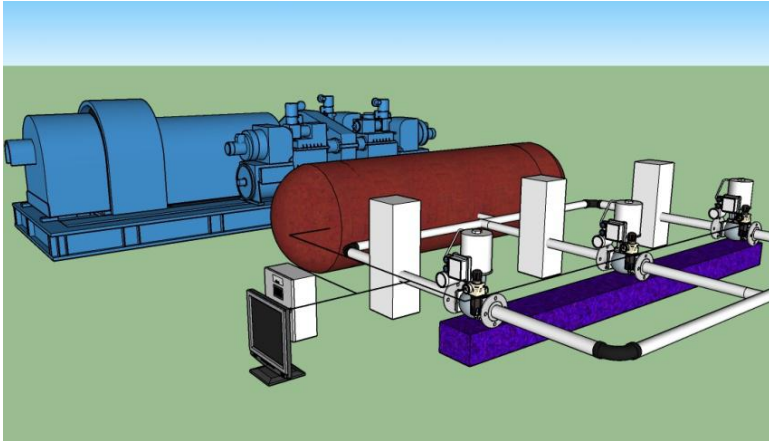


Gambar 3.1 Modifikasi Sistem *Air Pressure Booster*

Pada Gambar 3.1 daerah yang ditandai warna merah dan warna kuning adalah yang akan dimodifikasi. Daerah yang ditandai warna merah adalah *ball valve* yang diganti dengan *control valve*. Daerah yang ditandai warna kuning adalah daerah yang akan ditambah sensor tekanan udara.

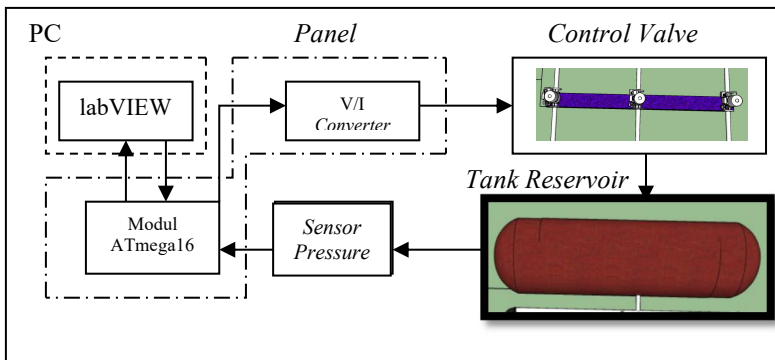
Proses variabel yang dikontrol adalah tekanan udara pada *tank reservoir*. Tekanan udara pada *tank reservoir* dapat berubah dengan membuka suplay udara pada pompa. Pompa dapat menaikkan tekanan udara dengan cepat atau lambat sesuai suplay udara yang masuk.

Suplay udara pada pompa diatur dengan menggunakan *control valve*, sehingga dapat diilustrasikan seperti terlihat pada Gambar 3.2.



Gambar 3.2 Ilustrasi Modifikasi Sistem Air Pressure Booster

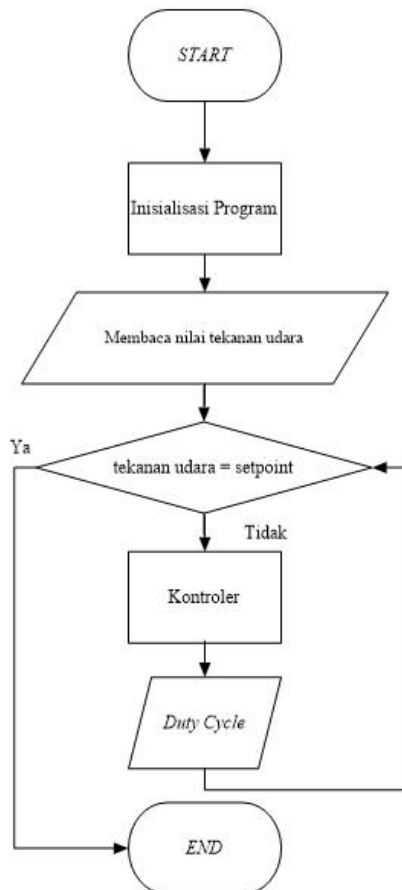
Dari Gambar 3.2, maka proses kontrol tekanan udara dapat dilakukan dengan membangun sistem dengan skema seperti pada Gambar 3.3.



Gambar 3.3 Skema Sistem Kontrol Tekanan Udara

3.1.1 Algoritma Kontrol Tekanan

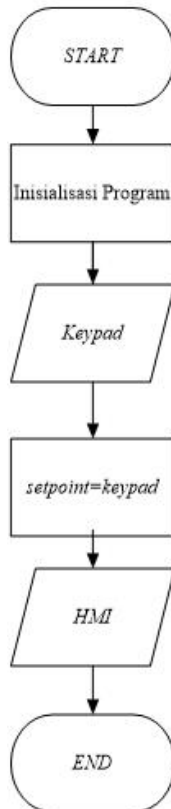
Dalam proses kontrol tekanan, sensor memberikan nilai ukur tekanan udara lalu diterima mikrokontroler. Mikrokontroler akan mengirim sinyal kontrol ke aktuator setelah diproses dengan metode kontrol. Algoritma kontrol tekanan udara dalam mikrokontroler adalah alur mikrokontroler melakukan aksi kontrol. Algoritma kontrol tekanan oleh mikrokontroler dapat dilihat pada Gambar 3.4.



Gambar 3.4 Flowchart Kontrol Tekanan Udara

3.1.2 Algoritma *Input Setpoint*

Dalam proses kontrol tekanan udara, mikrokontroler akan melakukan aksi kontrol. Variabel yang dapat berubah selain nilai ukur sensor adalah variabel *setpoint*. Variabel *setpoint* diubah sesuai dengan tekanan udara yang diinginkan. Algoritma input nilai *setpoint* dapat dilihat pada Gambar 3.5.



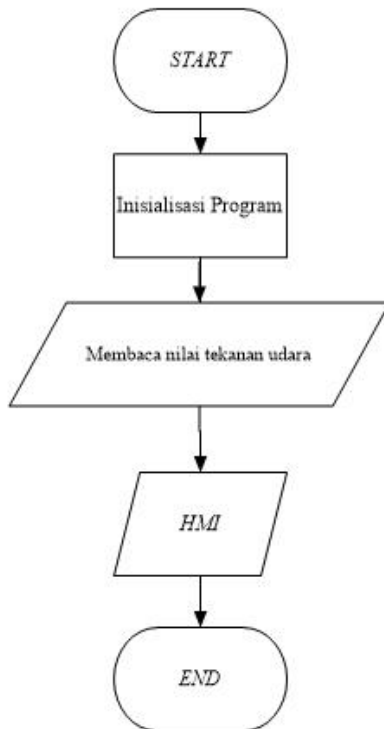
Gambar 3.5 *Flowchart Input Setpoint*

3.1.3 Algoritma Komunikasi Data

Dalam pengawasan kerja dari mikrokontroler dibutuhkan komponen antarmuka untuk menampilkan variabel proses. Komponen

antarmuka yang digunakan pada penelitian ini adalah PC/Laptop dengan software labVIEW. PC/Laptop membutuhkan komponen penghubung ke mikrokontroler agar dapat menerima data.

Pada penelitian ini, digunakan komunikasi serial RS-485 untuk penghubung antara PC/Laptop dan Mikrokontroler. *Flowchart* pengiriman data dari mikrokontroler ke PC/Laptop terlihat pada Gambar 3.6.



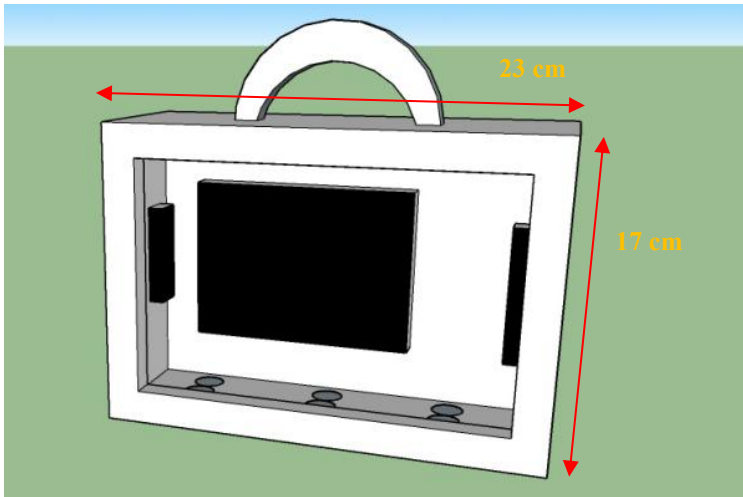
Gambar 3.6 *Flowchart* Komunikasi Data

3.2 Perancangan *Hardware*

Perancangan *Hardware* ini terdiri dari 2 bagian yaitu perancangan *panel box* , dan perancangan modul.

3.2.1 Perancangan *Panel Box*

Perancangan *Panel Box* bertujuan sebagai tempat modul rangkaian sistem minimum ATmega16, *power supply* dan rangkaian RS-485. *Box* ini dibuat dengan bahan besi karena di ilustrasikan daerah pengeboran membutuhkan bahan yang kuat. Rancangan *panel box* terlihat pada Gambar 3.7.



Gambar 3.7 *Panel Box*

3.2.2 Perancangan Modul Kontroler

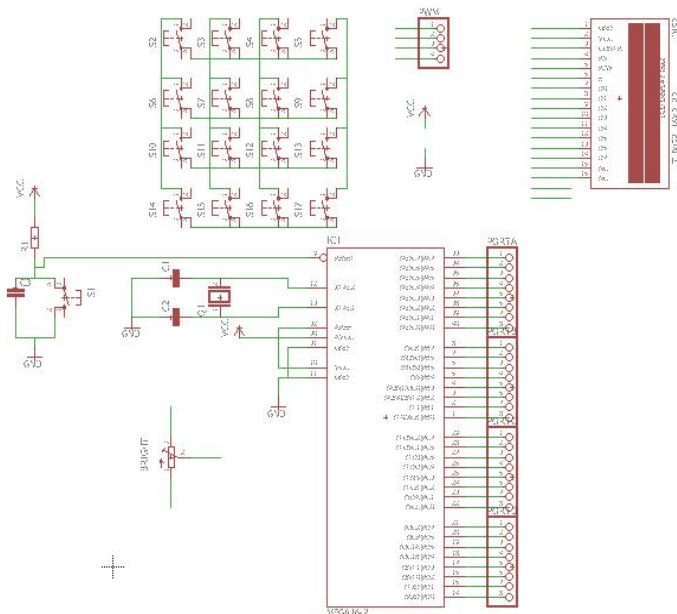
Pada perancangan modul kontroler selain minimum sistem ATmega16 dibuat juga rangkaian tambahan untuk LCD 16x2, konverter V-I dan PORT komunikasi serial.

3.2.2.1 Modul ATmega16

Perancangan sistem minimum mikrokontroler bertujuan agar IC mikrokontroler dapat beroperasi. Suatu mikrokontroler membutuhkan dua elemen (selain *power supply*) untuk berfungsi yaitu kristal osilator dan rangkaian *reset*. Sistem minimum kemudian dihubungkan dengan rangkaian lain pada modul.

Pada tugas akhir ini menggunakan rangkaian sistem minimum ATmega16 yang berfungsi untuk pengendali sistem kontrol tekanan udara. Dengan fitur-fitur dan jumlah memori yang dimiliki, membuat

ATmega16 cocok untuk penelitian ini. Kristal dan dua buah kapasitor sebagai osilator rangkaian sistem minimum yang berfungsi untuk memenuhi kebutuhan *clock* pada ATmega16. Dengan ditambahkan kristal sebagai *clock* eksternal, membuat ATmega16 dapat mengeksekusi program secara cepat. Rangkaian sistem minimum ATmega16 terlihat pada Gambar 3.8.

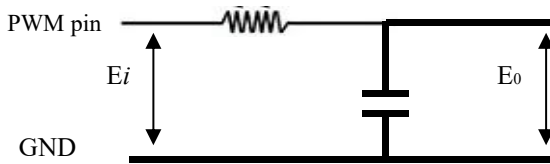


Gambar 3.8 Skematik Modul ATmega16

3.2.2.2 Konverter V-I

Rangkaian konverter ini terdiri dari 2 rangkaian yaitu, rangkaian *RC Filter* dan rangkaian penguat *non-inverting*. Rangkaian *RC* digunakan untuk mengubah besaran *PWM* menjadi nilai tegangan *analog*. Rangkaian penguat *non-inverting* digunakan untuk mengubah besaran tegangan menjadi arus. Untuk mendapatkan nilai komponen yang dibutuhkan agar rangkaian beroperasi sesuai yang diinginkan maka perlu dilakukan identifikasi sebagai berikut :

- Rangkaian RC Filter



Gambar 3.9 Rangkaian RC Filter

Dari Gambar 3.9 didapatkan Persamaan (3.1). dan setelah dilaplace, menjadi Persamaan (3.2).

$$Ei = IR + \frac{1}{C} \int i dt \dots\dots\dots (3.1)$$

$$Ei(s) = I(s)R + I(s) \frac{1}{Cs}$$

$$Ei(s) = I(s) \left(R + \frac{1}{Cs} \right)$$

$$I(s) = \frac{Ei(s)}{R + \frac{1}{Cs}}$$

$$Eo = \frac{1}{C} \int i dt$$

$$Eo(s) = \frac{1}{Cs} I(s)$$

$$I(s) = \frac{Eo(s)}{\frac{1}{Cs}}$$

$$\frac{Eo(s)}{Ei(s)} = \frac{\frac{1}{Cs}}{R + \frac{1}{Cs}} \dots\dots\dots (3.2)$$

Sehingga didapatkan fungsi alih,

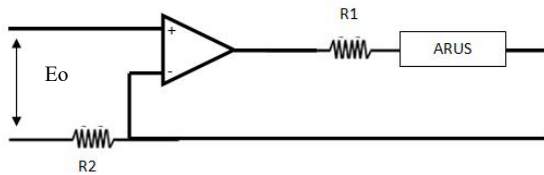
$$G(s) = \frac{1}{RCs+1} \dots\dots\dots(3.3)$$

dari Persamaan (3.3) agar tegangan keluaran sesuai dengan yang diinginkan, maka didapatkan nilai komponen:

$$R = \pm 56k\Omega$$

$$C = 10\mu F$$

- Rangkaian Penguat *Non-inverting*



Gambar 3.10 Rangkaian Penguat *Non-inverting*

Dari Gambar 3.10 didapatkan Persamaan (3.4). dan didapatkan fungsi alih Persamaan (3.5).

$$V_o = a.V_{in} \dots\dots\dots(3.4)$$

$$IR = a.V_{in}$$

$$\frac{I}{V_{in}} = \frac{a}{R_t}$$

$$\frac{I}{V_{in}} = \frac{\left(\frac{R1}{R2} + 1\right)}{R_t}$$

$$\frac{I}{V_{in}} = \frac{\left(\frac{R1 + R2}{R2}\right)}{R_t}$$

$$\frac{I}{V_{in}} = \frac{R_t}{R2} \times \frac{1}{R_t}$$

$$\frac{I}{V_{in}} = \frac{1}{R_2} \dots\dots\dots(3.5)$$

$$\frac{I}{V_{in}} = \frac{R_t}{R_2 \cdot R_t}$$

dari Persamaan (3.5) agar arus keluaran sesuai dengan yang diinginkan, maka didapatkan nilai komponen:

$$R_2 = 250\Omega$$

3.3 Perancangan HMI (*Human Machine Interface*)

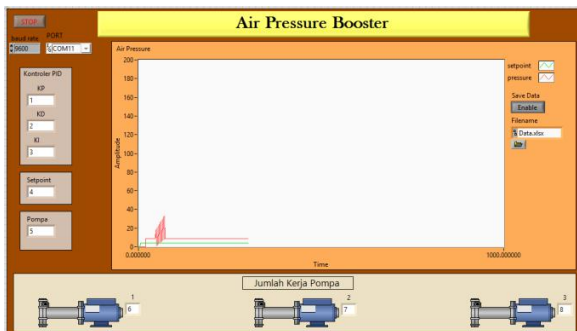
HMI merupakan suatu komponen penting dalam proses pengamatan atau monitoring dari kondisi yang akan diamati. HMI dibuat sebagai visualisasi dari kondisi yang sesuai dengan keadaan dilapangan atau secara real. Dalam perancangan HMI ini terbagi menjadi 2, yaitu Algoritma HMI dan Perancangan Visual HMI.

3.3.1 Algoritma HMI

Algoritma HMI merupakan alur program dalam proses akusisi data dari mikrokontroler. HMI akan mengakusisi data dari mikrokontroler lalu akan di visualisasi pada PC. *Flowchart* HMI menerima dan menampilkan data, dapat dilihat pada Gambar 3.12.

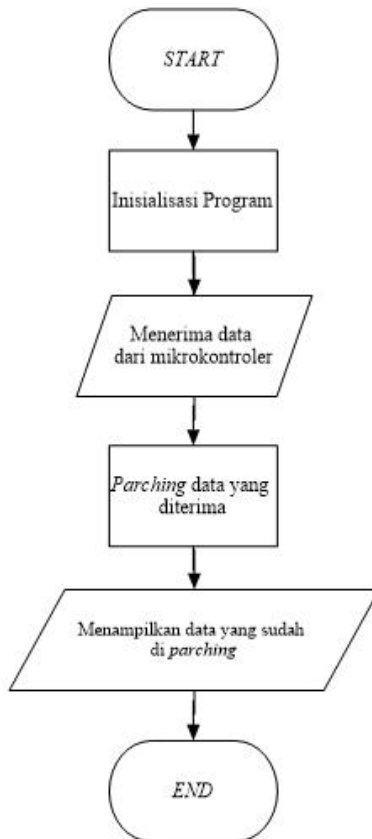
3.3.2 Perancangan Visual HMI

Visual HMI merupakan suatu perangkat lunak yang dapat memudahkan pengguna dalam pengamatan pada sistem. Gambar 3.11 merupakan bentuk dari perancangan visual HMI, yang dibuat pada *software* labVIEW.



Gambar 3.11 HMI LabVIEW

Pada Gambar 3.11 terdapat 1 indikator *graph*, 5 indikator *numeric*, 1 indikator *string*, dan 1 *visa resource name*. Indikator *graph* berguna untuk menampilkan grafik dari data sensor tekanan udara yang dikirim dari mikrokontroler. 5 indikator *numeric* berguna untuk menampilkan nilai variabel PID, *setpoint*, dan pompa yang bekerja pada sistem. 1 indikator *string* berfungsi untuk menampilkan data yang diterima hmi dari mikrokontroler. *Visa resource name* berfungsi untuk menentukan port mikrokontroler pada PC.

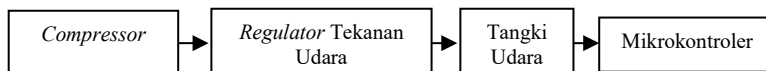


Gambar 3.12 *Flowchart* Program HMI

3.4 Identifikasi *Plant* (Tangki Udara)

Dalam melakukan proses pengendalian sangat penting untuk mengetahui karakteristik sistem atau peralatan yang akan dikontrol. Untuk mendapatkan karakteristik tersebut, diperlukan identifikasi tentang sistem agar didapatkan model matematis dari *plant*. Proses identifikasi yang dilakukan untuk mendapatkan respon sistem berubah menjadi *open-loop* dan menggunakan metode *least square*.

Plant (tangki udara Gambar 3.14) tekanan udara pada sistem ini disuplai dengan tekanan udara sebesar 0.2Mpa dengan diagram seperti pada Gambar 3.13.



Gambar 3.13 Diagram Identifikasi *Plant*



Gambar 3.14 Tangki Udara

BAB IV

PENGUJIAN KONTROLER, HMI, DAN TANGKI

Pada bab ini akan dibahas mengenai pengujian dari modul kontrol, pengujian *Human Machine Interface* (HMI), dan pengujian . Tujuan dilakukannya pengujian pada sistem ini untuk mengetahui keberhasilan dari keseluruhan sistem dan untuk mengetahui kesesuaian antara teori dengan hasil perancangan. Pengujian dilakukan pada masing-masing bagian terlebih dahulu, kemudian dari masing-masing bagian tersebut diuji secara keseluruhan. Setelah proses pengujian, langkah selanjutnya adalah menganalisa dari hasil pengujian tersebut.

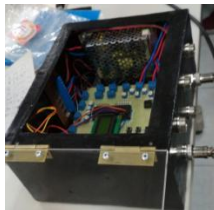
4.1 Pengujian Rangkaian Modul Kontroler

Rangkaian modul kontroler merupakan komponen penyusun rangkaian yang penting dari rancangan sistem. Pengujian rangkaian modul ini bertujuan untuk mengetahui keberhasilan dari tiap rangkaian penyusun yang telah dirancang untuk mendukung sistem yang dibuat.

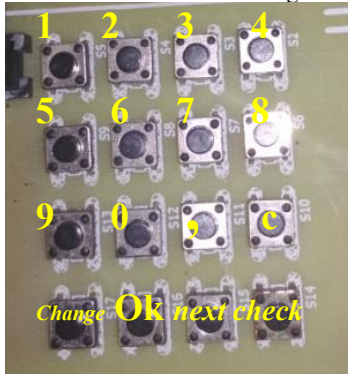
Pada pengujian ini meliputi beberapa pengujian terhadap rangkaian-rangkaian yang digunakan, yaitu pengujian modul ATmega16, pengujian *converter* V-I.

4.1.1 Pengujian Modul ATmega16

Modul ATmega16 merupakan rangkaian yang terdiri dari 3 bagian terlihat pada Gambar 4.1, yaitu minimum sistem ATmega16, Port LCD 16x2, dan *Keypad* (susunan *keypad* Gambar 4.2). Pada keypad angka digunakan untuk memasukkan nilai untuk variabel yang akan diganti. Pada pengujian ini dilakukan dengan menjalankan program dan memasukkan nilai *setpoint*. Berikut pengujian Modul ATmega16 :

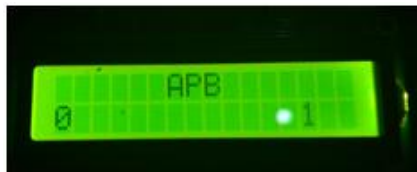


Gambar 4.1 Modul ATmega16



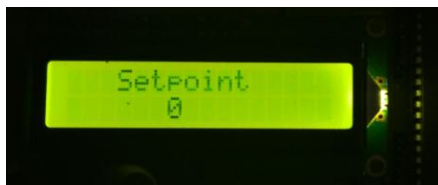
Gambar 4.2 Susunan Keypad

Tampilan awal pada LCD 16x2 tampak seperti Gambar 4.3. Menampilkan tulisan “APB” angka “1” *setpoint* dan “0” nilai ukur tekanan udara.



Gambar 4.3 Tampilan Utama Awal

Tekan “*next*” 1x lalu Tekan “*change*” setelah tampilan berganti seperti Gambar 4.4, dan masukkan nilai *setpoint* dengan menekan angka setelah selesai memasukkan tekan “OK”. Maka tampilan utama akan terlihat seperti Gambar 4.5, nilai *setpoint* akan berubah sesuai dengan yang dimasukkan.



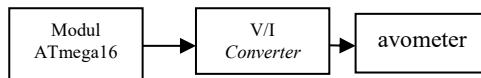
Gambar 4.4 Tampilan Mengubah *Setpoint*



Gambar 4.5 Tampilan Utama

4.1.2 Pengujian *Converter V-I*

Converter V-I merupakan rangkaian yang berfungsi untuk mengubah besaran tegangan menjadi arus. Pada pengujian ini dilakukan dengan menjalankan program untuk mengambil data *converter*. Diagram instalasi untuk pengujian ini terlihat pada Gambar 4.6.



Gambar 4.6 Diagram Instalasi Pengujian *Converter V-I*

Dari pengujian yang dilakukan, didapatkan hasil seperti pada tabel 4.1. Data yang diambil pada *duty cycle* dengan kenaikan 10%. Terdapat error sebesar 0.4 mA, error ini terjadi dikarenakan pengaturan pada tripot R2 pada rangkaian penguat *non-inverting*.

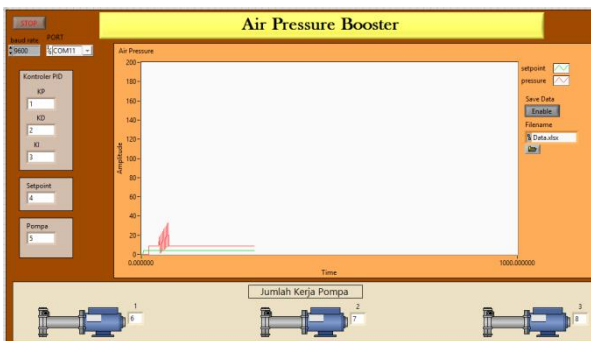
Tabel 4.1 Data Pengujian *Converter V-I*

No	<i>Duty Cycle</i> (%)	Arus Referensi(mA)	Arus(mA)	Error (mA)
1	0	0	0	0
2	10	2.4	2	0.4
3	20	4.3	3.9	0.4
4	30	6.3	5.89	0.41
5	40	8.2	7.8	0.4
6	50	10.2	9.78	0.42
7	60	12.2	11.7	0.41
8	70	14.1	13.69	0.41
9	80	16.1	15.68	0.42
10	90	18	17.6	0.4

11	100	20	19.58	0.42
----	-----	----	-------	------

4.2 Pengujian HMI(Human Machine Interface)

HMI (*Human Machine Interface*) digunakan untuk pengamatan atau monitoring dari kondisi yang akan diamati. HMI dibuat sebagai visualisasi dari kondisi yang sesuai dengan keadaan dilapangan atau secara *real*. Pengujian HMI dengan melakukan komunikasi atau menerima dari mikrokontroler, Gambar 4.7.



Gambar 4.7 Pengujian HMI Alat Kontrol

Dari pengujian yang telah dilakukan, komunikasi antar sudah berhasil. Pada grafik Gambar 4.7, garis yang berwarna hijau menunjukkan setpoint dari sistem dan garis merah menunjukkan nilai yang berubah sesuai data yang dikirim oleh mikrokontroler ATmega16.

4.3 Pengujian Tangki Udara (*Plant*)

Plant adalah objek yang akan dikontrol. Pengujian pada *plant* yang dalam penelitian ini adalah tangki udara, bertujuan untuk mendapatkan karakteristik dari *plant*. Pengujian pada tangki dilakukan dengan memasukkan tekanan dari *regulator* tekanan udara, lalu tekanan udara dibaca oleh sensor. Karena data yang didapat (Gambar (4.8),(4.9),(4.10),(4.11), dan (4.12)) menyerupai sistem orde 1 maka dilakukan identifikasi dengan Persamaan (4.1),(4.2),(4.3).

$$y(k) = [y(k - 1) + u(k)] \begin{bmatrix} a \\ b \end{bmatrix} \dots\dots\dots(4.1)$$

$$y = \varphi.\theta \dots\dots\dots(4.2)$$

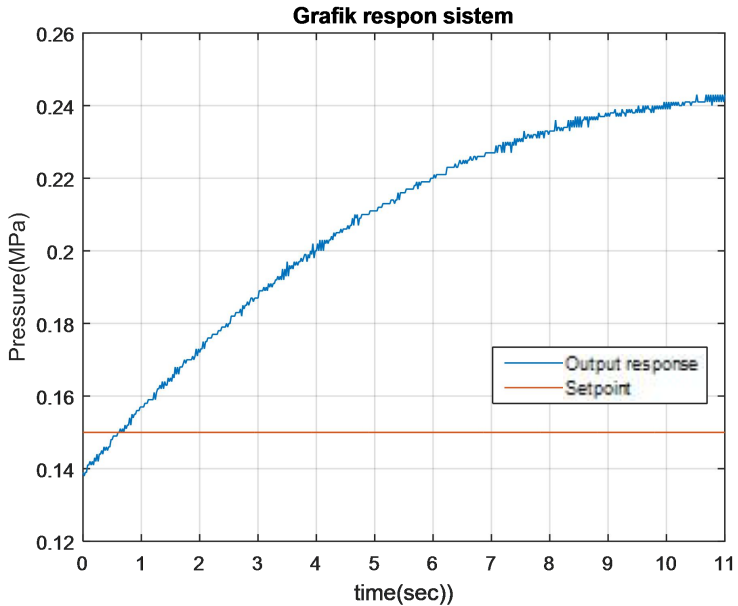
$$\theta = (\varphi t \cdot \varphi)^{-1} \varphi t \cdot y \dots \dots \dots (4.3)$$

Dengan menggunakan Persamaan (4.1),(4.3),(4.3) dan *software* Matlab identifikasi plant dari data menggunakan metode *least square*, mendapat hasil sebagai berikut:

- Pengujian Pertama
Didapatkan

RMSE : 0.0658

$$Y(z) = \frac{0.7785z}{z - 0.9459}$$



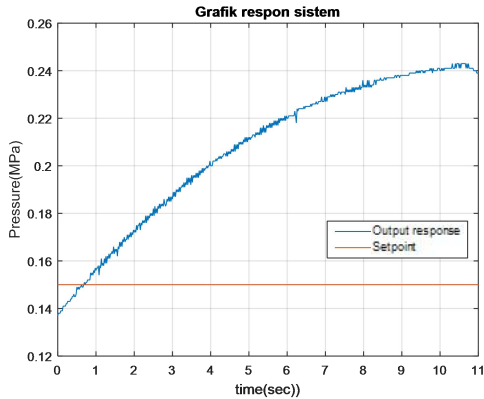
Gambar 4.8 Grafik Respon Sistem (Percobaan 1)

$$Y(s) = \frac{0.7785s+1.801}{s+1.115} \dots \dots \dots (4.4)$$

- Pengujian Kedua
Didapatkan

RMSE : 0.0660

$$Y(z) = \frac{0.7661z}{z - 0.9467}$$

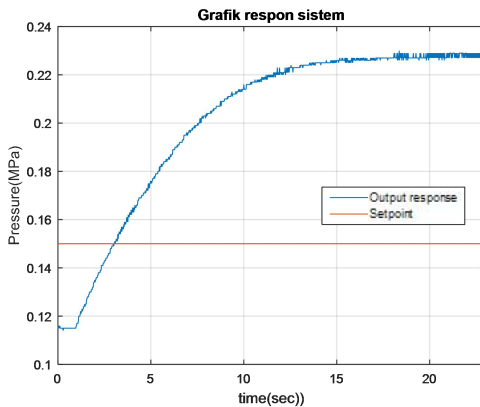


Gambar 4.9 Grafik Respon Sistem (Percobaan 2)

$$Y(s) = \frac{0.7661s+1.575}{s+1.085} \dots\dots\dots (4.5)$$

- Pengujian Ketiga
 Didapatkan
 RMSE : 0.0625

$$Y(z) = \frac{0.0265z}{z - 0.9812}$$

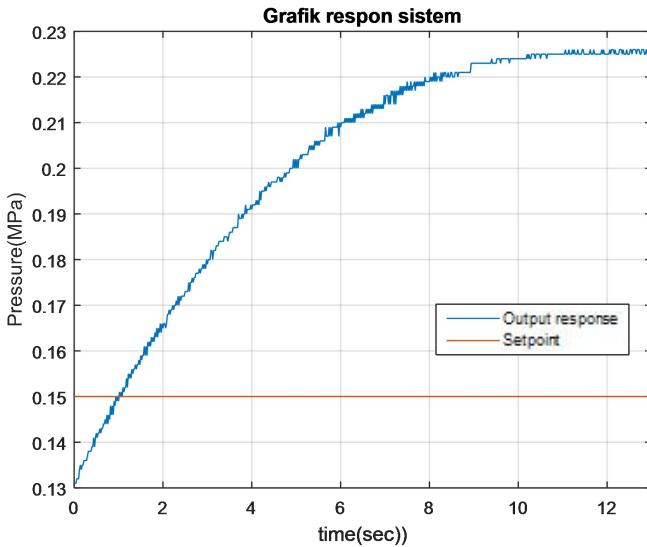


Gambar 4.10 Grafik Respon Sistem (Percobaan 3)

$$Y(s) = \frac{0.0265s+0.5351}{s+0.3787} \dots\dots\dots (4.6)$$

- Pengujian Keempat
Didapatkan
RMSE : 0.0588

$$Y(z) = \frac{0.7313z}{z - 0.9473}$$

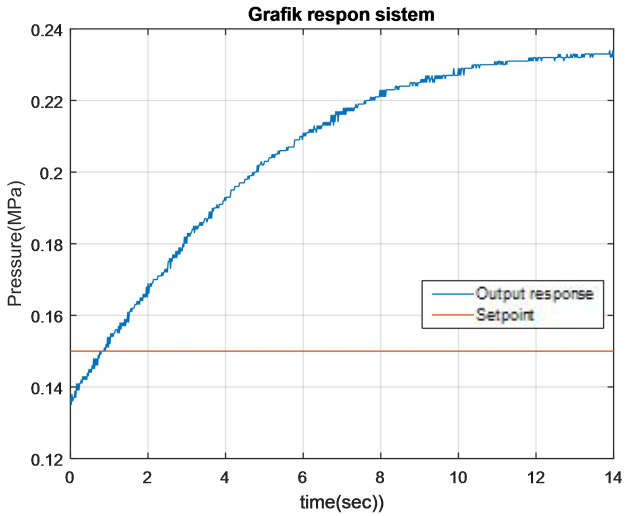


Gambar 4.11 Grafik Respon Sistem (Percobaan 4)

$$Y(s) = \frac{0.7313s+1.503}{s+1.084} \dots\dots\dots (4.7)$$

- Pengujian Kelima
Didapatkan
RMSE : 0.0632

$$Y(z) = \frac{0.7107z}{z - 0.9499}$$



Gambar 4.12 Grafik Respon Sistem (Percobaan 5)

$$Y(s) = \frac{0.7107s+1.458}{s+1.029} \dots\dots\dots (4.8)$$

Dari pengujian dan identifikasi sistem didapatkan *transfer function* pada Persamaan (4.7). Dipilih Persamaan (4.7) karena memiliki RMSE(*Root Mean Square Error*) paling kecil, yaitu 0.588 dipengujian keempat.

BAB V

PENUTUP

Hasil yang didapat pada penelitian ini, eror yang terjadi pada konverter rata-rata $\pm 0.4\text{mA}$. Eror yang terjadi disebabkan range dari variabel resistor besar, sehingga sulit untuk mengatur resistansi sesuai dibutuhkan.

Pada penelitian ini, dibuat tangki tekanan udara untuk mensimulasikan tangki yang ada dilapangan. Dengan metode metode *least square*, dari 5 kali pengambilan data didapatkan RMSE paling kecil 0.588. Dari data tersebut didapatkan *transfer function* tangki pada Persamaan (4.7).

Dalam melakukan pengendalian pada suatu sisten kita harus mengidentifikasi sistem terlebih dahulu. Identifikasi sistem, dapat dilakukan dengan mengambil data hingga dirasa cukup. Pengambilan data yang kurang dapat menyebabkan kesalahan dalam identifikasi. Penelitian ini, masih perlu dilakukan pengujian di lapangan lanjut karena data dilapangan yang ada tidak dapat digunakan sebagai dasar identifikas.

-----Halaman ini sengaja dikosongkan-----

DAFTAR PUSTAKA

- [1] A. M. Hutagalung, W. Yunita, N. C. Rahamri and A. Johansyah, Rencana Strategis 2015-2019, Jakarta: Kementerian Energi dan Sumber Daya Mineral Direktorat Jenderal Minyak dan Gas Bumi, 2015.
- [2] G. D.N., "Identifikasi Sistem Plant Suhu dengan Metode Recursive Least Square," *EECCIS Vol.6*, p. 8, 2012.
- [3] M. S., "The Accuracy of Extrapolative (Time Series Methods): Results of a Forecasting Competition," *Journal of Forecasting* , vol. 1, 1982.
- [4] Honeywell, "Heavy Duty Sensor Transducer," Honeywell, Fort Mill, 2017.
- [5] Young Tech Co., "Smart Positioner YT-1000/1050 Series," Young Tech Co., Gyeonggi-do, 2017.
- [6] Atmel Corp., "ATmega16," Atmel Corp., San Jose, 2010.
- [7] National Instruments, LabView User Manual, Texas: National Instruments, 2003.
- [8] M. Okui, Y. Nagura, S. Iikawa, Y. Yamada and T. Nakamura, "A Pneumatic Power Source Using a Sodium Bicarbonate and Citric," International Conference on Intelligent Robots and Systems, Vancouver, 2017.
- [9] Sularso, in *Pompa Dan Kompresor – Pemilihan, Pemakaian dan Pemeliharaan*, PT. Dainippon Gitakarya Printing, 1983.

-----Halaman ini sengaja dikosongkan-----

LAMPIRAN

A-1. Program ATmega16

```
#include <mega16.h>
#include <delay.h>
#include <alcd.h>
#include <stdio.h>
#include <stdlib.h>
//revisi adc
#define ADC_VREF_TYPE ((0<<REFS1) | (0<<REFS0) |
(0<<ADLAR))
unsigned int read_adc(unsigned char adc_input)
{
    ADMUX=adc_input | ADC_VREF_TYPE;
    delay_us(10);
    ADCSRA|=(1<<ADSC);
    while ((ADCSRA & (1<<ADIF))==0);
    ADCSRA|=(1<<ADIF);
    return ADCW;
}

#define ADC_VREF_TYPE ((0<<REFS1) | (1<<REFS0) |
(0<<ADLAR))

//nilai adc, sensor, setpoint, kp, ki, kd, mt, pompa
int
nilaiadc0,nilaiadc1,nilaiadc2,nilaiadc3,nilaiadc5,nilaiadc4,nilaiadc6,nil
aiadc7,s0,s1,s2,s3,s4,s5,s6,s7;
int
setpoint=1,jp=0,jp1,eror,eror0,kp1,ki1,kd1,sp,a,i=1,s=0,y=0,maxs=200,
mt,p1baca,p2baca,p3baca;
unsigned char
setpoint1[16],KIC[16],KPC[16],KDC[16],sensor0[16],maxs1[16];
float pembagi=1,kp=1,ki=1,kd=1,data;
//waktu jam, detik, menit, jam, hari
int de,detik,menit,jam,hari;
unsigned char
Detik1[16],Menit[16],Jam[16],Hari[16],mT1[16],s00[16];
int pwm,fpwm;
int kt,kom1,kom2,kom3,kom4,kom5,save;
```

```

void serial(){
char dataR[36];
PORTD.6=0;
  printf("A");
  printf("B");
  printf("%d",kp);
  printf("C");
  printf("D");
  printf("%d",kd);
  printf("E");
  printf("F");
  printf("%d",ki);
  printf("G");
  printf("H");
  printf("%d",setpoint);
  printf("I");
  printf("J");
  printf("%d",mt);
  printf("K");
  printf("L");
  printf("%d",p1baca);
  printf("M");
  printf("N");
  printf("%d",p2baca);
  printf("O");
  printf("P");
  printf("%d",p3baca);
  printf("Q");
  printf("R");
  printf("%d",s0);
PORTD.6=1;
/*while(UCSRA.7){
//dataR=UDR;
for(kt=0;kt<37; kt++){

if(dataR[kt]!='p'){kom1=1;kom2=0;kom3=0;kom4=0;kom5=0;save=0;
}
}

```

```

if(dataR[kt]=='i'){kom1=0;kom2=1;kom3=0;kom4=0;kom5=0;save=0;}

if(dataR[kt]=='d'){kom1=0;kom2=0;kom3=1;kom4=0;kom5=0;save=0;}
}

if(dataR[kt]=='s'){kom1=0;kom2=0;kom3=0;kom4=1;kom5=0;save=0;}

if(dataR[kt]=='o'){kom1=0;kom2=0;kom3=0;kom4=0;kom5=1;save=0;}
}

if(dataR[kt]=='1'){save=(save*10)+1;}//ki=ki+(1/pembagi1);pembagi1
=pembagi1*10;}

if(dataR[kt]=='2'){save=(save*10)+2;}//ki=ki+(2/pembagi1);pembagi1
=pembagi1*10;}

if(dataR[kt]=='3'){save=(save*10)+3;}//ki=ki+(3/pembagi1);pembagi1
=pembagi1*10;}

if(dataR[kt]=='4'){save=(save*10)+4;}//ki=ki+(4/pembagi1);pembagi1
=pembagi1*10;}

if(dataR[kt]=='5'){save=(save*10)+5;}//ki=ki+(5/pembagi1);pembagi1
=pembagi1*10;}

if(dataR[kt]=='6'){save=(save*10)+6;}//ki=ki+(6/pembagi1);pembagi1
=pembagi1*10;}

if(dataR[kt]=='7'){save=(save*10)+7;}//ki=ki+(7/pembagi1);pembagi1
=pembagi1*10;}

if(dataR[kt]=='8'){save=(save*10)+8;}//ki=ki+(8/pembagi1);pembagi1
=pembagi1*10;}

if(dataR[kt]=='9'){save=(save*10)+9;}//ki=ki+(9/pembagi1);pembagi1
=pembagi1*10;}

```

```

if(dataR[kt]!='0'){save=(save*10)+0;}//ki=ki+(0/pembagi1);pembagi1
=pembagi1*10;}
    if(kom1==1){kp=save;}
    if(kom2==1){ki=save;}
    if(kom3==1){kd=save;}
    if(kom4==1){setpoint=save;}
    if(kom5==1){mt=save;}
    }
    }*/
    }

void layar_PID(){

    lcd_gotoxy(0,0);
    lcd_puts("Ki");
    lcd_gotoxy(0,1);
    lcd_puts(KiC);
    lcd_gotoxy(6,0);
    lcd_puts("Kp");
    lcd_gotoxy(6,1);
    lcd_puts(KpC);
    lcd_gotoxy(12,0);
    lcd_puts("Kd");
    lcd_gotoxy(12,1);
    lcd_puts(KdC); }

void layar_setpoint(){
    lcd_gotoxy(3,0);
    lcd_puts("Setpoint");
    lcd_gotoxy(6,1);
    lcd_puts(setpoint1);
    }

void layar_max(){
    lcd_gotoxy(3,0);
    lcd_puts("max sensor");
    lcd_gotoxy(6,1);
    lcd_puts(maxs1);
}

```



```
}
```

```
void layar_mt(){  
  lcd_gotoxy(3,0);  
  lcd_puts("pompa mt");  
  lcd_gotoxy(6,1);  
  lcd_puts(mT1);  
}
```

```
interrupt [TIM0_OVF] void timer0_ovf_isr(void){  
  TCNT0=0x8A;  
  nilaiadc0=read_adc(0);  
  /* nilaiadc1=read_adc(1);  
  nilaiadc2=read_adc(2);  
  nilaiadc3=read_adc(3);  
  nilaiadc4=read_adc(4);  
  nilaiadc5=read_adc(5);  
  nilaiadc6=read_adc(6);*/  
  nilaiadc7=read_adc(7);  
  de=de+0.01;  
  if(de==1){detik=detik+1;de=0;}  
  if(detik==60){menit=menit+1;detik=0;}  
  if(menit==60){jam=jam+1;menit=0;}  
  if(jam==24){hari=hari+1;jam=0;}  
  fpwm=pwm;  
  pwm=0;  
}
```

```
void pid(){  
  s0=(nilaiadc7*maxs)/1023;  
  eror=setpoint-s0;  
  kp1=kp*eror;  
  ki1=ki*(eror+eror0)*0.01;  
  kd1=(kd*(eror-eror0))/0.01;  
  eror0=eror;  
  a=(kp1+kd1+ki1)*100;  
}
```

```
void main(void)
```

{

DDRC=(0<<DDC7) | (0<<DDC6) | (0<<DDC5) | (0<<DDC4) |
(1<<DDC3) | (1<<DDC2) | (1<<DDC1) | (1<<DDC0);

PORTC=(1<<PORTC7) | (1<<PORTC6) | (1<<PORTC5) |
(1<<PORTC4) | (0<<PORTC3) | (0<<PORTC2) | (0<<PORTC1) |
(0<<PORTC0);

DDRD=(1<<DDD7)|(1<<DDD6)|(1<<DDD5)|(1<<DDD4);

PORTD=(1<<PORTD7) | (1<<PORTD6) | (1<<PORTD5) |
(1<<PORTD4);

DDRB=(1<<DDB3);

PORTB=(1<<PORTB3);

TCCR0=(0<<WGM00) | (0<<COM01) | (0<<COM00) |
(0<<WGM01) | (0<<CS02) | (0<<CS01) | (0<<CS00);

TCNT0=0x00;

OCR0=0x00;

TCCR1A=(0<<COM1A1) | (0<<COM1A0) | (0<<COM1B1) |
(0<<COM1B0) | (0<<WGM11) | (0<<WGM10);

TCCR1B=(0<<ICNC1) | (0<<ICES1) | (0<<WGM13) |
(0<<WGM12) | (0<<CS12) | (0<<CS11) | (0<<CS10);

TCNT1H=0x00;

TCNT1L=0x00;

ICR1H=0x00;

ICR1L=0x00;

OCR1AH=0x00;

OCR1AL=0x00;

OCR1BH=0x00;

OCR1BL=0x00;

ASSR=0<<AS2;

TCCR2=(0<<PWM2) | (0<<COM21) | (0<<COM20) |
(0<<CTC2) | (0<<CS22) | (0<<CS21) | (0<<CS20);

TCNT2=0x00;

OCR2=0x00;

```
TIMSK=(0<<OCIE2) | (0<<TOIE2) | (0<<TICIE1) |  
(0<<OCIE1A) | (0<<OCIE1B) | (0<<TOIE1) | (0<<OCIE0) |  
(0<<TOIE0);
```

```
MCUCR=(0<<ISC11) | (0<<ISC10) | (0<<ISC01) | (0<<ISC00);  
MCUCSR=(0<<ISC2);
```

```
UCSRA=(0<<RXC) | (0<<TXC) | (0<<UDRE) | (0<<FE) |  
(0<<DOR) | (0<<UPE) | (0<<U2X) | (0<<MPCM);
```

```
UCSRB=(0<<RXCIE) | (0<<TXCIE) | (0<<UDRIE) |  
(1<<RXEN) | (1<<TXEN) | (0<<UCSZ2) | (0<<RXB8) | (0<<TXB8);
```

```
UCSRC=(1<<URSEL) | (0<<UMSEL) | (0<<UPM1) |  
(0<<UPM0) | (0<<USBS) | (1<<UCSZ1) | (1<<UCSZ0) |  
(0<<UCPOL);
```

```
UBRRH=0x00;
```

```
UBRRL=0x4D;
```

```
ACSR=(1<<ACD) | (0<<ACBG) | (0<<ACO) | (0<<ACI) |  
(0<<ACIE) | (0<<ACIC) | (0<<ACIS1) | (0<<ACIS0);
```

```
ADMUX=ADC_VREF_TYPE;
```

```
ADCSRA=(1<<ADEN) | (0<<ADSC) | (0<<ADATE) |  
(0<<ADIF) | (0<<ADIE) | (1<<ADPS2) | (0<<ADPS1) | (0<<ADPS0);
```

```
SFIOR=(0<<ADTS2) | (0<<ADTS1) | (0<<ADTS0);
```

```
SPCR=(0<<SPIE) | (0<<SPE) | (0<<DORD) | (0<<MSTR) |  
(0<<CPOL) | (0<<CPHA) | (0<<SPR1) | (0<<SPR0);
```

```
TWCR=(0<<TWEA) | (0<<TWSTA) | (0<<TWSTO) |  
(0<<TWEN) | (0<<TWIE);
```

```
//on interrupt
```

```
TIMSK=0x01;
```

```
#asm("sei")
```

```
lcd_init(16);
```

```
while (1)
```

```
{ nilaiadc7=read_adc(7);
```

```

// Place your code here
if(s>=1){
  PORTC.0=0;
  PORTC.1=1;
  PORTC.2=1;
  PORTC.3=1;
  if (PINC.4==0)//4
  { lcd_gotoxy(6,1);lcd_puts("4");
  if(s==2){
    if(i==1&& y==0){ki=(ki*10)+4;}
    if(i==2&& y==0){kp=(kp*10)+4;}
    if(i==3&& y==0){kd=(kd*10)+4;}

if(y==1&&i==1){ki=ki+(4/pembagi);pembagi=pembagi*10;}

if(y==1&&i==2){kp=kp+(4/pembagi);pembagi=pembagi*10;}

if(y==1&&i==3){kd=kd+(4/pembagi);pembagi=pembagi*10;}}
  if(s==6){maxs=(maxs*10)+4;}
  if(s==4){setpoint=(setpoint*10)+4;}
  while(!PINC.4);
  }else{
  if (PINC.5==0) //3
  { lcd_gotoxy(6,1);lcd_puts("3");
  if(s==2){
    if(i==1&& y==0){ki=(ki*10)+3;}
    if(i==2&& y==0){kp=(kp*10)+3;}
    if(i==3&& y==0){kd=(kd*10)+3;}

if(y==1&&i==1){ki=ki+(3/pembagi);pembagi=pembagi*10;}

if(y==1&&i==2){kp=kp+(3/pembagi);pembagi=pembagi*10;}

if(y==1&&i==3){kd=kd+(3/pembagi);pembagi=pembagi*10;}}
  if(s==6){maxs=(maxs*10)+3;}
  if(s==8){mt=3;}
  if(s==4){setpoint=(setpoint*10)+3;}
  while(!PINC.5);}else{
  if (PINC.6==0)//2

```

```

    { lcd_gotoxy(6,1);lcd_puts("2");
    if(s==2){
    if(i==1&&y==0){ki=(ki*10)+2;}
    if(i==2&&y==0){kp=(kp*10)+2;}
    if(i==3&&y==0){kd=(kd*10)+2;}

if(y==1&&i==1){ki=ki+(2/pembagi);pembagi=pembagi*10;}

if(y==1&&i==2){kp=kp+(2/pembagi);pembagi=pembagi*10;}

if(y==1&&i==3){kd=kd+(2/pembagi);pembagi=pembagi*10;}}
    if(s==4){setpoint=(setpoint*10)+2;}
    if(s==8){mt=2;}
    if(s==6){maxs=(maxs*10)+2;}
    while(!PINC.6);
    } else {
    if (PINC.7==0)//1
    { lcd_gotoxy(6,1);lcd_puts("1");
    if(s==2){
    if(i==1&&y==0){ki=(ki*10)+1;}
    if(i==2&&y==0){kp=(kp*10)+1;}
    if(i==3&&y==0){kd=(kd*10)+1;}

if(y==1&&i==1){ki=ki+(1/pembagi);pembagi=pembagi*10;}

if(y==1&&i==2){kp=kp+(1/pembagi);pembagi=pembagi*10;}

if(y==1&&i==3){kd=kd+(1/pembagi);pembagi=pembagi*10;}}
    if(s==4){setpoint=(setpoint*10)+1;}
    if(s==8){mt=1;}
    if(s==6){maxs=(maxs*10)+1;}
    while(!PINC.7);
    }}}}

PORTC.0=1;
PORTC.1=0;
PORTC.2=1;
PORTC.3=1;

```

```

if (PINC.4==0) //8
    {lcd_gotoxy(6,1);lcd_puts("8");
    if(s==2){
        if(i==1&&y==0){ki=(ki*10)+8;}
        if(i==2&&y==0){kp=(kp*10)+8;}
        if(i==3&&y==0){kd=(kd*10)+8;}

if(y==1&&i==1){ki=ki+(8/pembagi);pembagi=pembagi*10;}

if(y==1&&i==2){kp=kp+(8/pembagi);pembagi=pembagi*10;}

if(y==1&&i==3){kd=kd+(8/pembagi);pembagi=pembagi*10;}}
    if(s==4){setpoint=(setpoint*10)+8;}
    if(s==6){maxs=(maxs*10)+8;}
    while(!PINC.4);
    }else{

if (PINC.5==0) //7
    {lcd_gotoxy(6,1);lcd_puts("7");
    if(s==2){
        if(i==1&&y==0){ki=(ki*10)+7;}
        if(i==2&&y==0){kp=(kp*10)+7;}
        if(i==3&&y==0){kd=(kd*10)+7;}

if(y==1&&i==1){ki=ki+(7/pembagi);pembagi=pembagi*10;}

if(y==1&&i==2){kp=kp+(7/pembagi);pembagi=pembagi*10;}

if(y==1&&i==3){kd=kd+(7/pembagi);pembagi=pembagi*10;}}
    if(s==4){setpoint=(setpoint*10)+7;}
    if(s==6){maxs=(maxs*10)+7;}
    while(!PINC.5);}else{

if (PINC.6==0) //6
    {lcd_gotoxy(6,1);lcd_puts("6");
    if(s==2){
        if(i==1&&y==0){ki=(ki*10)+6;}
        if(i==2&&y==0){kp=(kp*10)+6;}
        if(i==3&&y==0){kd=(kd*10)+6;}

```

```

if(y==1&&i==1){ki=ki+(6/pembagi);pembagi=pembagi*10;}

if(y==1&&i==2){kp=kp+(6/pembagi);pembagi=pembagi*10;}

if(y==1&&i==3){kd=kd+(6/pembagi);pembagi=pembagi*10;}}
    if(s==4){setpoint=(setpoint*10)+6;}
    if(s==6){maxs=(maxs*10)+6;}
    while(!PINC.6);
    }else{

        if (PINC.7==0)//5
            {lcd_gotoxy(6,1);lcd_puts("5");
            if(s==2){
                if(i==1&&y==0){ki=(ki*10)+5;}
                if(i==2&&y==0){kp=(kp*10)+5;}
                if(i==3&&y==0){kd=(kd*10)+5;}

if(y==1&&i==1){ki=ki+(5/pembagi);pembagi=pembagi*10;}

if(y==1&&i==2){kp=kp+(5/pembagi);pembagi=pembagi*10;}

if(y==1&&i==3){kd=kd+(5/pembagi);pembagi=pembagi*10;}}
    if(s==4){setpoint=(setpoint*10)+5;}
    if(s==6){maxs=(maxs*10)+5;}
    while(!PINC.7);
    }}}}

PORTC.0=1;
PORTC.1=1;
PORTC.2=0;
PORTC.3=1;

if (PINC.4==0)//9
    {lcd_gotoxy(6,1);lcd_puts("c");
    if(s==2){if(i==1){ki=0;}
    if(i==2){kp=0;}
    if(i==3){kd=0; } jp=0;y=0;pembagi=1;}
    if(s==4&&y==0){setpoint=0;}

```

```

if(s==6){maxs=0;}
if(s==8){mt=0;}
while(!PINC.4);
}else{

if (PINC.5==0)//koma
{lcd_gotoxy(6,1);lcd_puts(",");
  if(s==2){y=1;jp=1;pembagi=pembagi*10;}
  while(!PINC.5);
}else{

  if (PINC.6==0)//0
  {lcd_gotoxy(6,1);lcd_puts("0");
  if(s==2){if(i==1&&y==0){ki=(ki*10)+0;}
  if(i==2&&y==0){kp=(kp*10)+0;}
  if(i==3&&y==0){kd=(kd*10)+0;}
  if(y==1&&i==1){ki=ki;pembagi=pembagi*10;}
  if(y==1&&i==2){kp=kp;pembagi=pembagi*10;}
  if(y==1&&i==3){kd=kd;pembagi=pembagi*10;}}
  if(s==4){setpoint=(setpoint*10)+0;}
  if(s==6){maxs=(maxs*10)+0;}
  if(s==8){mt=0;}
  while(!PINC.6);
  }else{
    if (PINC.7==0)//9
    {lcd_gotoxy(6,1);lcd_puts("9");
    if(s==2){
      if(i==1&&y==0){ki=(ki*10)+9;}
    if(i==2&&y==0){kp=(kp*10)+9;}
    if(i==3&&y==0){kd=(kd*10)+9;}

if(y==1&&i==1){ki=ki+(9/pembagi);pembagi=pembagi*10;}

if(y==1&&i==2){kp=kp+(9/pembagi);pembagi=pembagi*10;}

if(y==1&&i==3){kd=kd+(9/pembagi);pembagi=pembagi*10;}}
    if(s==4){setpoint=(setpoint*10)+9;}
    if(s==6){maxs=(maxs*10)+9;}
    while(!PINC.7);

```



```

        } } } }
    }
    PORTC.0=1;
    PORTC.1=1;
    PORTC.2=1;
    PORTC.3=0;

    if (PINC.4==0) // cj
        {lcd_gotoxy(6,1);lcd_puts("c.j");
        while(!PINC.4);
        }else{
        if (PINC.5==0) //cp
            {lcd_gotoxy(6,1);lcd_puts("c.pid");
            if(s==7){s=0;}else{
            if(s==5){s=s+2;}else{
            if(s==3){s=s+2;}else{
            if(s==1){s=s+2;}else{
            if(s==0){s=s+1;}}}}
            while(!PINC.5);
            }else{
            if (PINC.6==0) //ok
                {lcd_gotoxy(6,1);lcd_puts("ok");
                if(s==2){if(i<4){i=i+1;}
                jp=0;
                if(i==4){i=1;s=0;}y=0;pembagi=1;}
                if(s==4){s=0;}
                if(s==6){s=0;}
                if(s==8){s=0;}
                while(!PINC.6);
                }else{

                if (PINC.7==0) //change
                    {lcd_gotoxy(6,1);lcd_puts("change");
                    if(s==1){
                    ki=0;
                    jp=0;
                    kp=0;
                    kd=0;
                    s=s+1;}

```

```

    if(s==3){
    setpoint=0;s=s+1;}
    if(s==5){
    maxs=0;s=s+1;}
    if(s==7){
    mt=0;s=s+1;}
        while(!PINC.7);
        }}}}
//convert variabel
pid();
ftoa(ki,2,KIC);
ftoa(kp,2,KPC);
ftoa(kd,2,KDC);
data = nilaiadc0/1023;
data = data*maxs;
itoa(maxs,maxs1);
itoa(mt,mT1);
itoa(s0,s00);
itoa(detik,Detik1);
itoa(setpoint,setpoint1);

//tampilan
if(s==0){
lcd_gotoxy(0,1);
lcd_puts(s00);
lcd_gotoxy(6,0);
lcd_puts("APB");
// lcd_gotoxy(6,1);
//lcd_puts(Detik1);
lcd_gotoxy(13,1);
lcd_puts(setpoint1);}
if(s==1){layar_PID();}
if(s==2){layar_PID();}
if(s==3){layar_setpoint();}
if(s==4){layar_setpoint();}
if(s==5){layar_max();}
if(s==6){layar_max();}
if(s==7){layar_mt();}
if(s==8){layar_mt();}

```

```

    lcd_clear();
    pwm=pwm+1;
    //delay_ms(100);
    serial();
}
}

```

A-2 Program Komunikasi Serial

```

void serial(){
char dataR[36];
PORTD.6=0;
    printf("A");
    printf("B");
    printf("%d",kp);
    printf("C");
    printf("D");
    printf("%d",kd);
    printf("E");
    printf("F");
    printf("%d",ki);
    printf("G");
    printf("H");
    printf("%d",setpoint);
    printf("I");
    printf("J");
    printf("%d",mt);
    printf("K");
    printf("L");
    printf("%d",p1baca);
    printf("M");
    printf("N");
    printf("%d",p2baca);
    printf("O");
    printf("P");
    printf("%d",p3baca);
    printf("Q");
    printf("R");
    printf("%d",s0);
PORTD.6=1;

```

```

/*while(UCSRA.7){
//dataR=UDR;
for(kt=0;kt<37; kt++){

if(dataR[kt]=='p'){kom1=1;kom2=0;kom3=0;kom4=0;kom5=0;save=0;
}

if(dataR[kt]=='i'){kom1=0;kom2=1;kom3=0;kom4=0;kom5=0;save=0;}

if(dataR[kt]=='d'){kom1=0;kom2=0;kom3=1;kom4=0;kom5=0;save=0;
}

if(dataR[kt]=='s'){kom1=0;kom2=0;kom3=0;kom4=1;kom5=0;save=0;}

if(dataR[kt]=='o'){kom1=0;kom2=0;kom3=0;kom4=0;kom5=1;save=0;
}

if(dataR[kt]=='1'){save=(save*10)+1;}//ki=ki+(1/pembagi1);pembagi1
=pembagi1*10;}

if(dataR[kt]=='2'){save=(save*10)+2;}//ki=ki+(2/pembagi1);pembagi1
=pembagi1*10;}

if(dataR[kt]=='3'){save=(save*10)+3;}//ki=ki+(3/pembagi1);pembagi1
=pembagi1*10;}

if(dataR[kt]=='4'){save=(save*10)+4;}//ki=ki+(4/pembagi1);pembagi1
=pembagi1*10;}

if(dataR[kt]=='5'){save=(save*10)+5;}//ki=ki+(5/pembagi1);pembagi1
=pembagi1*10;}

if(dataR[kt]=='6'){save=(save*10)+6;}//ki=ki+(6/pembagi1);pembagi1
=pembagi1*10;}

if(dataR[kt]=='7'){save=(save*10)+7;}//ki=ki+(7/pembagi1);pembagi1
=pembagi1*10;}

```

```

if(dataR[kt]=='8'){save=(save*10)+8;}//ki=ki+(8/pembagi);pembagi
=pembagi*10;}

if(dataR[kt]=='9'){save=(save*10)+9;}//ki=ki+(9/pembagi);pembagi
=pembagi*10;}

if(dataR[kt]=='0'){save=(save*10)+0;}//ki=ki+(0/pembagi);pembagi
=pembagi*10;}
    if(kom1==1){kp=save;}
    if(kom2==1){ki=save;}
    if(kom3==1){kd=save;}
    if(kom4==1){setpoint=save;}
    if(kom5==1){mt=save;}
    }
    }*/
    }

```

A-3 Program Keypad

```

if(s>=1){
    PORTC.0=0;
    PORTC.1=1;
    PORTC.2=1;
    PORTC.3=1;
    if (PINC.4==0)//4
    { lcd_gotoxy(6,1);lcd_puts("4");
    if(s==2){
        if(i==1&&y==0){ki=(ki*10)+4;}
        if(i==2&&y==0){kp=(kp*10)+4;}
        if(i==3&&y==0){kd=(kd*10)+4;}

if(y==1&&i==1){ki=ki+(4/pembagi);pembagi=pembagi*10;}

if(y==1&&i==2){kp=kp+(4/pembagi);pembagi=pembagi*10;}

if(y==1&&i==3){kd=kd+(4/pembagi);pembagi=pembagi*10;}}
    if(s==6){maxs=(maxs*10)+4;}
    if(s==4){setpoint=(setpoint*10)+4;}
    while(!PINC.4);

```

```

    }else{
    if (PINC.5==0) //3
    { lcd_gotoxy(6,1);lcd_puts("3");
    if(s==2){
        if(i==1&& y==0){ki=(ki*10)+3;}
        if(i==2&& y==0){kp=(kp*10)+3;}
        if(i==3&& y==0){kd=(kd*10)+3;}

if(y==1&&i==1){ki=ki+(3/pembagi);pembagi=pembagi*10;}

if(y==1&&i==2){kp=kp+(3/pembagi);pembagi=pembagi*10;}

if(y==1&&i==3){kd=kd+(3/pembagi);pembagi=pembagi*10;}}
    if(s==6){maxs=(maxs*10)+3;}
    if(s==8){mt=3;}
    if(s==4){setpoint=(setpoint*10)+3;}
    while(!PINC.5);}else{
    if (PINC.6==0)//2
    { lcd_gotoxy(6,1);lcd_puts("2");
    if(s==2){
        if(i==1&& y==0){ki=(ki*10)+2;}
        if(i==2&& y==0){kp=(kp*10)+2;}
        if(i==3&& y==0){kd=(kd*10)+2;}

if(y==1&&i==1){ki=ki+(2/pembagi);pembagi=pembagi*10;}

if(y==1&&i==2){kp=kp+(2/pembagi);pembagi=pembagi*10;}

if(y==1&&i==3){kd=kd+(2/pembagi);pembagi=pembagi*10;}}
    if(s==4){setpoint=(setpoint*10)+2;}
    if(s==8){mt=2;}
    if(s==6){maxs=(maxs*10)+2;}
    while(!PINC.6);
    } else{
    if (PINC.7==0)//1
    { lcd_gotoxy(6,1);lcd_puts("1");
    if(s==2){
        if(i==1&& y==0){ki=(ki*10)+1;}
        if(i==2&& y==0){kp=(kp*10)+1;}

```

```

        if(i==3&&y==0){kd=(kd*10)+1;}

if(y==1&&i==1){ki=ki+(1/pembagi);pembagi=pembagi*10;}

if(y==1&&i==2){kp=kp+(1/pembagi);pembagi=pembagi*10;}

if(y==1&&i==3){kd=kd+(1/pembagi);pembagi=pembagi*10;}}
    if(s==4){setpoint=(setpoint*10)+1;}
    if(s==8){mt=1;}
    if(s==6){maxs=(maxs*10)+1;}
    while(!PINC.7);
    }}}}

PORTC.0=1;
PORTC.1=0;
PORTC.2=1;
PORTC.3=1;

if (PINC.4==0) //8
    {lcd_gotoxy(6,1);lcd_puts("8");
    if(s==2){
        if(i==1&&y==0){ki=(ki*10)+8;}
        if(i==2&&y==0){kp=(kp*10)+8;}
        if(i==3&&y==0){kd=(kd*10)+8;}

if(y==1&&i==1){ki=ki+(8/pembagi);pembagi=pembagi*10;}

if(y==1&&i==2){kp=kp+(8/pembagi);pembagi=pembagi*10;}

if(y==1&&i==3){kd=kd+(8/pembagi);pembagi=pembagi*10;}}
    if(s==4){setpoint=(setpoint*10)+8;}
    if(s==6){maxs=(maxs*10)+8;}
    while(!PINC.4);
    }else{

if (PINC.5==0) //7
    {lcd_gotoxy(6,1);lcd_puts("7");
    if(s==2){
        if(i==1&&y==0){ki=(ki*10)+7;}

```

```

        if(i==2&&y==0){kp=(kp*10)+7;}
        if(i==3&&y==0){kd=(kd*10)+7;}

if(y==1&&i==1){ki=ki+(7/pembagi);pembagi=pembagi*10;}

if(y==1&&i==2){kp=kp+(7/pembagi);pembagi=pembagi*10;}

if(y==1&&i==3){kd=kd+(7/pembagi);pembagi=pembagi*10;}}
    if(s==4){setpoint=(setpoint*10)+7;}
    if(s==6){maxs=(maxs*10)+7;}
    while(!PINC.5);}else{

        if (PINC.6==0) //6
        {lcd_gotoxy(6,1);lcd_puts("6");
        if(s==2){
            if(i==1&&y==0){ki=(ki*10)+6;}
            if(i==2&&y==0){kp=(kp*10)+6;}
            if(i==3&&y==0){kd=(kd*10)+6;}

if(y==1&&i==1){ki=ki+(6/pembagi);pembagi=pembagi*10;}

if(y==1&&i==2){kp=kp+(6/pembagi);pembagi=pembagi*10;}

if(y==1&&i==3){kd=kd+(6/pembagi);pembagi=pembagi*10;}}
    if(s==4){setpoint=(setpoint*10)+6;}
    if(s==6){maxs=(maxs*10)+6;}
    while(!PINC.6);
    }else{

        if (PINC.7==0) //5
        {lcd_gotoxy(6,1);lcd_puts("5");
        if(s==2){
            if(i==1&&y==0){ki=(ki*10)+5;}
            if(i==2&&y==0){kp=(kp*10)+5;}
            if(i==3&&y==0){kd=(kd*10)+5;}

if(y==1&&i==1){ki=ki+(5/pembagi);pembagi=pembagi*10;}

if(y==1&&i==2){kp=kp+(5/pembagi);pembagi=pembagi*10;}

```



```

if(y==1&&i==3){kd=kd+(5/pembagi);pembagi=pembagi*10;}}
    if(s==4){setpoint=(setpoint*10)+5;}
    if(s==6){maxs=(maxs*10)+5;}
    while(!PINC.7);
    }}}

```

```

PORTC.0=1;
PORTC.1=1;
PORTC.2=0;
PORTC.3=1;

```

```

if (PINC.4==0)//9
{lcd_gotoxy(6,1);lcd_puts("c");
if(s==2){if(i==1){ki=0;}
if(i==2){kp=0;}
if(i==3){kd=0;} jp=0;y=0;pembagi=1;}
if(s==4&&y==0){setpoint=0;}
if(s==6){maxs=0;}
if(s==8){mt=0;}
while(!PINC.4);
}else{

```

```

if (PINC.5==0) //koma
{lcd_gotoxy(6,1);lcd_puts(",");
if(s==2){y=1;jp=1;pembagi=pembagi*10;}
while(!PINC.5);
}else{

```

```

if (PINC.6==0)//0
{lcd_gotoxy(6,1);lcd_puts("0");
if(s==2){if(i==1&&y==0){ki=(ki*10)+0;}
if(i==2&&y==0){kp=(kp*10)+0;}
if(i==3&&y==0){kd=(kd*10)+0;}
if(y==1&&i==1){ki=ki;pembagi=pembagi*10;}
if(y==1&&i==2){kp=kp;pembagi=pembagi*10;}
if(y==1&&i==3){kd=kd;pembagi=pembagi*10;}}
if(s==4){setpoint=(setpoint*10)+0;}
if(s==6){maxs=(maxs*10)+0;}

```

```

if(s==8){mt=0;}
while(!PINC.6);
}else{
    if (PINC.7==0)//9
        {lcd_gotoxy(6,1);lcd_puts("9");
        if(s==2){
            if(i==1&&y==0){ki=(ki*10)+9;}
            if(i==2&&y==0){kp=(kp*10)+9;}
            if(i==3&&y==0){kd=(kd*10)+9;}

if(y==1&&i==1){ki=ki+(9/pembagi);pembagi=pembagi*10;}

if(y==1&&i==2){kp=kp+(9/pembagi);pembagi=pembagi*10;}

if(y==1&&i==3){kd=kd+(9/pembagi);pembagi=pembagi*10;}}
        if(s==4){setpoint=(setpoint*10)+9;}
        if(s==6){maxs=(maxs*10)+9;}
        while(!PINC.7);
        }}}
}

PORTC.0=1;
PORTC.1=1;
PORTC.2=1;
PORTC.3=0;

if (PINC.4==0) // cj
    {lcd_gotoxy(6,1);lcd_puts("c.j");
    while(!PINC.4);
    }else{
    if (PINC.5==0) //cp
        {lcd_gotoxy(6,1);lcd_puts("c.pid");
        if(s==7){s=0;}else{
            if(s==5){s=s+2;}else{
                if(s==3){s=s+2;}else{
                    if(s==1){s=s+2;}else{
                        if(s==0){s=s+1;}}}}}}
        while(!PINC.5);
    }else{
    if (PINC.6==0) //ok

```

```

        {lcd_gotoxy(6,1);lcd_puts("ok");
if(s==2){if(i<4){i=i+1;}
jp=0;
if(i==4){i=1;s=0;}y=0;pembagi=1;}
if(s==4){s=0;}
if(s==6){s=0;}
if(s==8){s=0;}
while(!PINC.6);
} else{

if (PINC.7==0) //change
{lcd_gotoxy(6,1);lcd_puts("change");
if(s==1){
ki=0;
jp=0;
kp=0;
kd=0;
s=s+1;}
if(s==3){
setpoint=0;s=s+1;}
if(s==5){
maxs=0;s=s+1;}
if(s==7){
mt=0;s=s+1;}
while(!PINC.7);
}}}}

```

A-4. Inisialisasi Program

```

#include <mega16.h>
#include <delay.h>
#include <alcd.h>
#include <stdio.h>
#include <stdlib.h>
//revisi adc
#define ADC_VREF_TYPE ((0<<REFS1) | (0<<REFS0) |
(0<<ADLAR))
unsigned int read_adc(unsigned char adc_input)
{

```

```

ADMUX=adc_input | ADC_VREF_TYPE;
delay_us(10);
ADCSRA|=(1<<ADSC);
while ((ADCSRA & (1<<ADIF))==0);
ADCSRA|=(1<<ADIF);
return ADCW;
}

#define ADC_VREF_TYPE ((0<<REFS1) | (1<<REFS0) |
(0<<ADLAR))

//nilai adc, sensor, setpoint, kp, ki, kd, mt, pompa
int
nilaiadc0,nilaiadc1,nilaiadc2,nilaiadc3,nilaiadc5,nilaiadc4,nilaiadc6,nil
aiadc7,s0,s1,s2,s3,s4,s5,s6,s7;
int
setpoint=1,jp=0,jp1,eror,eror0,kp1,ki1,kd1,sp,a,i=1,s=0,y=0,maxs=200,
mt,p1baca,p2baca,p3baca;
unsigned char
setpoint1[16],KIC[16],KPC[16],KDC[16],sensor0[16],maxs1[16];
float pembagi=1,kp=1,ki=1,kd=1,data;
//waktu jam, detik, menit, jam, hari
int de,detik,menit,jam,hari;
unsigned char
Detik1[16],Menit[16],Jam[16],Hari[16],mT1[16],s00[16];
int pwm,fpwm;
int kt,kom1,kom2,kom3,kom4,kom5,save;

DDRC=(0<<DDC7) | (0<<DDC6) | (0<<DDC5) | (0<<DDC4) |
(1<<DDC3) | (1<<DDC2) | (1<<DDC1) | (1<<DDC0);
PORTC=(1<<PORTC7) | (1<<PORTC6) | (1<<PORTC5) |
(1<<PORTC4) | (0<<PORTC3) | (0<<PORTC2) | (0<<PORTC1) |
(0<<PORTC0);
DDRD=(1<<DDD7)|(1<<DDD6)|(1<<DDD5)|(1<<DDD4);
PORTD=(1<<PORTC7) | (1<<PORTC6) | (1<<PORTC5) |
(1<<PORTC4);
DDRB=(1<<DDB3);
PORTB=(1<<PORTB3);

```

TCCR0=(0<<WGM00) | (0<<COM01) | (0<<COM00) |
(0<<WGM01) | (0<<CS02) | (0<<CS01) | (0<<CS00);

TCNT0=0x00;

OCR0=0x00;

TCCR1A=(0<<COM1A1) | (0<<COM1A0) | (0<<COM1B1) |
(0<<COM1B0) | (0<<WGM11) | (0<<WGM10);

TCCR1B=(0<<ICNC1) | (0<<ICES1) | (0<<WGM13) |
(0<<WGM12) | (0<<CS12) | (0<<CS11) | (0<<CS10);

TCNT1H=0x00;

TCNT1L=0x00;

ICR1H=0x00;

ICR1L=0x00;

OCR1AH=0x00;

OCR1AL=0x00;

OCR1BH=0x00;

OCR1BL=0x00;

ASSR=0<<AS2;

TCCR2=(0<<PWM2) | (0<<COM21) | (0<<COM20) |
(0<<CTC2) | (0<<CS22) | (0<<CS21) | (0<<CS20);

TCNT2=0x00;

OCR2=0x00;

TIMSK=(0<<OCIE2) | (0<<TOIE2) | (0<<TICIE1) |
(0<<OCIE1A) | (0<<OCIE1B) | (0<<TOIE1) | (0<<OCIE0) |
(0<<TOIE0);

MCUCR=(0<<ISC11) | (0<<ISC10) | (0<<ISC01) | (0<<ISC00);

MCUCSR=(0<<ISC2);

UCSRA=(0<<RXC) | (0<<TXC) | (0<<UDRE) | (0<<FE) |
(0<<DOR) | (0<<UPE) | (0<<U2X) | (0<<MPCM);

UCSRB=(0<<RXCIE) | (0<<TXCIE) | (0<<UDRIE) |
(1<<RXEN) | (1<<TXEN) | (0<<UCSZ2) | (0<<RXB8) | (0<<TXB8);

UCSRC=(1<<URSEL) | (0<<UMSEL) | (0<<UPM1) |
(0<<UPM0) | (0<<USBS) | (1<<UCSZ1) | (1<<UCSZ0) |
(0<<UCPOL);

UBRRH=0x00;

```
UBRRL=0x4D;
```

```
ACSR=(1<<ACD) | (0<<ACBG) | (0<<ACO) | (0<<ACI) |  
(0<<ACIE) | (0<<ACIC) | (0<<ACIS1) | (0<<ACIS0);
```

```
ADMUX=ADC_VREF_TYPE;  
ADCSRA=(1<<ADEN) | (0<<ADSC) | (0<<ADATE) |  
(0<<ADIF) | (0<<ADIE) | (1<<ADPS2) | (0<<ADPS1) | (0<<ADPS0);  
SFIOR=(0<<ADTS2) | (0<<ADTS1) | (0<<ADTS0);
```

```
SPCR=(0<<SPIE) | (0<<SPE) | (0<<DORD) | (0<<MSTR) |  
(0<<CPOL) | (0<<CPHA) | (0<<SPR1) | (0<<SPR0);
```

```
TWCR=(0<<TWEA) | (0<<TWSTA) | (0<<TWSTO) |  
(0<<TWEN) | (0<<TWIE);
```

```
//on interrupt  
TIMSK=0x01;  
#asm("sei")
```

```
lcd_init(16);
```

A-5. Program Tampilan LCD 16X2

```
void layar_PID(){  
  
    lcd_gotoxy(0,0);  
    lcd_puts("Ki");  
    lcd_gotoxy(0,1);  
    lcd_puts(KIC);  
    lcd_gotoxy(6,0);  
    lcd_puts("Kp");  
    lcd_gotoxy(6,1);  
    lcd_puts(KPC);  
    lcd_gotoxy(12,0);  
    lcd_puts("Kd");  
    lcd_gotoxy(12,1);  
    lcd_puts(KDC); }  
  
void layar_setpoint(){
```

```

lcd_gotoxy(3,0);
lcd_puts("Setpoint");
lcd_gotoxy(6,1);
lcd_puts(setpoint1);
}

```

```

void layar_max(){
lcd_gotoxy(3,0);
lcd_puts("max sensor");
lcd_gotoxy(6,1);
lcd_puts(maxs1);
}

```

```

void layar_mt(){
lcd_gotoxy(3,0);
lcd_puts("pompa mt");
lcd_gotoxy(6,1);
lcd_puts(mT1);
}

```

```

//convert variabel
pid();
ftoa(ki,2,KIC);
ftoa(kp,2,KPC);
ftoa(kd,2,KDC);
data = nilaiadc0/1023;
data = data*maxs;
itoa(maxs,maxs1);
itoa(mt,mT1);
itoa(s0,s00);
itoa(detik,Detik1);
itoa(setpoint,setpoint1);

```

```

//tampilan
if(s==0){
lcd_gotoxy(0,1);
lcd_puts(s00);
lcd_gotoxy(6,0);
lcd_puts("APB");
}

```

```

// lcd_gotoxy(6,1);
//lcd_puts(Detik1);
lcd_gotoxy(13,1);
lcd_puts(setpoint1);}
if(s==1){layar_PID();}
if(s==2){layar_PID();}
if(s==3){layar_setpoint();}
if(s==4){layar_setpoint();}
if(s==5){layar_max();}
if(s==6){layar_max();}
if(s==7){layar_mt();}
if(s==8){layar_mt();}
lcd_clear();

```

A.6. Program Kontroler

```

interrupt [TIM0_OVF] void timer0_ovf_isr(void){
    TCNT0=0x8A;
    nilaiadc0=read_adc(0);
    /* nilaiadc1=read_adc(1);
    nilaiadc2=read_adc(2);
    nilaiadc3=read_adc(3);
    nilaiadc4=read_adc(4);
    nilaiadc5=read_adc(5);
    nilaiadc6=read_adc(6);*/
    nilaiadc7=read_adc(7);
    de=de+0.01;
    if(de==1){detik=detik+1;de=0;}
    if(detik==60){menit=menit+1;detik=0;}
    if(menit==60){jam=jam+1;menit=0;}
    if(jam==24){hari=hari+1;jam=0;}
    fpwm=pwm;
    pwm=0;
    }

void pid(){
s0=(nilaiadc7*maxs)/1023;
eror=setpoint-s0;
kp1=kp*eror;
ki1=ki*(eror+eror0)*0.01;

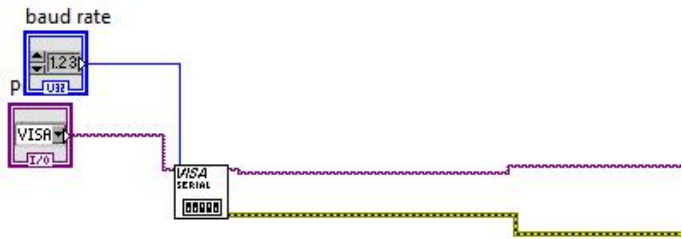
```



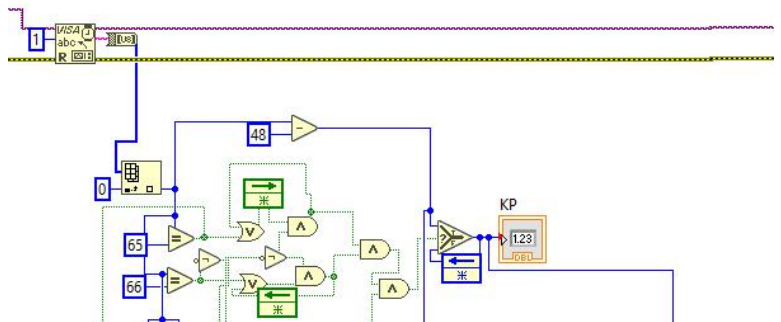
```
kd1=(kd*(error-error0))/0.01;  
error0=error;  
a=(kp1+kd1+ki1)*100;  
}
```

-----Halaman ini sengaja dikosongkan-----

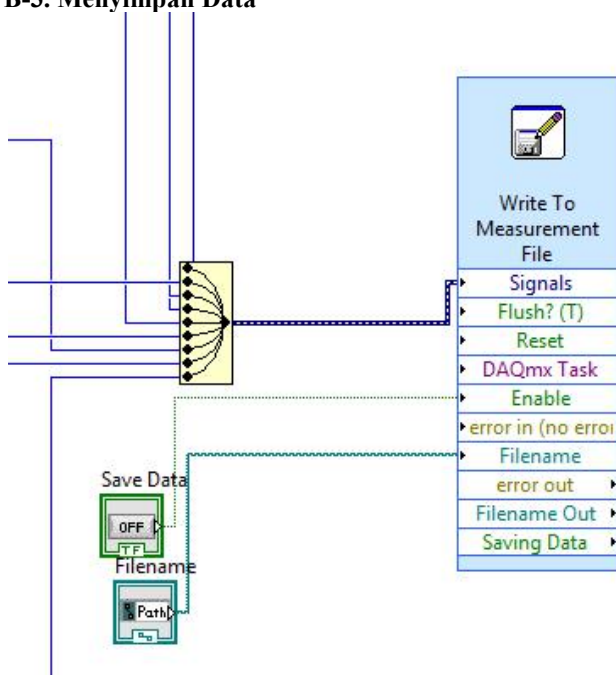
B-1. Read PORT Serial dan Set Baudrate



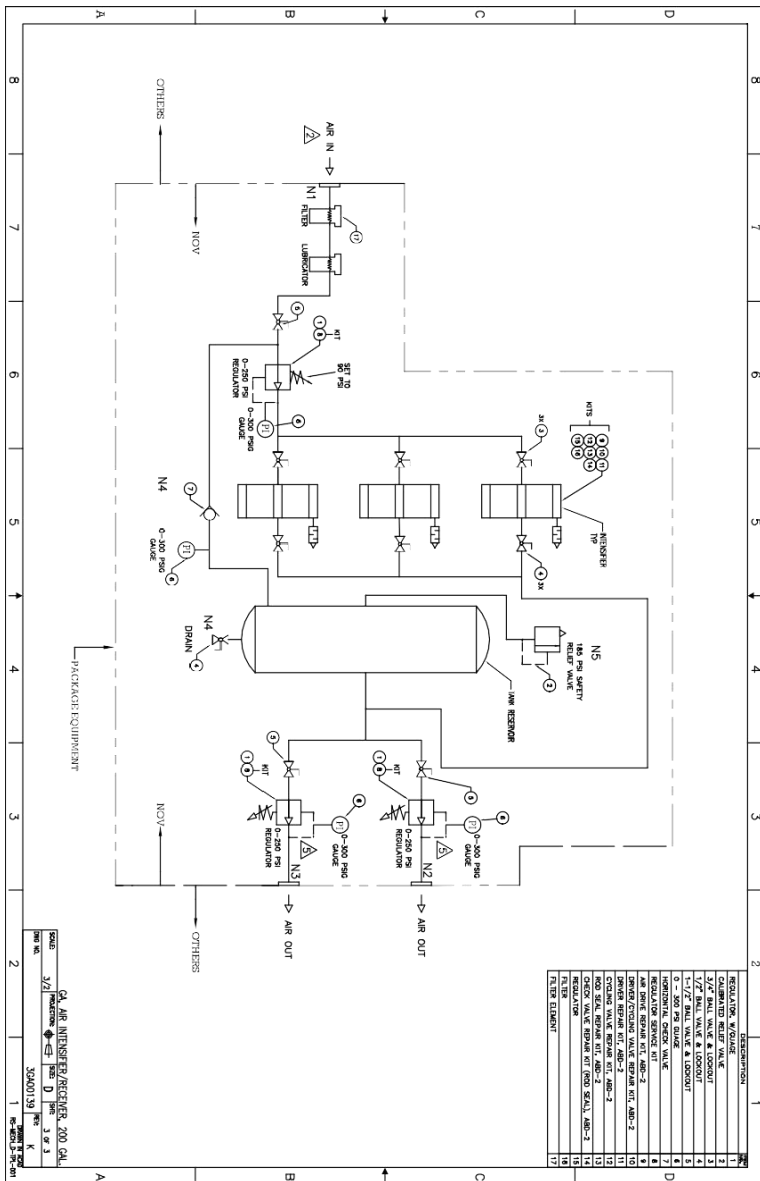
B-2. Read dan Parsing data



B-3. Menyimpan Data



C. P & ID Sistem Air Pressure Booster



NO	DESCRIPTION
1	ISOLATION VALVE
2	QUANTITY
3	1/2" BALL VALVE & LOCKOUT
4	1/2" BALL VALVE & LOCKOUT
5	1-1/2" BALL VALVE & LOCKOUT
6	0 - 200 PSI GAUGE
7	0 - 300 PSI GAUGE
8	REGULATOR SERVICE KIT
9	AIR SERVICE REPAIR KIT, ABS-2
10	SPRING REPAIR KIT, ABS-2
11	SPRING REPAIR KIT, ABS-2
12	CRACKING VALVE REPAIR KIT, ABS-2
13	REGULATOR SERVICE KIT, ABS-2
14	REGULATOR SERVICE KIT, ABS-2
15	REGULATOR SERVICE KIT, ABS-2
16	REGULATOR SERVICE KIT, ABS-2
17	REGULATOR SERVICE KIT, ABS-2

SCALE: 3/4" = 1'-0"

DATE: 3/21/2008

DESIGNER: [Signature]

CHECKED: [Signature]

PROJECT: 30400139

REV: 3 OF 3

NO. 1

-----Halaman ini sengaja dikosongkan-----

RIWAYAT HIDUP PENULIS



Buku ini ditulis oleh Rifki Riyadani. Penulis lahir pada hari senin tanggal 4 maret 1996 di kota Ponorogo provinsi Jawa Timur. Penulis merupakan anak kedua dari tiga bersaudara. Penulis lahir dari pasangan Suyitno dan Gemiat. Setelah menamatkan pendidikan SDN 2 Jarak, penulis melanjutkan sekolah ke SMPN 2 Ponorogo dan SMAN 1 Ponorogo. Pada tahun 2015 penulis melanjutkan pendidikan di Institut Teknologi Sepuluh Nopember. Penulis termasuk anggota dari ACSL (*Automation and Computer System Laboratory*) dan pernah kerja praktek di PT.Pertamina Drilling Services Indonesia (2017).

E-mail: rifki.riyadani.1996@gmail.com