PROYEK AKHIR – VE 180626

RANCANG BANGUN HMI *TOUCHSCREEN* UNTUK MODUL PNEUMATIK *SINGLE ACTING* SILINDER

Bayu Atma Wirandana
NRP 10311500000102

Dosen Pembimbing
Ir. Josaphat Pramudijanto, M.Eng.

Departemen Teknik Elektro Otomasi
Fakultas Vokasi
Institut Teknologi Sepuluh Nopember
Surabaya 2019

FINAL PROJECT – **VE 180626**

BUILDING DESIGN OF TOUCHSCREEN HMI FOR SINGLE ACTING CYLINDER PNEUMATIC MODULE

Bayu Atma Wirandana
NRP 10311500000102

Supervisor
Ir. Josaphat Pramudijanto, M.Eng.

Electrical and Automation Engineering Department
Vocational Faculty
Sepuluh Nopember Institute of Technology
Surabaya 2019

PERNYATAAN KEASLIAN
PROYEK AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Proyek Akhir saya dengan judul **"Rancang Bangun HMI *Touchscreen* Untuk Modul Pneumatik *single acting* Silinder"** 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, Januari 2019

Bayu Atma Wirandana
NRP 10311500000102

RANCANG BANGUN HMI *TOUCHSCREEN* UNTUK MODUL PNEUMATIK *SINGLE ACTING* SILINDER

PROYEK AKHIR

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

Menyetujui :

Dosen Pembimbing

Ir. Josaphat Pramudijanto M.Eng
NIP. 19621005 199003 1 003

SURABAYA
JANUARI, 2019

-----Halaman ini sengaja dikos*ong*kan-----

RANCANG BANGUN HMI *TOUCHSCREEN* UNTUK MODUL PNEUMATIK *SINGLE ACTING* SILINDER

Nama Mahasiswa	: Bayu Atma Wirandana
NRP	: 10311500000102
Dosen Pembimbing	: Ir. Josaphat Pramudijanto, M.Eng
NIP	: 19621005 199003 1 003

ABSTRAK

Industri yang terus berkembang saat ini di pengaruhi berbagai faktor. Salah satu faktor paling berpengaruh ialah kemajuan teknologi. Kemajuan ini di tandai dengan peng*on*trolan sistem kerja peralatan industri secara otomatis. Peralatan yang banyak digunakan saat ini berbasis kontroler dan pneumatik. Oleh karena itu banyak pengguna peralatan otomasi tersebut menjadi lebih mudah dan cepat dikendalikan.

Modul pneumatik yang ada di laboratorium PLC selama ini mengalami kerusakan. Karena kerusakan itulah proyek akhir ini dibuat. Modul pneumatik *single acting cyllinder* akan dikontrol menggunakan arduino dan di*monitoring* dengan HMI *touchscreen* untuk memudahkan proses pembelajaran.

Hasil yang diperoleh dari pembuatan dan pengujian HMI ini adalah HMI dapat menampilkan k*on*disi yang sesuai dengan modul pneumatik secara *real time* dengan waktu tempuh silinder 0,33 detik saat maju dan 0,49 detik saat mundur.

Kata kunci : HMI, *Arduino*, Pneumatik

-----Halaman ini sengaja dikos*ong*kan-----

BUILDING DESIGN OF TOUCHSCREEN HMI FOR SINGLE ACTING CYLINDER PNEUMATIK MODULE

Student Name	**: Bayu Atma Wirandana**
Registration Number	**: 10311500000102**
Supervisor	**: Ir. Josaphat Pramudijanto, M.Eng**
ID	**: 19621005 199003 1 003**

ABSTRACT

The industry that continues to grow is currently influenced by various factors. One of the most influential factors is technological progress. This progress is marked by automatically controlling the industrial work system. The equipment that is widely used today is Arduino and Pneumatik based. Therefore many users of the automation equipment become more easily and quickly controlled.

Pneumatik modules in the PLC laboratory have been damaged. Because of that damage this final project was made. The cylinder single acting pneumatik module will be controlled using ARDUINO and monitored with a touchscreen HMI to facilitate the learning process.

The results obtained from the making and testing of this HMI are that HMI can display conditions that are in accordance with pneumatic modules in real time with cylinder travel times of 0.33 seconds when forward and 0.49 seconds when backward.

Keywords : HMI, Arduino, Pneumatic

-----Halaman ini sengaja dikos*ong*kan-----

KATA PENGANTAR

Puji syukur penulis panjatkan kehadirat Allah SWT yang selalu memberikan rahmat dan hidayah-Nya sehingga Proyek 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.

Proyek Akhir ini disusun untuk memenuhi sebagian persyaratan guna menyelesaikan pendidikan Diploma-3 pada Program Studi Elektr*on*ika Industri, Departemen Teknik Elektro Otomasi, Fakultas Vokasi, Institut Teknologi Sepuluh Nopember Surabaya.

Pada kesempatan ini penulis menyampaikan terima kasih atas segala doa, bantuan dan dukungannya yang telah diberikan selama proses pembuatan Proyek Akhir ini kepada :

1. Kedua orang tua atas dukungan baik spiritual maupun material yang tak ternilai harganya.
2. Bapak Ir. Joko Susila, MT. Ketua departemen Teknik Elektro Otomasi, Fakultas Vokasi- ITS Surabaya.
3. Bapak Ir. Josaphat Pramudijanto, M Eng. Sebagai dosen pembimbing yang telah banyak memberikan ilmu, pengarahan dan bimbingan selama penulis mengerjakan Proyek Akhir ini, serta membimbing saya dengan kesabaran yang tiada batasnya.
4. Semua pihak yang telah membantu saya dalam menyelesaikan Proyek Akhir ini yang tidak dapat kami sebutkan satu persatu.

Penulis menyadari dan memoh*on* maaf atas segala kekurangan pada Proyek Akhir ini. Akhir kata, semoga Proyek Akhir ini dapat bermanfaat dalam pengembangan keilmuan di kemudian hari.

Surabaya, Januari 2019

Bayu Atma Wirandana

-----Halaman ini sengaja dikos*ong*kan-----

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

-----Halaman ini sengaja dikos*ong*kan-----

BAB I
PENDAHULUAN

1.1 Latar Belakang

Industri yang terus berkembang saat ini di pengaruhi berbagai faktor. Salah satu faktor paling berpengaruh ialah kemajuan teknologi. Kemajuan ini di tandai dengan pengontrolan sistem kerja peralatan industri secara otomatis. Peralatan yang banyak digunakan saat ini berbasis arduino dan pneumatik. Oleh karena itu banyak pengguna peralatan otomasi tersebut menjadi lebuh mudah dan cepat dikendalikan.

Modul pneumatik yang ada di laboratorium PLC selama ini mengalami kerusakan. Karena kerusakan itulah proyek akhir ini dibuat. Modul pneumatik *single acting* cyllinder akan dikontrol menggunakan arduino dan di*monitoring* dengan HMI *touchscreen* untuk memudahkan proses pembelajaran.

Diharapkan dengan di buatnya alat ini akan di hasilkan suatu sistem pengoperasian modul pembelajaran pneumatik yang lebih modern dan efisien serta dapat meningkatkan pemahaman pada mahasiswa yang menggunakan modul pembelajaran tersebut.

1.2 Permasalahan

Beberapa permasalahan yang ada dari judul yang di ambil adalah sebagai berikut:

- Pengendalian *trainer* pneumatik masih manual
- Modul pneumatik rusak dan butuh perbaikan.

1.3 Tujuan

Tujuan kami menuliskan proyek akhir ini adalah:

1. Membuat perangkat *monitoring* dan kontrol otomatis pada modul praktikum pneumatik *single acting cyllinder*
2. Mengimplementasikan modul training pneumatik yang di sertai HMI *touchscreen* dan kontroler arduino.

1.4 Sistematika Laporan

Pembahasan proyek akhir ini akan dibagi menjadi lima Bab dengan sistematika sebagai berikut:

Bab I Pendahuluan

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

Bab II Teori Dasar

Bab ini menjelaskan tentang tinjauan pustaka, k*on*sep dari pendistribusian air secara umum, Arduino Uno, Solenoid Valve, GLCD Toucscreen, dan Silinder Pneumati.

Bab III Perancangan Sistem

Bab ini membahas perencanaan dan pembuatan perangkat keras *(Hardware)* secara keseluruhan, dan pembuatan perangkat lunak *(Software)* yang berkaitan dengan perangkat keras yang di rancang

Bab IV Pengujian dan Analisa Sistem

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 proyek akhir dan saran – saran untuk pengembangan alat ini lebih lanjut.

1.5 Relevansi

Dengan adanya proyek akhir ini diharapkan dapat dijadikan suatu alat pembelajaran modul pneumatik dengan aplikasi HMI *Touchscreen* sebagai *monitoring* kerja sistem pneumatik beserta pengaturan secara otomatis dan dengan penerapan kontroler Arduino. Serta membantu memberikan pemahaman terhadap mahasiswa dalam proses pembelajaran mata kuliah teknik pneumatik dan hidrolik itu sendiri..

.

BAB II
TEORI DASAR

Pada bab ini akan dibahas teori dasar dan teori penunjang terkait perangkat dan bahan yang digunakan dalam Proyek akhir ini.

2.1 Pneumatik [1]

Pneumatik merupakan teori atau pengetahuan tentang udara yang bergerak, Keadaan-keadaan keseimbangan udara dan syarat-syarat keseimbangan. Perkataan pneumatik berasal dari bahasa yunani yang berarti napas dan udara. Jadi pneumatik berarti terisi udara atau digerakkan oleh udara. Pneumatik merupakan cabang teori aliran atau Mekanika fluida dan tidak hanya meliputi penelitian aliran-aliran udara melalui sistem saluran,yang terdiri atas selang,gawai,dan sebagainya, tetapi juga aksi dan penggunanan udara mampat.

Pneumatik menggunakan hukum-hukum aeromekanika,yang menentukan keadaan keseimbangan gas dan uap dengan adanya gaya-gaya luar dan teori aliran. Pneumatik dalam pelaksanaan teknik merupakan ilmu pengetahuan dari semua proses mekanik dimana udara memindahkan suata gaya atau gerakan. Jadi pneumatik meliputi semua komponen mesin atau peralatan, dimana terjadi proses-proses pneumatik.

Penggunaan udara bertekanan sebenarnya masih dapat dikembangkan untuk berbagai keperluan proses produksi, misalnya untuk melakukan gerakan mekanik yang selama ini dilakukan oleh tenaga manusia, seperti menggeser, mendor*on*g, mengangkat, menekan, dan lain sebagainya. Namun untuk penerapan di industri sendiri di kombinasikan dengan sensor dan aktuator sesuai fungsi dan tujuan produksi seperti halnya pengemasan,penggerak poros,membuka dan menutup katup, dan mengisi distribusi material.

2.1.1. FRL *(Filter, Regulator* , dan *Lubricator)*

Penggunaan dari peralatan unit pemelihara udara sangat perlu,karena udara bertekanan untuk pneumatik harus dapat memadai dan memiliki kualitas yang baik. Bila kualitas udara baik maka yang terjadi adalah minimalnya kerusakan yang tibul dalam sistem pneumatik. Seperti yang dapat dilihat pada Gambar 2.1 FRL merupakan satu kesatuan.

Gambar 2.1 *Filter Regulator Lubricator*

Oleh karena itu beberapa aspek yang perlu diperhatikan guna mendapat kualitas udara yang baik adalah sebagai berikut

-Tata letak system pendistribusian udara yang sesuai.

-Persyaratan pelumasan jika diperlukan.

-Jenis kompresor yang digunakan harus memenuhi kebutuhan

-Tangki penyimpanan harus memadahi.

-Kualitas udara harus memenuhi sistem.

-Temperatur udara dan pengaruh lain pada sistem

Untuk itu FRL dapat dikatan sebagai unit pemelihara udara. Karena *Filter* sebagai penyaring udara bertekanan *regulator* sebagai pengatur tekanan udara,dan *lubricator* sebagai pelumas udara bertekanan.

2.1.2. *Air Power* (*Compressor*)

Udara bertekanan sebagai energi utama dalam sistem pneumatik dihasilkan oleh *compressor*. Jenis dan kapasitas *compressor* yang diperlukan sesuai dengan kapasitas pneumatik atau jumlah kebutuhan udara yang bekerja dalam sistem pneumatik.

Kompresor adalah mesin atau alat mekanik yang berfungsi untuk meningkatkan tekanan atau memampatkan fluida gas atau udara. Kompresor biasanya menggunakan motor listrik, mesin diesel atau mesin bensin sebagai tenaga penggeraknya. Udara bertekanan hasil dari kompresor biasanya diaplikasikan atau digunakan pada pengecatan dengan teknik spray/ air brush, untuk mengisi angin ban, pembersihan, pneumatik, gerinda udara (air gerinder) dan lain sebagainya. Gambar kompresor dapat dilihat pada Gambar 2.2

Gambar 2.2 Kompresor Angin

Spesifikasi kompresor adalah sebagai berikut:

- Tekanan maksimal kompresor ini adalah 7 Bar
- Kapasitas angin adalah 36 liter udara.

2.1.3. Silinder

Silinder pneumatik merupakan aktuator yang memiliki gerakan maju (*extend*) dan mundur (*retract*). Silinder ada dua jenis yaitu silinder penggerak tunggal dan silinder penggerak ganda.

2.1.3.1. Silinder Pengerak Tunggal

Silinder kerja tunggal mempunyai *seal* pist*on* tunggal yang dipasang pada sisi suplai udara bertekanan. Pembuangan udara pada sisi batang pist*on* silinder dikeluarkan ke luar melalui saluran pembuangan. Jika lubang pembuangan tidak diproteksi dengan sebuah penyaring akan memungkinkan masuknya partikel halus dari debu ke dalam silinder yang bisa merusak *seal*.

Apabila lubang pembuangan ini tertutup akan membatasi atau menghentikan udara yang akan dibuang pada saat silinder gerakan keluar dan gerakan akan menjadi tersentak - sentak atau terhenti. *Seal* terbuat dari bahan yang fleksibel yang ditanamkan di dalam pist*on* dari logam atau plastik. Selama bergerak permukaan *seal* bergeser dengan per-mukaan silinder. Gambar silinder penggerak tunggal dapat dilihat pada Gambar 2.3

Gambar 2.3 Silinder Penggerak Tunggal

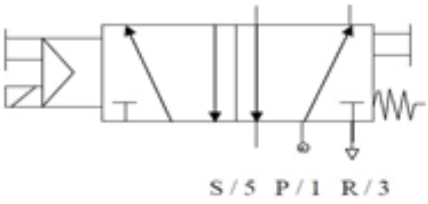
2.1.4. *Solenoid Valve*

Solenoid valve adalah katup yang digerakan oleh energi listrik melalui *solenoid*, mempunyai kumparan sebagai penggeraknya yang berfungsi untuk menggerakan pist*on* yang dapat digerakan oleh arus AC maupun DC, *solenoid valve pneumatik* atau katup *(valve) solenoid* mempunyai lubang keluaran, lubang masukan dan lubang *exhaust*.

2.1.4.1. Silinder Pengerak Tunggal

Solenoid jenis ini mempunyai bagian dalam yang terdiri dari lima saluran dan dua ruangan. Dalam *solenoid* ini terdapat dua *out-put* yang mana ke salah satu *output*-nya bekerja sebelum *solenoid*-nya mendapat tegangan dan arus, serta *system solenoid valve* ini ter-dapat *spring* yang mempunyai fungsi sebagai penarik kembali ba-tang pelat yang ada dalam *valve*-nya untuk menyalurkan tekanan pneumatik pada fungsi *output* yang bekerja pada saat *solenoid* tidak mendapat tegangan dan arus.

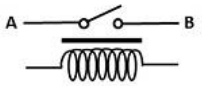
Waktu *solenoid*-nya mendapat tegangan dan arus, induksi yang terjadi dalam *solenoid* tersebut menarik batang pelat yang mempunyai gaya tarik lebih besar dari gaya *spring* dan akibatnya sumber *supply input* menyalurkan *supply*-nya pada *output* yang lainnya, akibatnya *output* yang satunya dapat aktif. Pada beberapa literatur, simbol *single solenoid* diGambarkan seperti Gambar 2.4



Gambar 2.4 Simbol *Single Selenoid*

2.1.5. *Relay*

Relay adalah Saklar (*Switch*) yang dioperasikan secara listrik dan merupakan komponen Electromechanical (Elektromekanikal) yang terdiri dari 2 bagian utama yakni Elektromagnet (Coil) dan Mekanikal (seperangkat K*on*tak Saklar/*Switch*). *Relay* menggunakan Prinsip Elektromagnetik untuk menggerakkan K*on*tak Saklar sehingga dengan arus listrik yang kecil (*low power*) dapat menghantarkan listrik yang bertegangan lebih tinggi. Sebagai c*on*toh, dengan *Relay* yang menggunakan Elektromagnet 5V dan 50 mA (*output* Arduino) mampu menggerakan Armature *Relay* (yang berfungsi sebagai saklarnya) untuk menghantarkan listrik 24V DC 2A (Spesifikasi *Solenoid Valve*). Simbol *Relay* dapat dilihat pada Gambar 2.5



Gambar 2.5 Simbol *Relay*

Prinsip kerja *relay* yaitu apabila kumparan coil diberi arus listrik, maka akan timbul gaya elektromagnetik yang kemudian menarik armature untuk berpindah dari posisi sebelumnya (NC) ke posisi baru (NO). Pada saat tidak dialiri listrik maka armature akan kembali ke posisi awal (NC). Coil yang digunakan oleh *relay* untuk menarik c*on*tact poin pada umumnya hanya membutuhkan arus yang relative kecil.

2.1.5.1 4 *Channel* Modul *Relay* 24 Volt

Modul ini adalah papan antarmuka *Relay* 4 saluran yang memungkinkan untuk meng*on*trol berbagai peralatan dengan arus besar dan dapat dikontrol langsung dengan mikrokontroler seperti Arduino. 4 *channel* modul *relay* 24V dapat dilihat pada Gambar 2.6

Gambar 2.6 4 *Channel* Modul *Relay* 24V

spesifikasi *relay* ini adalah sebagai berikut:

- Papan antarmuka *relay* 4 *channel* ini masing masing membutuhkan *driver* 15-20 mA.
- Dikontrol dengan menggunakan 24 Volt dan 5 Volt
- Dilengkapi dengan *relay* arus tinggi AC250V 10A, DC24V 10A.
- Indikasi LED untuk status *output relay*.

2.2 HMI (*Human Machine Interface)* [2]

Human Machine Interface untuk meningkatkan interaksi antara mesin dan operator melalui tampilan layar komputer dan memenuhi kebutuhan pengguna terhadap informasi sistem. Dengan membuat desain pada HMI yang sesuai dengan sistem kerja dapat memberikan kemudahan untuk dapat me*monitoring* sistem kerja tersebut.

K*on*sep HMI pada industri adalah sebagai media untuk komunikasi antara bagian operator dengan perencanaan serta

perancangan sistem dengan maksimum serta sebagai sarana bagi operator untuk mengakses sistem kerja dilapangan.

Pada umumnya HMI berupa komputer dengan tampilan di-mo*ni*tor dimana pada m*oni*tor tersebut kita bisa melihat keseluruhan sistem kerja dari layar tersebut. Adapun fungsi dari HMI dijabarkan sebagai berikut :

- *Monitoring*, kita dapat melakukan *monitoring* ko*n*disi sistem kerja dilapangan secara *real time* tanpa harus melihat lang-sung kelapangan sehingga bisa menghemat waktu dan tena-ga.

- Dapat mengendalikan sistem kerja dengan seketika melalui m*oni*tor komputer. Sehingga dengan sistem ini dapat menghemat waktu dan tenaga kerja untuk me*monitoring* dan mengendalikan setiap sistem kerja produksi.

2.2.1 5 Inch TFT LCD Arduino For Mega (SSD1963)

TFT LCD (*Thin Film Transistor Liquid Crystal Display*) adalah varian liquid crystal display yang menggunakan transistor Film tipis untuk meningkatkan kualitas Gambar.LCD. LCD jenis ini sudah built in *resistive touchscreen* dan dapat digunakan sebagai *input*. Ukuran dari display LCD ini adalah 5 inch dan dapat dikontrol dengan AVR, ARDUINO, dan ARM. Dapat digunakan dalam sistem tertanam yang memerlukan tampilan Gambar berwarna berkualitas tinggi. Modul ini menggunakan pengontrol LCD SSD1963 dengan Modul LCD 5 inci dengan touchpad. LCD ini memiliki kualitas tampilan superior dan sudut pandang super lebar. Tampilan TFT LCD dapat dilihan pada Gambar 2.10

Gambar 2.10 TFT LCD *Touchscreen* 5 Inch.

- 5 Inch resolusi 800x480
- 5V / 3,3 V *Power Supply*
- *Controller driver* SSD1963
- 8/16 *bit parallel bus* interface PCB

2.3 Sensor [3]

Sensor adalah peralatan yang digunakan untuk merubah suatu besaran fisik menjadi besaran listrik sehingga dapat dianalisa dengan rangkaian listrik tertentu. Pada saat ini, sensor tersebut telah dibuat dengan ukuran sangat kecil. Ukuran yang sangat kecil ini sangat memudahkan pemakaian dan menghemat energi. Sensor merupakan bagian dari transduser yang berfungsi untuk melakukan *sensing* atau "merasakan dan menangkap" adanya perubahan energi eksternal yang akan masuk ke bagian *input* dari transduser, sehingga perubahan kapasitas energi yang ditangkap segera dikirim kepada bagian k*on*vertor dari transduser untuk dirubah menjadi energi listrik.

2.3.1 Limit Switch

Limit switch adalah salah satu sensor yang akan berkerja apabila pada bagian aktuatornya tertekan suatu benda, baik dari samping kiri ataupun kanan, mempunyai micro *switch* dibagian dalamnya yang berfungsi untuk peng*on*tak seperti ditunjukkan pada Gambar 2.7

Gambar 2.7 Simbol *Limit switch*

2.4 Arduino Mega 2560 [3]

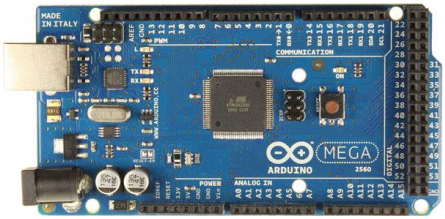
Arduino merupakan papan rangkaian elektr*on*ik atau kit elektr*on*ik yang bersifat *open-source* yang didalamnya terdapat komponen utama yaitu sebuah mikrokontroler dengan jenis *Atmel AVR* dan menggunakan *software* sendiri untuk pemrograman.

Mikrokontroler sendiri adalah sebuah komputer kecil disuatu sikuit terpadu yang berisi tentang inti prosesor, memori dan *input*/*output* yang telah diprogram. Program disimpan dalam bentuk *Ferroelectric* RAM, *Nor Flash*, OTP ROM yang disertakan dalam chip. Mikrokontroler digunakan untuk aplikasi *embedded*, tidak seperti mikroprosesor yang digunakan dalam komputer pribadi. Fungsi dari mikrokontroler adalah untuk mengontrol produk atau perangkat secara otomatis seperti sistem kontrol mesin mobil, mesin kantor, alat-alat listrik, dan sistem *embedded* lainnya.

Salah satu jenis Arduino yang menggunakan mikrokontroler tipe *Atmel AVR* (8-bit) adalah Arduino Mega . Arduino Mega adalah papan mikrokontroler berbasiskan ATmega 2560. Spesifikasinya adalah seperti berikut:

- Memiliki kecepatan *clock* mencapai 16MHz.
- Memiliki *Flash Memory* 256KB, 8kb diantaranya sebagai *bootloader*.
- Rentang *input* sumber tegangan 7-20 Volt, dan beroperasi pada tegangan 5 Volt.
- Memiliki EEPROM (*Electrically Erasable Programmable Read Only Memory*) sebesar 4kb.
- Memiliki SRAM (*Static Random Access Memory*) sebesar 8kb.
- Memiliki pin I/O digital sebanyak 54 pin dan 15 pin diantaranya digunakan sebagai *output* PWM (*Pulse Width Modulation*).
- Memiliki pin I/O analog sebanyak 15 pin.
- Arus DC per pin I/O 40 mA dan pada pada pin 3,3V 50mA.

Letak pin pada *board* Arduino Mega dapat dilihat pada Gambar 2.8.



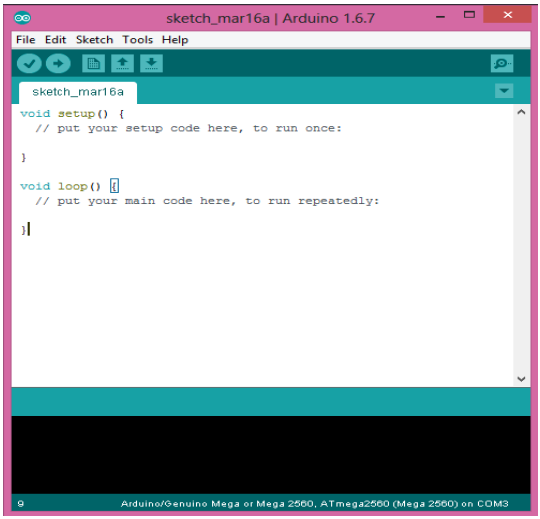
Gambar 2.8. *Board* Arduino Mega 2560

2.4.1 Arduino IDE [3]

IDE itu merupakan kependekan dari *Integrated Developtment Enviroenment*, atau secara bahasa mudahnya merupakan lingkungan terintegrasi yang digunakan untuk melakukan pengembangan. Disebut sebagai lingkungan karena melalui *software* inilah Arduino dilakukan pemrograman untuk melakukan fungsi-fungsi yang dibenamkan melalui sintaks pemrograman. Arduino menggunakan bahasa pemrograman sendiri yang menyerupai bahasa C. Bahasa pemrograman Arduino (*Sketch*) sudah dilakukan perubahan untuk memudahkan pemula dalam melakukan pemrograman dari bahasa aslinya. Sebelum dijual ke pasaran, IC mikrokontroler Arduino telah ditanamkan suatu program bernama *Bootloader* yang berfungsi sebagai penengah antara *compiler* Arduino dengan mikrokontroler. Arduino IDE dibuat dari bahasa pemrograman JAVA. Arduino IDE juga dilengkapi dengan library bahasa pemrograman C/C++ yang biasa disebut *Wiring* yang membuat operasi *input* dan *output* menjadi lebih mudah.ArduinoIDE ini dikembangkan dari *software* Processing yang dirombak menjadi Arduino IDE khusus untuk pemrograman dengan Arduino.

Program yang ditulis dengan menggunaan Arduino *Software* (IDE) disebut sebagai *sketch*. Sketch ditulis dalam suatu editor teks dan disimpan dalam file dengan ekstensi ***.ino***. Teks editor pada menu pemrograman *Software* Arduino memiliki fitur" seperti *cutting/paste* dan *seraching/replacing* sehingga memudahkan dalam menulis kode program.

Pada *Software* Arduino IDE, terdapat semacam *message box* berwarna hitam yang berfungsi menampilkan status, seperti pesan *error*, *compile*, dan *upload* program. Di bagian bawah paling kanan Sotware Arduino IDE, menunjukan board yang terkonfigurasi beserta COM *Port* yang digunakan. Tampilan Program Arduino dapat dilihat pada Gambar 2.9



Gambar 2.9 Tampilan Program Arduino

-  *New* digunakan untuk membuat project baru
-  *Open* berfungsi untuk membuka sketch yang pernah kita buat di arduino dan membuka kembali untuk dilakukan editing atau *upload* ulang ke arduino.
-  *save* berfungsi untuk menyimpan project yang kita buat.
-  *Serial Monitor* Berfungsi untuk menampilkan data apa saja yang didapat ketika arduino bekerja.
-  *Upload* Berfungsi untuk mentransfer Program yang telah kita buat kepada Arduino.
-  *Verify* berfungsi untuk melakukan pengecekan kode yang kita buat dan melakukan pemeriksaan terhadap kode yang *error*

--- Halaman ini sengaja dikosongkan ---

.

BAB III
PERANCANGAN *HARDWARE* DAN *SOFTWARE*

Pada bab ini akan menjelaskan mengenai proses perancangan alat yang berkaitan dengan Proyek akhir "Rancang Bangun HMI *Touchscreen* Untuk Modul Pneumatik *single acting cyllinder*". Pada perancangan alat ini digunakan komponen komponen pneumatik. Komponen pneumatik yang digunakan antara lain ialah 1 buah FRL, 1 buah *cylinder single acting*, 2 buah 5/2 *way valve Single selenoid*, dan selang pneumatik berwarna biru dengan ukuran 6 mm yang menghubungkan komponen – komponen pneumatik ter-sebut. Silinder dihubungkan dengan sebuah *valve* dan 2 buah sensor *Limit switch.*

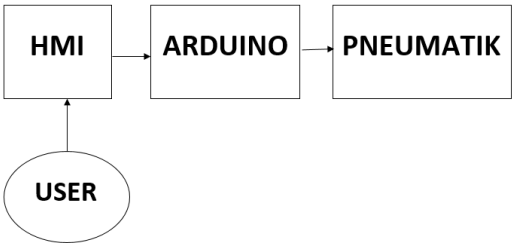
Modul Pneumatik ini akan dikendalikan menggunakan HMI *Touchscreen*. HMI berfungsi sebagai *monitoring* untuk objek pneumatik yang digunakan dan bisa digunakan sebagai masukan Untuk menghubungkan HMI dengan pneumatik HMI dihubungkan dengan arduino menggunakan Shield LCD TFT V2.1. Lalu Arduino disambungkan dengan *relay*. Setelah itu *relay* disambungkan dengan modul pneumatik

Secara garis besar bab ini dibagi menjadi dua bagian yaitu:

- Modul pneumatik,*wiring* modul pneumatik,dan konfigurasi yang menghubungkan arduino dengan pneumatik
- Bagian perancangan dan pembuatan perangkat lunak (*Software*) meliputi perancangan dan pembuatan program arduino dan pembuatan HMI

3.1 Perancangan Perangkat Keras (*Hardware*)

Perancangan sistem secara keseluruhan dalam pembuatan alat Dari Gambar 3.1 dapat dijelaskan bahwa *user* hanya mengendalikan modul pneumatik melalui HMI sebagai *monitoring* dan *controlling*. Arduino mengolah data dan perintah dari *user* dan mengirim perintah untuk ditampilkan pada HMI sehingga memudahkan dalam pengoperaian modul.



Gambar 3.1 Diagram Fungsional

3.1.1 Modul Pneumatik

Tujuan awal modul pneumatik ini sebagai modul pembelajaran.Agar menjadi sebuah modul pembelajaran maka diperlukan komponen-komponen seperti yang tertera pada Table 3.1

Tabel 3.1 Komponen Perlengkapan Alat.

Komponen	**Jumlah**	**Keterangan**
FRL	1 buah	
Solenoid Valve	2 buah	5/3 Single *Solenoid Valve*
Single Acting Cylinder	1 buah	
Limit switch	2 buah	
Selang Angin	5 meter	

Modul pneumatik yang digunakan berupa *box* dengan satu sisi terbuka yang terbuat dari kayu Bagian dalam *box* ini terdapat papan *acrylic* dengan ukuran panjang 90 cm dan lebar 61 cm yang digunakan untuk tempat meletakkan komponen - komponen modul pneumatik yang akan digunakan. Untuk bagian - bagian pengkabelan sensor – sensor ini diletakkan digian belakang dari *acrylic* putih letak dari komponen pneumatik dengan tujuan agar terlihat rapi dan lebih aman dari gangguan luar serta mengurangi resiko kerusakan. Bagian atas dari *box* kayu terdapat PLC yang akan digunakan beserta dengan terminal stopk*on*tak untuk sumber tegangannya.

3.1.2 Modul Pneumatik Sebelum Dilengkapi

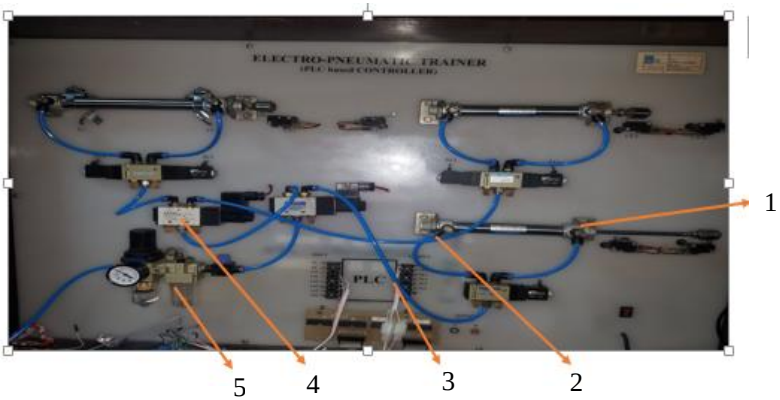
Pada Kondisi modul pneumatik awalnya terdiri dari tiga silinder (2 double acting dan 1 *single acting*) ,tiga *solenoid valve* (2 double *solenoid* dan 1 single *solenoid*) Pada kondisi ini belum bisa digunakan sebagai modul belajar karena belum memiliki *port* yang dapat menghubungkan dengan kontroler arduino. Selain itu banyak komponen pneumatik yang rusak seperti fitting pneumatik dan laiinya. Modul pneumatik sebelum di perbaiki dapat dilihat pada Gambar 3.2

Gambar 3.2 Modul Pneumatik Sebelum Diperbaiki

3.1.3 Modul Pneumatik Setelah Dilengkapi

Modul ini perlu dilengkapi dengan beberapa komponen pneumatik. Dalam modul ini sudah terdapat FRL (*filter, Lubricator , lubricator*) sebagai jalan awal angin sebelum masuk ke modul. FRL juga sebagai penyaring udara yang masuk, mengatur besarnya tekanan yang masuk ke modul dan menampilkan besar tekanannya. Selain itu ada *Valve 5/2 Single Selenoid* yang bertujuan sebagai katup yang mengatur arah angin.

Pada modul ini, ditambahkan *single selenoid valve* dengan *5/2 way valve selenoid*. *5/2 way valve solenoid* yang memiliki arti bahwa *solenoid* tersebut memiliki 5 lubang dengan 2 lubangnya sebagai *output* udara yang mengalir ke *double acting silinder*. *Solenoid* tersebut berfungsi seperti saklar yang berguna untuk mengalirkan udara ke silinder yang akan di aktifkan. Kondisi modul setelah diperbaiki dapat dilihat pada Gambar 3.3



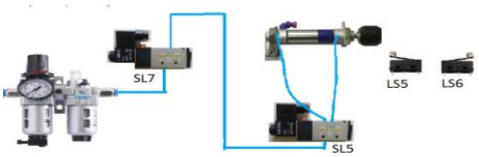
Gambar 3.3 Modul Pneumatik Setelah Diperbaiki

Pada Gambar 3.3 dapat dilihat sebagai berikut

1. Pergantian *fitting* silinder.
2. Pergantian *fitting* silinder.
3. Pembelian selang angin.
4. Pembelian *single solenoid valve.*
5. Pengisian oli pada FRL

3.1.4 *Wiring* Modul Pneumatik

Peran Modul pneumatik memiliki beberapa komponen elektrik se-perti sensor dan *selenoid*. Komponen – komponen elektrik harus dihubungkan dengan sumber tegangan agar dapat bekerja. Sensor akan menjadi masukan (*input*) bagi Arduino. *Wiring* diagram pneumatik dapat dilihat pada Gambar 3.4

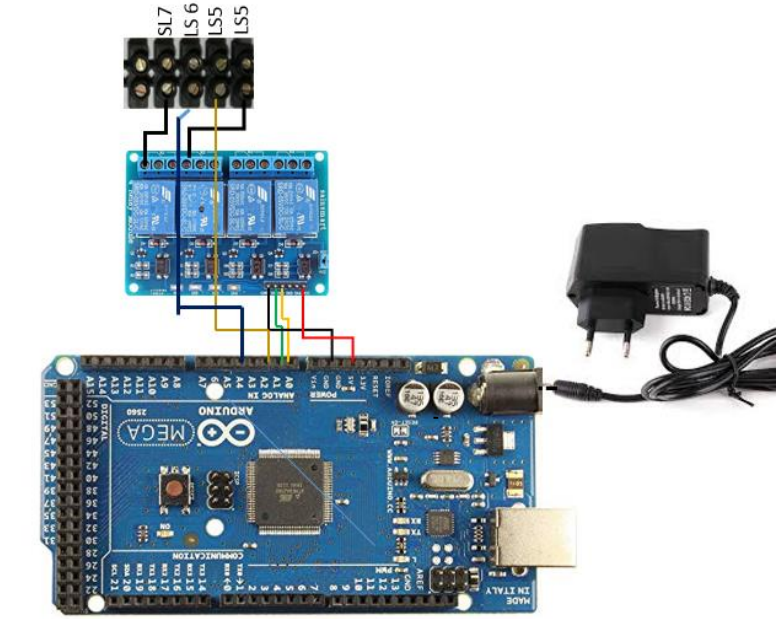


Gambar 3.4 *Wiring* Modul Pneumatik

Wiring diagram pneumatik dapat dilihat sensor *Limit switch* yang dipakai adalah LS5 dan LS6. Pada *wiring* diagram ini pergerakan hanya 1 pergerakan silinder.

3.1.5 Konfigurasi Konektor I/O Arduino Dengan *Wiring* Elektrik Pneumatik [4]

Pada perangkat pneumatik ini kami menggunakan terminal untuk *input output* untuk menghubungkan modul pneumatik dengan arduino. Terminal *input output* dapat dilihat pada Gambar 3.5



Gambar 3.5 Terminal *Input Output* Modul

Pada Gambar dapat dilihat Terminal pin *input output* sudah diberi pengalamatan dan nantinya setiap pin yang dibutuhkan akan dihubungkan dengan arduino dan dapat dilihat pada Tabel 3.2

Tabel 3.2 Pin *Input Output* Arduino

Solenoid 5	A0
Solenoid 7	A1
Limit switch 5	A2
Limit switch 6	A4

3.2 Perancangan Pembuatan Perangkat Lunak

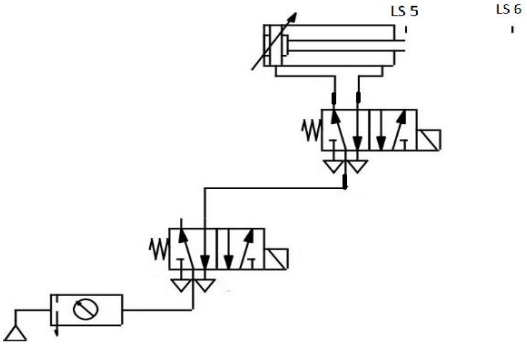
Modul Pneumatik ini dirancang untuk menjalankan empat percobaan yaitu:

1. Karakteristik *single acting cyllinder*.
2. Aplikasi *Counter up* pada Pneumatik.
3. Aplikasi *Timer ON* pada pneumatik.
4. Kombinasi *Timer* dan Pneumatik.

Berikut penjelasan lebih lanjut mengenai pembuatan *Software* dalam pengerjaan Proyek akhir ini.

3.2.1. Perancangan Program Arduino

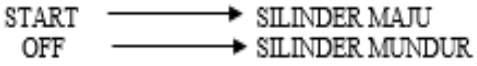
Setelah menetapkan empat percobaan yang akan dilaksanakan maka melalui modul ini kami membuat rancangan gerakan silinder di tiap percobaan. Gerakan silinder diGambarkan melalui diagram langkah. Rangkaian pneumatik dapat dilihat pada Gambar 3.6



Gambar 3.6 Rangkaian Pneumatik

Dari Gambar 3.6 dapat dibuat diagram alur untuk setiap percobaan sebagai berikut:

a.Percobaan 1:

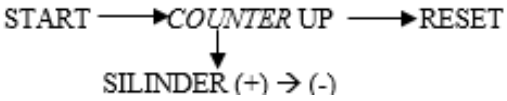


.

Gambar 3.7 Diagram Alur Percobaan 1

Dari Gambar 3.7 dapat dijelaskan bahwa ketika tombol *Start* disentuh maka Silinder akan bergerak maju dan ketika tombol *Off* disentuh maka silinder akan bergerak mundur.

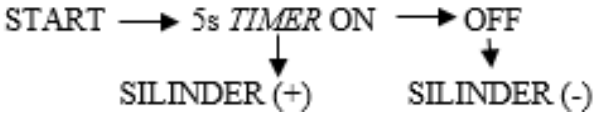
b.Perobaan 2:



Gambar 3.8 Diagram Alur Percobaan 2

Dari Gambar 3.8 dapat dijelaskan bahwa ketika Tombol *counter up* ditekan maka yang silinder akan bergerak maju mundur satu kali dan tercatat jumlahnya dalam tampilan *touchscreen*. Tombol reset akan mengulang jumlah *counter* menjadi 0

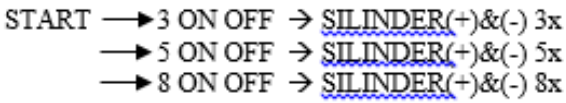
c.Percobaan 3:



Gambar 3.9 Diagram Alur Percobaan 3

Dari Gambar 3.9 dapat dijelaskan bahwa ketika tombol *timer* 5s disentuh maka yang terjadi silinder akan maju setelah 5 detik. Ketika tombol *off* ditekan maka Silinder akan mundur

d.Percobaan 4:

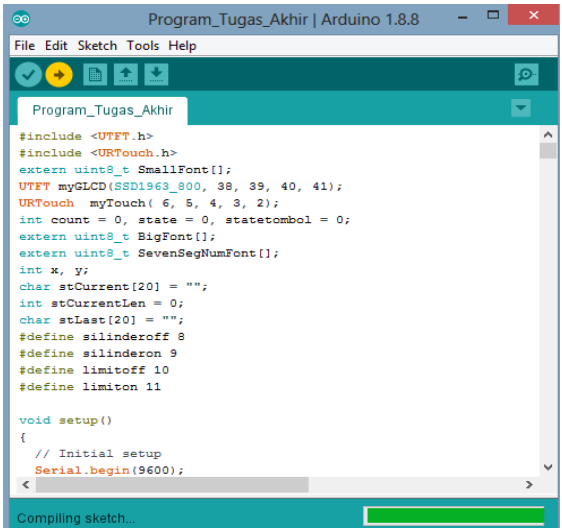


Gambar 3.10 Diagram Alur Percobaan 4

Dari Gambar 3.10 dapat dijelaskan bahwa ketika 3 *On Off* ditekan maka silinder akan bergerak maju dan mundur sebanyak 3 kali. ketika 5 *On Off* ditekan maka silinder akan bergerak maju dan mundur sebanyak 5 kali. Dan ketika Tombol 8 *On off* ditekan maka silinder akan bergerak maju mundur sebanyak 8 kali.

Diagram langkah inilah yang mendasari program Arduino. Langkah selanjutnya adalah membuat program arduino dengan *Software* Arduino IDE. Selanjutnya seletah program dibuat program

dari Arduino IDE *compile* terlebih dahulu seperti pada Gambar 3.11 untuk memastikan tidak *error*.



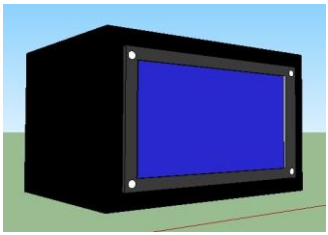
Gambar 3.11 *Compile Data* Arduino

Setelah itu *Upload* Program pada Arduino seperti pada Gambar 3.12

Gambar 3.12 *Upload* Program ke Arduino

3.2.2. Desain HMI

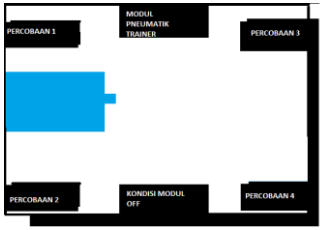
HMI disini dirancang dengan sederhana yaitu dimana relay dan arduino dimasukkan kedalam akrilik dengan bentuk balok dengan ukuran panjang sisi 15 cm, lebar 7 cm dan tinggi 9 cm. Kemudian LCD ditaruh di luar. Desain HMI dapat dilihat pada Gambar 3.13 .



Gambar 3.13 Desain HMI

3.2.2.1 Desain Halaman Awal

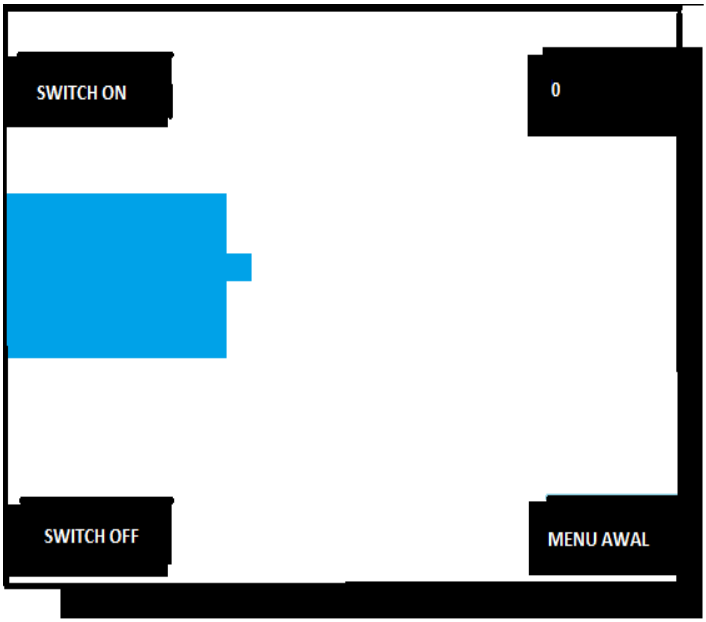
Halaman Awal adalah tampilan pertama pada *touchscreen* ketika pertama kali dihidupkan. Pada pojok kiri atas terdapat tombol untuk memulai percobaan 1. Pada pojok kiri bawah terdapat menu untuk memulai percobaan 2. Pada pojok kanan atas terdapat menu untuk memulai percobaan 3. Pada pojok kanan bawah terdapat menu untuk memulai percobaan 4. Pada halaman bawah.Desain halaman awal dapat dilihat pada Gambar 3.14



Gambar 3.14 Desain Halaman Awal

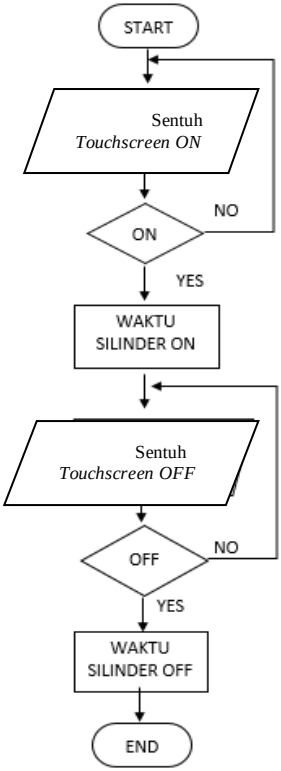
3.2.2.2 Desain HMI Percobaan 1

Pada percobaan 1 dapat dilihat pada pojok kiri bawah terdapat tombol *switch on* untuk menghidupkan pneumatik. Pada pojok kiri atas terdapat menu *switch off* untuk mematikan pneumatik. Pada pojok kanan atas terdapat satuan waktu untuk mengetahui selang waktu ketika pneumatik bekerja dari *off* ke *on* dan *On* ke *off*. Dan pada pojok kanan bawah terdapat menu awal untuk kembali pada halaman pertama. Menu Percobaan 1 dapat dilihat pada Gambar 3.15



Gambar 3.15 Menu Percobaan 1

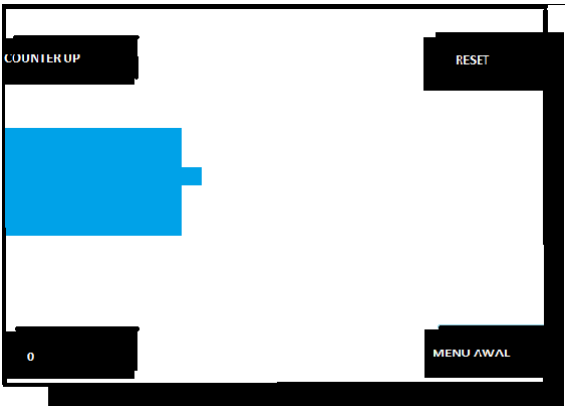
Ketika *start* masukan *touchscreen* akan menggerakkan maju silinder dan waktu akan tercatat saat silinder maju. Ketika ditekan masukan *touchscreen off* maka silinder akan bergerak mundur. *Flowchart* pada percobaan 1 dapat dilihat pada Gambar 3.16



Gambar 3.16 *Flowchart* Percobaan 1

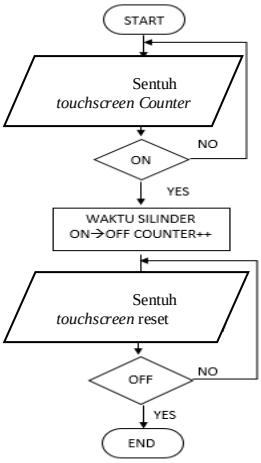
3.2.2.3 Desain HMI Percobaan 2

Pada percobaan 2 dapat dilihat pada pojok kiri atas terdapat tombol *Counter up* untuk membuat silinder maju mundur satu kali. Pada pojok kiri bawah terdapat nilai *counter* hasil *counter up*. Pojok kanan atas terdapat reset untuk mengulang nilai counter, Dan pada pojok kanan bawah terdapat menu awal untuk kembali pada halaman pertama. Menu Percobaan 2 dapat dilihat pada Gambar 3.16



Gambar 3.17 Menu Percobaan 2

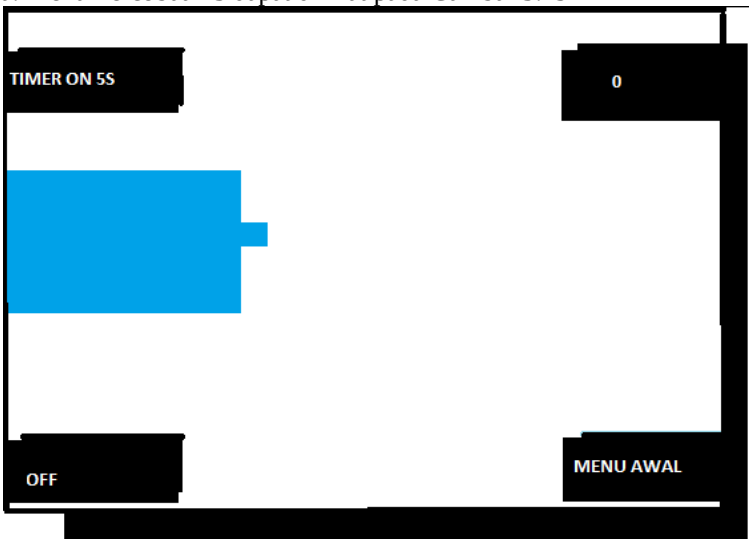
Ketika *start* maka masukan *touchscreen counter up* akan menggerakkan silinder maju dan mundur dan akan tercatat waktu tempuh silinder ketika maju mundur. Ketika *touchscreen* reset akan mengulang system kembali semula. *Flowchart* pada percobaan 2 bisa dilihat pada Gambar 3.18



Gambar 3.18 *Flowchart* Percobaan 2

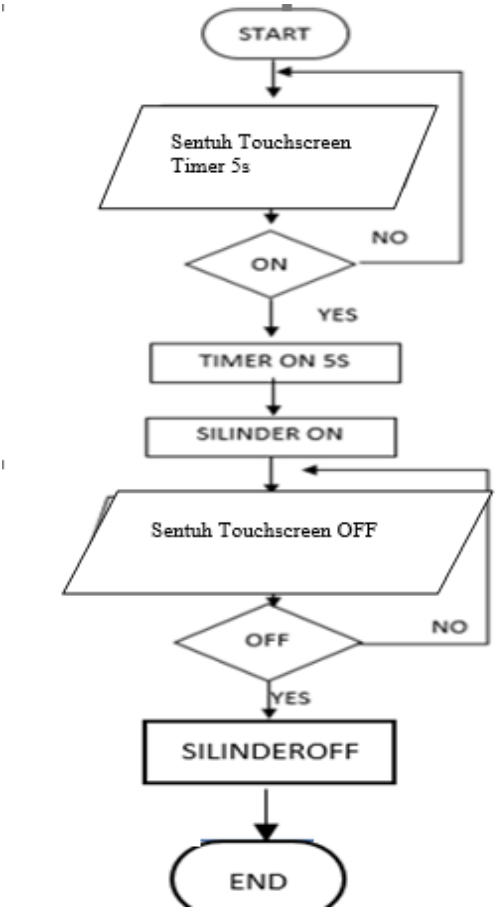
3.2.2.4 Desain HMI Percobaan 3

Pada percobaan 3 ini pojok kiri atas terdapat timer *on* 5s untuk memajukan silinder pneumatik dengan delay *on* 5 detik. Pojok kanan atas untuk silinder mundur. Pojok kiri bawah untuk nilai waktu saat silinder dari *off* ke *on* dan *on* ke *off*. Dan pada pojok kanan bawah terdapat menu awal untuk kembali pada halaman pertama. Menu Percobaan 3 dapat dilihat pada Gambar 3.19



Gambar 3.19 Menu Percobaan 3

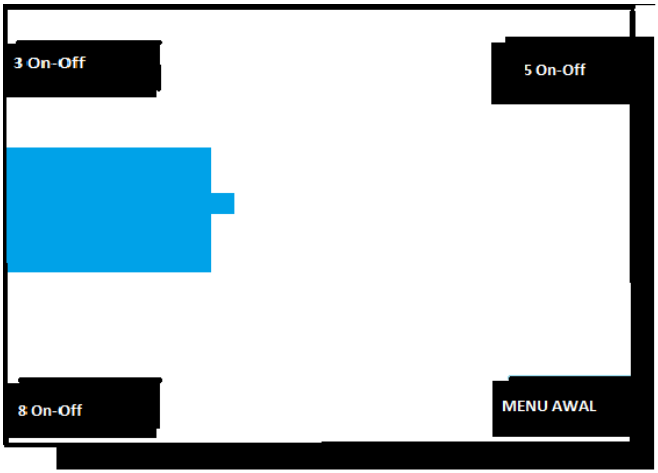
Pada saat *start* dan diberi masukan Timer 5s akan menghitung waktu selama 5 detik kemudian silinder akan bergerak maju. Ketika diberi masukan *off* pada *touchscreen* maka silinder akan bergerak mundur. *Flowchart* pada percobaan 3 bisa dilihat pada Gambar 3.20



Gambar 3.20 *Flowchart* Percobaan 3

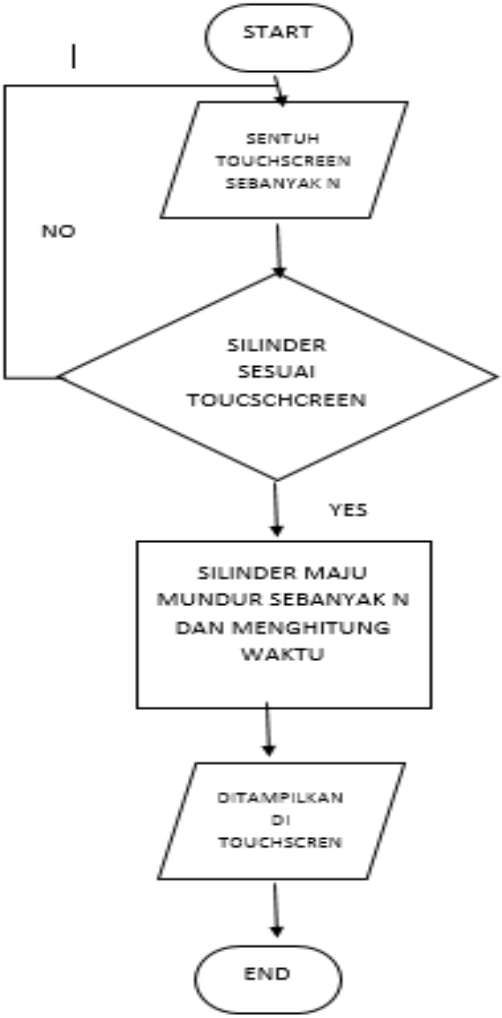
3.2.2.5 Desain HMI Percobaan 4

Pada percobaan 4 ini pojok kiri atas terdapat 3 *on off* yang membuat silinder maju mundur sebanyak 3 kali.. Pojok kanan atas untuk silinder maju mundur sebanyak 5 kali. Pojok kiri bawah Silinder maju mundur sebanyak 8 kali. Dan pada pojok kanan bawah terdapat menu awal untuk kembali pada halaman pertama. Menu Percobaan 4 dapat dilihat pada Gambar 3.21



Gambar 3.21 Tampilan Percobaan 4

Pada percobaan 4 ketika *start* maka masukan *touchscreen* 3 kali akan mengaktifkan silinder maju dan mundur 3 kali dan menghitung waktu tempuh selama silinder maju mundur 3 kali. Pada percobaan 4 ketika *start* maka masukan *touchscreen* 5 kali akan mengaktifkan silinder maju dan mundur 5 kali dan menghitung waktu tempuh selama silinder maju mundur 5 kali. Masukan *touchscreen* 8 kali akan mengaktifkan silinder maju dan mundur 8 kali dan menghitung waktu tempuh selama silinder maju mundur 8 kali *Flowchart* pada percobaan 4 dapat dilihat pada Gambar 3.22



Gambar 3.22 *Flowchart* Percobaan 4

BAB IV
PENGUJIAN DAN ANALISA

Setelah proses perancangan dan pembuatan alat selesai, kemudian dilakukan penggabungan *hardware* dan *software* alat sehingga menjadi sebuah sistem kerja. Kemudian dilakukan pengujian awal apakah perancangan alat telah sesuai dengan yang percobaan pengukuran dan menganalisa data. Berikut ini adalah pengukuran dan analisa data yang dilakukan.di kehendaki. Jika sudah sesuai, di lakukan suatu

4.1 Pengujian *Hardware*

Dalam pengujian *hardware* dibutuhkan alat ukur berupa avometer yang berguna untuk mempresentasikan kondisi dari komponen masih layak atau tidak kondisi komponen tersebut.

4.1.1 Pengujian Arduino

Pengujian ini dilakukan pada rangkaian board Arduino Mega dan disambungkan dengan *Relay*. Pengujian dilakukan untuk mengetahui tegangan *output* pada setiap *pin* arduino jika diberi *input high* dan disambungkan dengan *relay* 24 Volt dan diukur dengan menggunakan *voltmeter*. Hasil pengukuran pada tiap pin dari *board* Arduino Mega dapat dilihat pada Tabel 4.1

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

Pin Analog	Tegangan (Volt)
Pin A0	24
Pin A1	24
Pin A2	24
Pin A4	24

Dari hasil Tabel 4.1 dapat disimpulkan bahwa ketika board Arduino Mega diberi *input high* pada tiap *pin* maka tegangan yang dihasilkan bernilai rata-rata 24 Volt yang artinya tegangan yang dikeluarkan telah maksimal dan pin tersebut dapat berfungsi sebagai *supply* untuk beban yang diinginkan karena *output board* Arduino Mega bernilai *high* dengan tegangan 24 Volt.

4.1.2 Pengujian Spesifikasi Kompresor

Pada saat pengujian spesifikasi Kompresor,Kompresor harus terisi penuh,kemudian angin dialirkan melalui FRL,lalu angin melalui *Solenoid Valve* dan mengalir ke silinder.

Pada FRL dapat ditentukan tekanan udara yang akan dialirkan disilinder dan dapat dilihat kedalam *Lubricator* . Pada pengujian kali ini menggunakan tekanan 2 bar dan 3 bar.

Sebelum melakukan percobaan pastikan, kompresor dalam keadaan kapasitas angin di dalamnya penuh. Jika kapasitas angin di kompresor telah penuh maka siap di lakukan percobaan. Dimulai dengan mengalirkan tekanan angin ke FRL. Setelah FRL di aliri tekanan angin dari kompresor, FRL akan menunjukkan nilai dari tekanan angin yang masuk ke *plant* dan dapat mengatur berapa tekanan angin yang di inginkan (disini melakukan pengujian dengan mengatur tekanan angin mulai dari 2 bar dan 3 bar secara bergantian).Kemudian hitung berapa kali silinder dapat bergerak maju dan mundur. Berikut Tabel 4.2 hasil kemampuan masing masing silinder dalam beberapa tekanan yang berbeda. Spesifikasi Kompressor dapat dilihat pada Tabel 4.2

Tabel 4.2 Spesifikasi Kompresor

Percobaan	Tekanan	Silinder Maju	Silinder Mundur
1	2 Bar	1032 kali	1032 kali
	3 Bar	725 kali	725 kali
2	2 Bar	1100 kali	1100 kali
	3 Bar	745 kali	745 kali
Rata-rata	2 Bar	1066 kali	1066 kali
	3 Bar	735 kali	735 kali

Pada Percobaan Pengujian Kompressor kali ini,sebuah kompresor dengan kapasitas 36 liter jika digunakan untuk modul pneumatik dengan tekanan 2 bar silinder maju dan mundur rata-rata sebanyak 1066 kali dan tekanan 3 bar silinder maju dan mundur rata-rata 735 kali.

Program pengambilan data Tabel 4.2 menggunakan program timer solenoid valve 6 meyala setiap 1 detik sekali dengan arduino dengan program pada Gambar 4.1

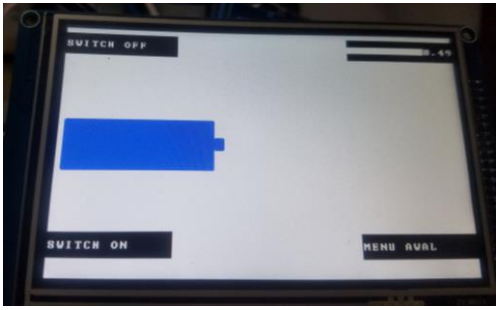
Gambar 4.1 Program Pengujian Kompresor

4.2 Pengujian *Software*

Pengujian *software* yaitu dengan melakukan beberapa uji coba untuk memastikan HMI yang sudah dibuat telah sesuai dengan yang di harapkan atau tidak.

4.2.1 Pengujian HMI Percobaan 1

Pada percobaan 1 Program HMI yang digunakan ialah arduino ID. Pengujian dilakukan untuk melihat apakah HMI telah sesuai dengan yang di harapkan. Mulai dari tampilan dan perintah logika yang dibuat. Gambar 4.2 memperlihatkan tampilan *design* HMI yang telah dibuat.



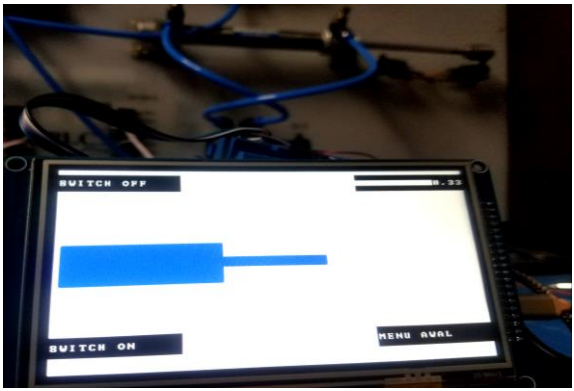
Gambar 4.2 Tampilan Design HMI Percobaan 1

Pada percobaan 1 ketika *Switch On* ditekan maka waktu silinder maju akan muncul pada HMI,kemudian ketika *switch off* disentuh waktu tempuh juga akan muncul pada HMI. Hasil percobaan 1 dapat dilihat pada Tabel 4.3

Tabel 4.3 Data Percobaan 1

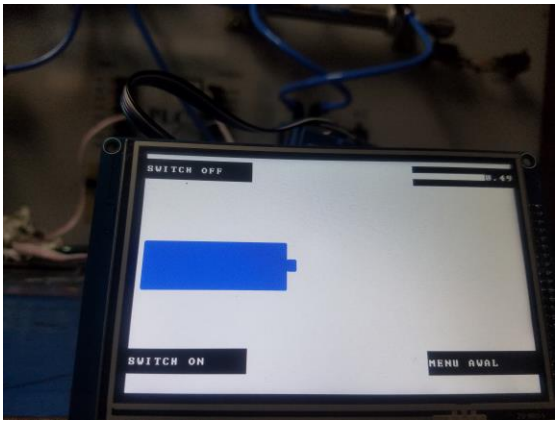
No	Silinder Maju (detik)	Silinder Mundur(detik)
1	0,33	0,49
2	0,35	0,5
3	0,34	0,5
4	0,33	0,49
5	0,34	0,49
Rata-rata	0,34	0,49

Dalam Percobaan 1 dapat dilihat pada Table 4.3 waktu tempuh silinder maju rata-rata adalah 0,34 detik dan rata rata waktu tempuh silinder mundur adalah 0,49 detik. Tampilan salah satu pengujian dapat dilihat pada Gambar 4.3



Gambar 4.3 Tampilan Salah Satu Pengujian HMI Percobaan 1 *on*

Dan ketika *Switch on* ditekan maka silinder pada modul pneumatik akan maju sesuai dengan HMI dan waktu yang tercatat ketika maju adalah sekian detik dan dapat dilihat pada Gambar 4.3. Ketika *Switch Off* ditekan maka silinder akan mundur sesuai dengan HMI dan waktu yang tercatat adalah sekian detik dan dapat dilihat pada Gambar 4.4



Gambar 4.4 Tampilan Salah Satu Penghujian HMI Percobaan 1 *Switch Off*

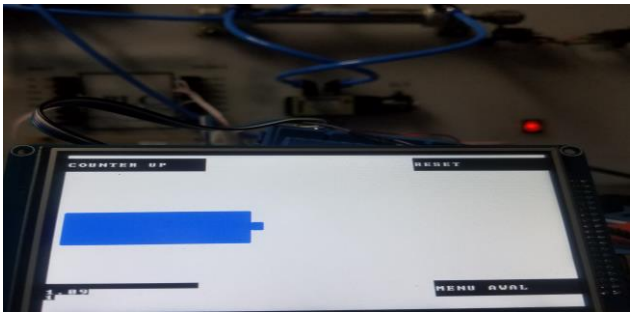
4.2.2 Pengujian HMI Percobaan 2

Pada percobaaa 2 adalah menampilkan karakteristik *counter up* dimana ketika *counter up* ditekan maka silinder akan maju dan mundur 1 kali dan tercatat pada bagian kanan bawah menampilkan jumlah maju mundur silinder dan menampilkan waktu yang ditempuh silinder untuk sekali maju mundur sehingga didapatkan data dan dapat dilihat pada Tabel 4.4

Tabel 4.4 Data Percobaan 2

No	Silinder Maju-Mundur (detik)
1	1,09
2	1,1
3	1,1
4	1,11
5	1,11
Rata-rata	1,1

ketika 1 kali maju mundur silinder rata-rata membutuhkan waktu 1,1 detik. Tampilan salah satu data Percobaan 2 dapat dilihat pada Gambar 4.5



Gambar 4.5 Tampilan Percobaan ke 2

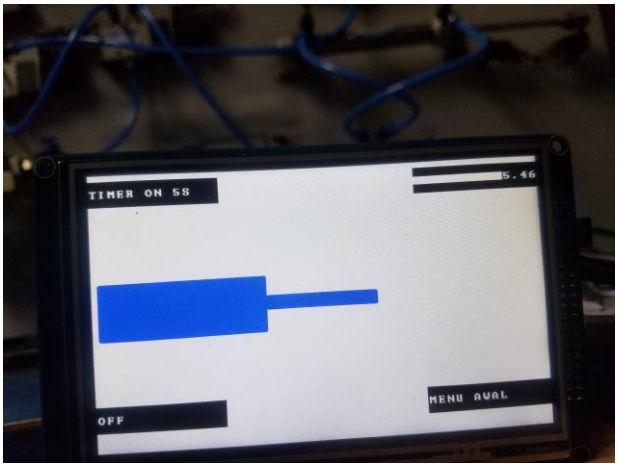
4.2.3 Pengujian HMI Percobaan 3

Pada Percobaan 3 adalah percobaan *Timer On Delay* yaitu adalah ketika *timer on 5s* ditekan maka arduino akan menunda waktu *ON* pada silinder selama 5 detik setelah itu menyala. Ketika Tombol *off* ditekan silinder akan menutup kembali. Dari percobaan 3 diperoleh data waktu yang dibutuhkan silinder untuk bergerak maju dapat dilihat pada Tabel 4.5

Tabel 4.5 Data Percobaan 3

No	Silinder Maju-Mundur (detik)
1	5,46
2	5,46
3	5,47
4	5,48
5	5,47
Rata-rata	5,47

Dari data yang didapat dapat disimpulkan waktu tempuh silinder hingga maksimal rata- rata adalah 5,47 detik. Tampilan pengujian salah satu percobaan ke 3 dapat dilihat pada Gambar 4.6



Gambar 4.6 Tampilan Pengujian Percobaan 3

4.2.4 Pengujian HMI Percobaan 4

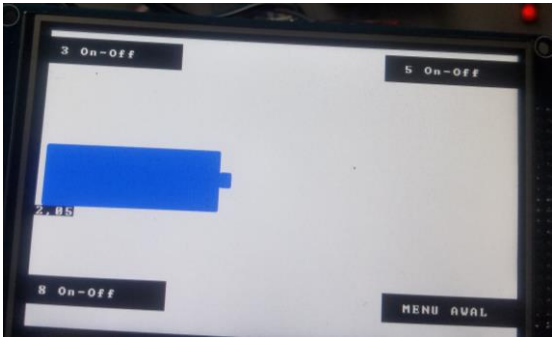
Pada percobaan 4 adalah percobaan dimana silinder akan bergerak maju mundur selama 3 kali,5 kali, dan 8 kali. Dari percobaan 4 diperoleh data waktu yang dibutuhkan silinder untuk bergerak maju dan mundur sebanyak 3,5 dan 8 kali dapat dilihat pada Tabel 4.6

Tabel 4.6 Data Percobaan 4

No	3x *On-Off* (detik)	5x *On-Off* (detik)	8x *on-off* (detik)
1	2	4,49	6,32
2	2,01	4,49	6,33
3	2,02	4,5	6,33
4	2,02	4,5	6,34
5	2,02	4,51	6,35
Rata-rata	2,01	4,5	6,34

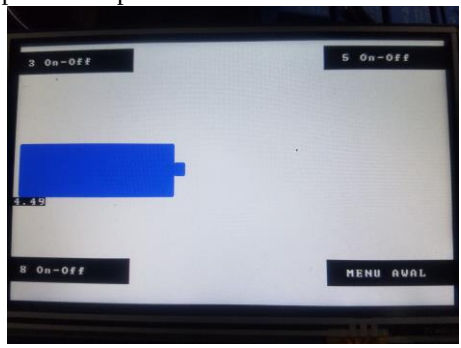
Dari data tersebut dapat disimpulkan bahwa ketika silinder maju mundur sebanyak 3 kali membutuhkan rata-rata waktu 2,01 Detik. Dan untuk maju mundur 5 kali membutuhkan waktu rata-rata 4,5 Detik. Dan untuk 8 kali maju mundur membutuhkan waktu rata-

rata 6.34 detik. Tampilan salah satu pengujian Percobaan 4 dapat dilihat pada Gambar 4.7



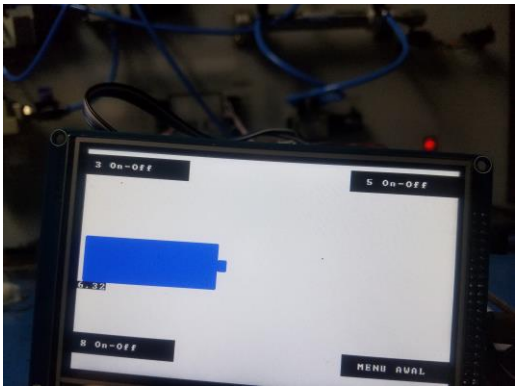
Gambar 4.7 Tampilan Percobaan 4 Saat 3 Kali Maju Mundur

Dari percobaan bagian 5 kali maju mundur didapatkan data waktu ketika silinder 5 kali maju mundur menempuh waktu 4,49 detik. Tampilan salah satu prngujian HMI percobaan 4 saat 5 kali maju mundur dapat dilihat pada Gambar 4.8



Gambar 4.8 Tampilan Salah Satu Pengujian Percobaan 4 Saat 5 Kali Maju Mundur

Dari percobaan bagian 8 kali maju mundur didapatkan data waktu ketika silinder 8 kali maju mundur menempuh waktu 4,49 detik. Tampilan HMI percobaan 4 saat 8 kali maju mundur dapat dilihat pada Gambar 4.9



Gambar 4.9 Tampilan Salah Satu Pengujian Percobaan 4 Saat 8 Kali Maju Mundur

--- Halaman ini sengaja dikosongkan ---

BAB V
PENUTUP

Setelah melakukan perancangan dan pembuatan alat serta pengujian dan analisa, maka dapat ditarik kesimpulan dan saran dari kegiatan yang telah dilakukan untuk pengembangan Proyek Akhir ini.

5.1 Kesimpulan

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

1. Silinder dapat bergerak dengan tekanan dari FRL 2 bar dan menempuh waktu 0,33 detik saat maju dan mundur 0,49 detik
2. Dalam 1 unit kompresor 36 liter angin dan dengan tekanan 2 bar silinder mampu bergerak maju mundur rata-rata sebanyak 1045 kali dan saat 3 bar.
3. HMI dapat menampilkan k*on*disi yang sesuai dengan modul pneumatik.

5.2 Saran

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

1. Perhatikan terlebih dahulu k*on*ektor HMI dengan pneumatik sebelum digunakan
2. Untuk mengambil data diperlukan ketelitian karena data bisa berubah sewaktu waktu dikarenakan factor tekanan pada kapasitas kompresor yang menurun dan pengaturan baut pada silinder yang berpengaruh terhadap kecepatan laju silinder.

--- Halaman ini sengaja dikosongkan ---

DAFTAR PUSTAKA

[1] Prasetyo, rudy dan Kamal, M.Imadudin. "Modul Training Pneumatik Menggunakan PLC dengan HMI W*on*derware InTouch", **Tugas Akhir**, Program D3 Teknik Elektro FTI-ITS, Surabaya, 2013.

[2] Prabowo Tri, **Panduan Belajar HMI,** Yogyakarta 2018

[3] Kadir A. **Panduan Praktis Mempelajari Aplikasi Mikrokontroler dan Arduino**, Yogyakarta,2013

[4] Prabowo, Agus Dian."Aplikasi HMI *Touchscreen* Untuk Modul Pneumatik Double Acting Silinder",**Tugas Akhir,**Program Departemen Teknik Elektro Otomasi Fakultas Vokasi-ITS,Surabaya,2018

LAMPIRAN A
LISTING PROGRAM

```
#include <UTFT.h>
#include <URTouch.h>
extern uint8_t SmallFont[];
UTFT myGLCD(SSD1963_800, 38, 39, 40, 41);
URTouch  myTouch( 6, 5, 4, 3, 2);
float hasilcount, count = 0, state = 0, statetombol = 0, timer = 0;
extern uint8_t BigFont[];
extern uint8_t SevenSegNumFont[];
int x, y;
char stCurrent[20] = "";
int stCurrentLen = 0, count1 = 0, count2 = 0, counter = 0;
char stLast[20] = "";
#define solenoid A0
#define silinderon A1
#define limitoff A2
#define limiton A4
int buttonState = 0;
int lastButtonState = 0;
int buttonState2 = 0;
int lastButtonState2 = 0;
unsigned long prevtime, prevtime2;
unsigned long currenttime, currenttime2;
void setup()
{
  // Initial setup
  Serial.begin(9600);
  pinMode(solenoid, OUTPUT);
  pinMode(silinderon, OUTPUT);
  pinMode(limitoff, INPUT_PULLUP);
  pinMode(limiton, INPUT_PULLUP);
  digitalWrite(solenoid, LOW);

  digitalWrite(silinderon, HIGH);
  myGLCD.InitLCD();
```

```
  myGLCD.clrScr();

  myTouch.InitTouch();
  myTouch.setPrecision(PREC_MEDIUM);
  myGLCD.setFont(BigFont);
  myGLCD.setBackColor(0, 0, 255);
  Serial.begin(9600);
  // Setup the LCD
  myGLCD.InitLCD();
  myGLCD.setFont(20);
  int buf[798];
  int x, x2;
  int y, y2;
  int r;
  halamanawal();
}
void halamanawal()
{
  digitalWrite(silinderon, HIGH);
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRect(0, 14, 799, 465);
  myGLCD.fillScr(255, 255, 255);
  myGLCD.setColor(255, 0, 0);
  myGLCD.fillRoundRect(10, 190, 300, 289);
  myGLCD.fillRoundRect(300, 225, 320, 250);

  myGLCD.setColor(255, 255, 255);
  myGLCD.setBackColor(0, 0, 0);
  myGLCD.setFont(BigFont);
  myGLCD.print("    MODUL     ", CENTER, 0);
  myGLCD.print("  PELATIHAN   ", CENTER, 15);
  myGLCD.print("  PNEUMATIK   ", CENTER, 30);
  myGLCD.print("KONDISI MODUL ", CENTER, 415);
  myGLCD.print("   OFF        ", CENTER, 430);
  myGLCD.print("              ", CENTER, 445);
  myGLCD.print("              ", LEFT, 20);
  myGLCD.print("PERCOBAAN 1   ", LEFT, 35);
  myGLCD.print("              ", LEFT, 50);
  myGLCD.print("              ", LEFT, 400);
```

```
  myGLCD.print("PERCOBAAN 2   ", LEFT, 415);
  myGLCD.print("              ", LEFT, 430);
  myGLCD.print("              ", RIGHT, 20);
  myGLCD.print("PERCOBAAN 3   ", RIGHT, 35);
  myGLCD.print("              ", RIGHT, 50);
  myGLCD.print("              ", RIGHT, 400);
  myGLCD.print("PERCOBAAN 4   ", RIGHT, 415);
  myGLCD.print("              ", RIGHT, 430);
  myGLCD.setColor(0, 255, 0);
  myGLCD.setBackColor(0, 0, 255);
  //delay(2000);
}
void duaoff()
{
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRect(0, 14, 799, 465);
  myGLCD.fillScr(255, 255, 255);
  myGLCD.setColor(255, 0, 0);
  myGLCD.fillRoundRect(10, 190, 300, 289);
  myGLCD.fillRoundRect(300, 225, 320, 250);
  myGLCD.setColor(255, 255, 255);
  myGLCD.setBackColor(0, 0, 0);
  myGLCD.setFont(BigFont);
  myGLCD.print("              ", LEFT, 10);
  myGLCD.print("COUNTER UP    ", LEFT, 25);
  myGLCD.print("              ", LEFT, 40);
  myGLCD.print("              ", LEFT, 400);
  myGLCD.printNumF(hasilcount, 2    , LEFT, 415);
  myGLCD.printNumI(counter, LEFT, 430);
  myGLCD.print("              ", RIGHT, 10);
  myGLCD.print("RESET         ", RIGHT, 25);
  myGLCD.print("              ", RIGHT, 40);
  myGLCD.print("              ", RIGHT, 400);
  myGLCD.print("MENU AWAL     ", RIGHT, 415);
  myGLCD.print("              ", RIGHT, 430);
  //delay(2000);
}
void duaon()
{
```

```
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRect(0, 14, 799, 465);
  myGLCD.fillScr(255, 255, 255);
  myGLCD.setColor(255, 0, 0);
  myGLCD.fillRoundRect(10, 190, 300, 289);
  myGLCD.fillRoundRect(300, 225, 500, 250);
  myGLCD.setColor(255, 255, 255);
  myGLCD.setBackColor(0, 0, 0);
  myGLCD.setFont(BigFont);
  myGLCD.print("              ", LEFT, 10);
  myGLCD.print("COUNTER UP    ", LEFT, 25);
  myGLCD.print("              ", LEFT, 40);
  myGLCD.print("              ", LEFT, 400);
  myGLCD.printNumF(hasilcount, 2      , LEFT, 415);
  myGLCD.printNumI(counter, LEFT, 430);
  myGLCD.print("              ", RIGHT, 10);
  myGLCD.print("RESET         ", RIGHT, 25);
  myGLCD.print("              ", RIGHT, 40);
  myGLCD.print("              ", RIGHT, 400);
  myGLCD.print("MENU AWAL     ", RIGHT, 415);
  myGLCD.print("              ", RIGHT, 430);
  //delay(2000);
}

void satuon()
{
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRect(0, 14, 799, 465);
  myGLCD.fillScr(255, 255, 255);
  myGLCD.setColor(255, 0, 0);
  myGLCD.fillRoundRect(10, 190, 300, 289);
  myGLCD.fillRoundRect(300, 225, 500, 250);
  myGLCD.setColor(255, 255, 255);
  myGLCD.setBackColor(0, 0, 0);
  myGLCD.setFont(BigFont);
  myGLCD.print("              ", LEFT, 10);
  myGLCD.print("SWITCH OFF    ", LEFT, 25);
  myGLCD.print("              ", LEFT, 40);
  myGLCD.print("              ", LEFT, 400);
```

```
  myGLCD.print("SWITCH ON     ", LEFT, 415);
  myGLCD.print("              ", LEFT, 430);
  myGLCD.print("              ", RIGHT, 400);
  myGLCD.print("MENU AWAL     ", RIGHT, 415);
  myGLCD.print("              ", RIGHT, 430);
  myGLCD.print("              ", RIGHT, 10);
  myGLCD.printNumF(count, 2, RIGHT, 25);
  myGLCD.print("              ", RIGHT, 40);
  //delay(2000);
}
void satuoff()
{
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRect(0, 14, 799, 465);
  myGLCD.fillScr(255, 255, 255);
  myGLCD.setColor(255, 0, 0);
  myGLCD.fillRoundRect(10, 190, 300, 289);
  myGLCD.fillRoundRect(300, 225, 320, 250);
  myGLCD.setColor(255, 255, 255);
  myGLCD.setBackColor(0, 0, 0);
  myGLCD.setFont(BigFont);
  myGLCD.print("              ", LEFT, 10);
  myGLCD.print("SWITCH OFF    ", LEFT, 25);
  myGLCD.print("              ", LEFT, 40);
  myGLCD.print("              ", LEFT, 400);
  myGLCD.print("SWITCH ON     ", LEFT, 415);
  myGLCD.print("              ", LEFT, 430);
  myGLCD.print("              ", RIGHT, 400);
  myGLCD.print("MENU AWAL     ", RIGHT, 415);
  myGLCD.print("              ", RIGHT, 430);
  myGLCD.print("              ", RIGHT, 10);
  myGLCD.printNumF(count, 2, RIGHT, 25);
  myGLCD.print("              ", RIGHT, 40);

  // myGLCD.print("              ", RIGHT, 400);
  //  myGLCD.print("              ", RIGHT, 415);
  //  myGLCD.print("              ", RIGHT, 430);
  //delay(2000);
}
```

```
//halaman PERCOBAAN KETIGA OFF
void tigaoff ()
{
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRect(0, 14, 799, 465);
  myGLCD.fillScr(255, 255, 255);
  myGLCD.setColor(255, 0, 0);
  myGLCD.fillRoundRect(10, 190, 300, 289);
  myGLCD.fillRoundRect(300, 225, 320, 250);
  myGLCD.setColor(255, 255, 255);
  myGLCD.setBackColor(0, 0, 0);
  myGLCD.setFont(BigFont);
  myGLCD.print("              ", LEFT, 10);
  myGLCD.print("TIMER ON 5S   ", LEFT, 25);
  myGLCD.print("              ", LEFT, 40);
  myGLCD.print("              ", RIGHT, 10);
  myGLCD.printNumI(count          , RIGHT, 25);
  myGLCD.print("              ", RIGHT, 40);
  myGLCD.print("              ", RIGHT, 400);
  myGLCD.print("MENU AWAL     ", RIGHT, 415);
  myGLCD.print("              ", RIGHT, 430);
  myGLCD.print("              ", LEFT, 400);
  myGLCD.print("OFF           ", LEFT, 415);
  myGLCD.print("              ", LEFT, 430);


  // myGLCD.print("              ", RIGHT, 400);
  //  myGLCD.print("              ", RIGHT, 415);
  //  myGLCD.print("              ", RIGHT, 430);
  //delay(2000);
}
/*{

  //delay(2000);
   }
*/
```

```
//halaman PERCOBAAN KETIGA ON
void tigaon ()
{
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRect(0, 14, 799, 465);
  myGLCD.fillScr(255, 255, 255);
  myGLCD.setColor(255, 0, 0);
  myGLCD.fillRoundRect(10, 190, 300, 289);
  myGLCD.fillRoundRect(300, 225, 500, 250);
  myGLCD.setColor(255, 255, 255);
  myGLCD.setBackColor(0, 0, 0);
  myGLCD.setFont(BigFont);
  myGLCD.print("              ", LEFT, 10);
  myGLCD.print("TIMER ON 5S   ", LEFT, 25);
  myGLCD.print("              ", LEFT, 40);
  myGLCD.print("              ", RIGHT, 10);
  myGLCD.print("5.46", RIGHT, 25);
  myGLCD.print("              ", RIGHT, 40);
  myGLCD.print("              ", RIGHT, 400);
  myGLCD.print("MENU AWAL     ", RIGHT, 415);
  myGLCD.print("              ", RIGHT, 430);
  myGLCD.print("              ", LEFT, 400);
  myGLCD.print("OFF           ", LEFT, 415);
  myGLCD.print("              ", LEFT, 430);


  // myGLCD.print("              ", RIGHT, 400);
  //  myGLCD.print("              ", RIGHT, 415);
  //  myGLCD.print("              ", RIGHT, 430);
  //delay(2000);
}

//halaman PERCOBAAN KE EMPAT OFF
void empatoff()
{
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRect(0, 14, 799, 465);
  myGLCD.fillScr(255, 255, 255);
  myGLCD.setColor(255, 0, 0);
```

```
  myGLCD.fillRoundRect(10, 190, 300, 289);
  myGLCD.fillRoundRect(300, 225, 320, 250);
  myGLCD.setColor(255, 255, 255);
  myGLCD.setBackColor(0, 0, 0);
  myGLCD.setFont(BigFont);

  myGLCD.print("              ", LEFT, 10);
  myGLCD.print(" 3 On-Off     ", LEFT, 25);
  myGLCD.print("              ", LEFT, 40);
  myGLCD.printNumF(hasilcount, 2, LEFT, 300);
  myGLCD.print("              ", LEFT, 400);
  myGLCD.print(" 8 On-Off     ", LEFT, 415);
  myGLCD.print("              ", LEFT, 430);

  myGLCD.print("              ", RIGHT, 10);
  myGLCD.print("  5 On-Off    ", RIGHT, 25);
  myGLCD.print("              ", RIGHT, 40);

  myGLCD.print("2,05", RIGHT, 400);

  myGLCD.print("              ", RIGHT, 400);
  myGLCD.print("  MENU AWAL   ", RIGHT, 415);
  myGLCD.print("              ", RIGHT, 430);


  // myGLCD.print("              ", RIGHT, 400);
  //  myGLCD.print("              ", RIGHT, 415);
  //  myGLCD.print("              ", RIGHT, 430);
  //delay(2000);
}
//halaman PERCOBAAN KE EMPAT ON
void empaton()
{
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRect(0, 14, 799, 465);
  myGLCD.fillScr(255, 255, 255);
  myGLCD.setColor(255, 0, 0);
  myGLCD.fillRoundRect(10, 190, 300, 289);
  myGLCD.fillRoundRect(300, 225, 500, 250);
```

```
  myGLCD.setColor(255, 255, 255);
  myGLCD.setBackColor(0, 0, 0);
  myGLCD.setFont(BigFont);
  myGLCD.print("              ", LEFT, 10);
  myGLCD.print(" 3 On-Off     ", LEFT, 25);
  myGLCD.print("              ", LEFT, 40);
  myGLCD.printNumF(hasilcount, 2, LEFT, 300);
  myGLCD.print("              ", LEFT, 400);
  myGLCD.print(" 8 On-Off     ", LEFT, 415);
  myGLCD.print("              ", LEFT, 430);

  myGLCD.print("              ", RIGHT, 10);
  myGLCD.print("  5 On-Off    ", RIGHT, 25);
  myGLCD.print("              ", RIGHT, 40);

  myGLCD.print("              ", RIGHT, 400);
  myGLCD.print("  MENU AWAL   ", RIGHT, 415);
  myGLCD.print("              ", RIGHT, 430);
  //delay(2000);
}
void resethalaman()
{

  halamanawal();
  state = 0;
}

void loop()
{

  //  buttonState = digitalRead(limitoff);
  //  buttonState2 = digitalRead(limiton);
  //  if (buttonState != lastButtonState)
  //  {
  //    currenttime = millis();
  //    prevtime = currenttime;
  //  }
  //  if (buttonState2 != lastButtonState2)
  //  {
```

```
//    count = (millis() - prevtime);
//    Serial.println(count);
//  }


  while (true)
  {
   if (myTouch.dataAvailable())
   {
    myTouch.read();
    x = myTouch.getX();
    y = myTouch.getY();
    //      if (buttonState != lastButtonState)
    //      {
    //        currenttime = millis();
    //        prevtime = currenttime;
    //      }
    //      if (buttonState2 != lastButtonState2)
    //      {
    //        timer = (millis() - prevtime);
    //        Serial.println(timer);
    //}
    //      Serial.print(x);
    //      Serial.print("   ");
    //      Serial.println(y);
    //      Serial.print(digitalRead(limitoff));
    //      Serial.print("   ");
    //      Serial.println(digitalRead(limiton));
    // int bacalimiton = digitalRead(limiton);
    //      Serial.println(bacalimiton);
    ////////////////////////////////////////////////////////////perc 4

    if ((x > 0 && x <= 44) && (y >= 160) && state == 0) //D kanan
bawah
    {
      digitalWrite(silinderon, HIGH);
      state = 6;
      empatoff();
      break;
```

```
    }

    if ((x > 0 && x <= 44) && (y >= 160) && state == 6  ) //D
kanan bawah  halaman awal
    {
     state = 0;
     empatoff();
     resethalaman();
    }

    if ((x >= 280) && (y > 0 && y <= 60) && state == 6)//limit on
kiri atas
    {
     for (int j = 1; j < 4; j++)
     {
      for (int i = 0; i < 1000000; i++)
      {
       buttonState = digitalRead(limitoff);
       digitalWrite(silinderon, LOW);
       if (buttonState == 1 && lastButtonState == 0)
       {
        currenttime = millis();
        prevtime = currenttime;
        Serial.println(currenttime);
        lastButtonState = 1;
        break;
       }
      }
      for (int i = 0; i < 1000000; i++)
      {
       buttonState2 = digitalRead(limiton);
       if (buttonState2 == 0 && lastButtonState2 == 0)
       {
        empaton();
        lastButtonState2 = 1;
        break;
       }
      }
      lastButtonState = 0;
```

```
      lastButtonState2 = 0;
      for (int i = 0; i < 1000000; i++)
      {
       buttonState = digitalRead(limiton);
       digitalWrite(silinderon, HIGH);
       if (buttonState == 1 && lastButtonState == 0)
       {
        lastButtonState = 1;
        break;
       }
      }
      for (int i = 0; i < 1000000; i++)
      {
       buttonState2 = digitalRead(limitoff);
       if (buttonState2 == 0 && lastButtonState2 == 0)
       {
        count = (millis() - prevtime) * 0.001;
        Serial.print(millis());
        Serial.print(" ");
        Serial.println(count / 1000);
        hasilcount = count * j;
        empatoff();
        lastButtonState2 = 1;
        break;
       }
      }
      lastButtonState = 0;
      lastButtonState2 = 0;
     }
    }
    if ((x >= 280) && (y >= 160) && state == 6)// D KANAN
ATAS

    {
     for (int j = 1; j < 6; j++)
     {
      for (int i = 0; i < 1000000; i++)
      {
       buttonState = digitalRead(limitoff);
```

```
  digitalWrite(silinderon, LOW);
  if (buttonState == 1 && lastButtonState == 0)
  {
    currenttime = millis();
    prevtime = currenttime;
    Serial.println(currenttime);
    lastButtonState = 1;
    break;
  }
 }
 for (int i = 0; i < 1000000; i++)
 {
  buttonState2 = digitalRead(limiton);
  if (buttonState2 == 0 && lastButtonState2 == 0)
  {
    empaton();
    lastButtonState2 = 1;
    break;
  }
 }
 lastButtonState = 0;
 lastButtonState2 = 0;
 for (int i = 0; i < 1000000; i++)
 {
  buttonState = digitalRead(limiton);
  digitalWrite(silinderon, HIGH);
  if (buttonState == 1 && lastButtonState == 0)
  {
    lastButtonState = 1;
    break;
  }
 }
 for (int i = 0; i < 1000000; i++)
 {
  buttonState2 = digitalRead(limitoff);
  if (buttonState2 == 0 && lastButtonState2 == 0)
  {
    count = (millis() - prevtime) * 0.001;
    Serial.print(millis());
```

```
            Serial.print(" ");
            Serial.println(count / 1000);
            hasilcount = count * j;
            empatoff();
            lastButtonState2 = 1;
            break;
           }
          }
          lastButtonState = 0;
          lastButtonState2 = 0;
         }
        }
       if ((x > 0 && x <= 44) && (y > 0 && y <= 60) && state == 6)
//D kiri bawah
       {
        for (int j = 1; j < 9; j++)
        {
         for (int i = 0; i < 1000000; i++)
         {
          buttonState = digitalRead(limitoff);
          digitalWrite(silinderon, LOW);
          if (buttonState == 1 && lastButtonState == 0)
          {
           currenttime = millis();
           prevtime = currenttime;
           Serial.println(currenttime);
           lastButtonState = 1;
           break;
          }
         }
         for (int i = 0; i < 1000000; i++)
         {
          buttonState2 = digitalRead(limiton);
          if (buttonState2 == 0 && lastButtonState2 == 0)
          {
           empaton();
           lastButtonState2 = 1;
           break;
          }
```

```
      }
      lastButtonState = 0;
      lastButtonState2 = 0;
      for (int i = 0; i < 1000000; i++)
      {
       buttonState = digitalRead(limiton);
       digitalWrite(silinderon, HIGH);
       if (buttonState == 1 && lastButtonState == 0)
       {
        lastButtonState = 1;
        break;
       }
      }
      for (int i = 0; i < 1000000; i++)
      {
       buttonState2 = digitalRead(limitoff);
       if (buttonState2 == 0 && lastButtonState2 == 0)
       {
        count = (millis() - prevtime) * 0.001;
        Serial.print(millis());
        Serial.print(" ");
        Serial.println(count / 1000);
        hasilcount = count * j;
        empatoff();
        lastButtonState2 = 1;
        break;
       }
      }
      lastButtonState = 0;
      lastButtonState2 = 0;
     }

    }

    ////////////perc 1 sudah bisa//////////////////
    if ((x >= 280) && (y > 0 && y <= 60) && state == 0) //A kiri
atas
    {
     digitalWrite(silinderon, HIGH);
```

```
      count = 0;
      state = 1;
      Serial.println("hal");
      satuoff();
     }
    if ((x >= 280) && (y > 0 && y <= 60) && state == 2 &&
digitalRead(limiton) == 0) //A kiri atas off          range 60 belum
dikurangi limiton ditaruh di if, limit off dengan
    {
      state = 1;
      Serial.println("on");
      for (int i = 0; i < 1000000; i++)
      {
        buttonState = digitalRead(limiton);
        digitalWrite(silinderon, HIGH);
        if (buttonState == 1 && lastButtonState == 0)
        {
          currenttime = millis();
          prevtime = currenttime;
          Serial.println(currenttime);
          lastButtonState = 1;
          break;
        }
      }
      for (int i = 0; i < 1000000; i++)
      {
        buttonState2 = digitalRead(limitoff);
        if (buttonState2 == 0 && lastButtonState2 == 0)
        {
          count = (millis() - prevtime) * 0.001;
          Serial.print(millis());
          Serial.print(" ");
          Serial.println(count / 1000);
          satuoff();
          lastButtonState2 = 1;
          break;
        }
      }
      timer = 0;
```

```
      lastButtonState = 0;
      lastButtonState2 = 0;
    }
    if ((x > 0 && x <= 44) && (y > 0 && y <= 60) && state == 1
&& digitalRead(limitoff) == 0) //A kiri bawah  on
    {
      state = 2;
      Serial.println("off");
      for (int i = 0; i < 1000000; i++)
      {
        buttonState = digitalRead(limitoff);
        digitalWrite(silinderon, LOW);
        if (buttonState == 1 && lastButtonState == 0)
        {
          currenttime = millis();
          prevtime = currenttime;
          Serial.println(currenttime);
          lastButtonState = 1;
          break;
        }
      }
      for (int i = 0; i < 1000000; i++)
      {
        buttonState2 = digitalRead(limiton);
        if (buttonState2 == 0 && lastButtonState2 == 0)
        {
          count = (millis() - prevtime) * 0.001;
          Serial.print(millis());
          Serial.print(" ");
          Serial.println(count / 1000);
          satuon();
          lastButtonState2 = 1;
          break;
        }
      }

      lastButtonState = 0;
      lastButtonState2 = 0;
    }
```

```
    if ((x > 0 && x <= 44) && (y >= 160) && (state == 1 || state ==
2) ) //A kanan bawah  halaman awal
    {
     digitalWrite(silinderon, HIGH);
     state = 0;
     satuoff();
     resethalaman();
     count = 0;
    }
    ////////////////////perc 2 sudah bisa//////////////////////////

    if ((x > 0 && x <= 44) && (y > 0 && y <= 60) && state == 0)
//B kiri bawah
    {
     digitalWrite(silinderon, HIGH);
     state = 3;
     duaoff();
    }
    if ((x >= 280) && (y > 0 && y <= 60) && state == 3 &&
digitalRead(limitoff) == 0) //B kiri atas counter tambah
    {
     for (int i = 0; i < 1000000; i++)
     {
      buttonState = digitalRead(limitoff);
      digitalWrite(silinderon, LOW);
      if (buttonState == 1 && lastButtonState == 0)
      {
       currenttime = millis();
       prevtime = currenttime;
       Serial.println(currenttime);
       lastButtonState = 1;
       break;
      }
     }
     for (int i = 0; i < 1000000; i++)
     {
      buttonState2 = digitalRead(limiton);
      if (buttonState2 == 0 && lastButtonState2 == 0)
```

```
    {
      duaon();
      lastButtonState2 = 1;
      break;
    }
  }
  lastButtonState = 0;
  lastButtonState2 = 0;
  for (int i = 0; i < 1000000; i++)
  {
    buttonState = digitalRead(limiton);
    digitalWrite(silinderon, HIGH);
    if (buttonState == 1 && lastButtonState == 0)
    {
      lastButtonState = 1;
      break;
    }
  }
  for (int i = 0; i < 1000000; i++)
  {
    buttonState2 = digitalRead(limitoff);
    if (buttonState2 == 0 && lastButtonState2 == 0)
    {
      count = (millis() - prevtime) * 0.001;
      Serial.print(millis());
      Serial.print(" ");
      Serial.println(count / 1000);
      hasilcount = count;
      counter++;
      duaoff();
      lastButtonState2 = 1;
      break;
    }
  }
  hasilcount = 0;
  lastButtonState = 0;
  lastButtonState2 = 0;
}
```

```
    if ((x >= 280) && (y >= 160) && state == 3) //B kanan atas reset
counter
    {
     hasilcount = 0;
     counter = 0;
     duaoff();
     digitalWrite(silinderon, HIGH);
    }
    if ((x > 0 && x <= 44) && (y >= 160) && state == 3 ) //B kanan
bawah  halaman awal
    {
     state = 0;
     hasilcount = 0;
     counter = 0;
     duaoff();
     resethalaman();
     digitalWrite(silinderon, HIGH);
    }
    ////////////////////////////perc 3//////////////////////////////
    if ((x >= 280) && (y >= 160) && state == 0) //C kanan atas
    {
     state = 4;
     count = 0;
     tigaoff();
    }
    if ((x >= 280) && (y > 0 && y <= 60 && state == 4) &&
digitalRead(limitoff) == 0) //C kiri atas on delay 5s
    {

     state = 5;
     delay(5000);
     for (int i = 0; i < 1000000; i++)
     {
      buttonState = digitalRead(limitoff);
      digitalWrite(silinderon, LOW);
      if (buttonState == 1 && lastButtonState == 0)
      {
       currenttime = millis();
       prevtime = currenttime;
```

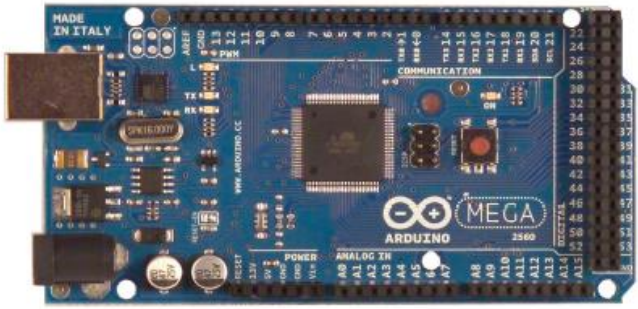
```
        Serial.println(currenttime);
        lastButtonState = 1;
        break;
       }
      }
      for (int i = 0; i < 1000000; i++)
      {
       buttonState2 = digitalRead(limiton);
       if (buttonState2 == 0 && lastButtonState2 == 0)
       {
        count = (millis() - prevtime) * 0.001 + 5000;
        Serial.print(millis());
        Serial.print(" ");
        Serial.println(count / 1000);
        tigaon();
        lastButtonState2 = 1;
        break;
       }
      }
      lastButtonState = 0;
      lastButtonState2 = 0;
     }
     if ((x > 0 && x <= 44) && (y > 0 && y <= 60) && state == 5
&& digitalRead(limiton) == 0) // C kiri bawah off
     {
      state = 4;
      for (int i = 0; i < 1000000; i++)
      {
       buttonState = digitalRead(limiton);
       digitalWrite(silinderon, HIGH);
       if (buttonState == 1 && lastButtonState == 0)
       {
        currenttime = millis();
        prevtime = currenttime;
        Serial.println(currenttime);
        lastButtonState = 1;
        break;
       }
      }
```

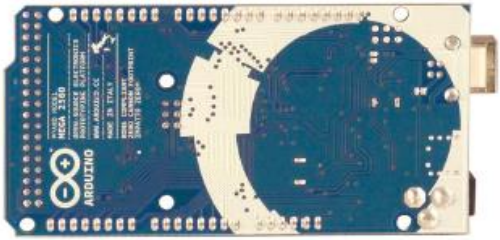
```
      for (int i = 0; i < 1000000; i++)
      {
       buttonState2 = digitalRead(limitoff);
       if (buttonState2 == 0 && lastButtonState2 == 0)
       {
        count = (millis() - prevtime) * 0.001;
        Serial.print(millis());
        Serial.print(" ");
        Serial.println(count / 1000);
        tigaoff();
        lastButtonState2 = 1;
        break;
       }
      }
      lastButtonState = 0;
      lastButtonState2 = 0;
     }
     if ((x > 0 && x <= 44) && (y >= 160) && (state == 4 || state ==
5) ) //C kanan bawah  halaman awal
     {
      state = 0;
      tigaoff();
      resethalaman();
     }
     if (state == 7 && limiton == 0)
     {
      state = 6;
      empaton();
      delay(1000);
      empatoff();
     }
    }
   }


}
```

LAMPIRAN B
DATASHEET

1.Datasheet Arduino Mega

Arduino Mega 2560





Overview

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

Schematic & Reference Design

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

Schematic: arduino-mega2560-schematic.pdf

Summary

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

Power

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

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

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

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

The power pins are as follows:

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

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

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

- **GND.** Ground pins.

Memory

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

Input and Output

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

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

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

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

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

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

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

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

There are a couple of other pins on the board:

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

Communication

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

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

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

Programming

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

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

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

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

Automatic (Software) Reset

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

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

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

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

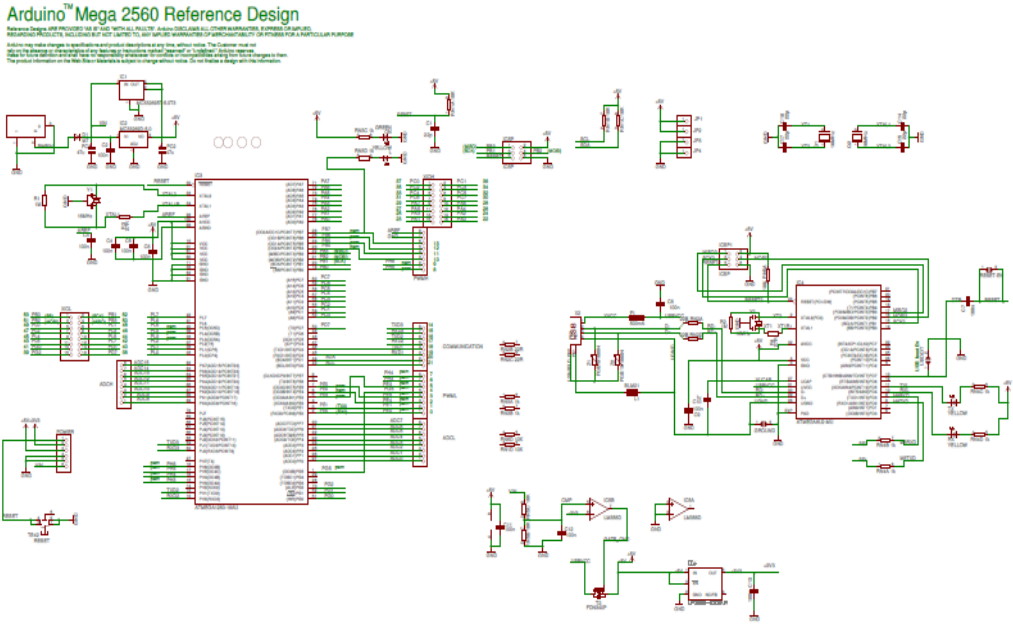
USB Overcurrent Protection

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

Physical Characteristics and Shield Compatibility

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

The Mega2560 is designed to be compatible with most shields designed for the Uno, Diecimila or Duemilanove. Digital pins 0 to 13 (and the adjacent AREF and GND pins), analog inputs 0 to 5, the power header, and ICSP header are all in equivalent locations. Further the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1 (pins 2 and 3 respectively). SPI is available through the ICSP header on both the Mega2560 and Duemilanove / Diecimila. *Please note that 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

EastRising
buydisplay.com

TFT LCD Module Datasheet ER-TFTM050-2

1. ORDERING INFORMATION

1.1 Order Number

Part Number(Order Number)	Description
ER-TFTM050-2	5" TFT LCD Display with RA8875 Controller Board
ER-DBTM050-2	8051 Microcontroller Development Board & Kit for ER-TFTM050-2
ER-TFTM050-2-4125	Arduino Shield for ER-TFTM050-2

1.2 Display Image

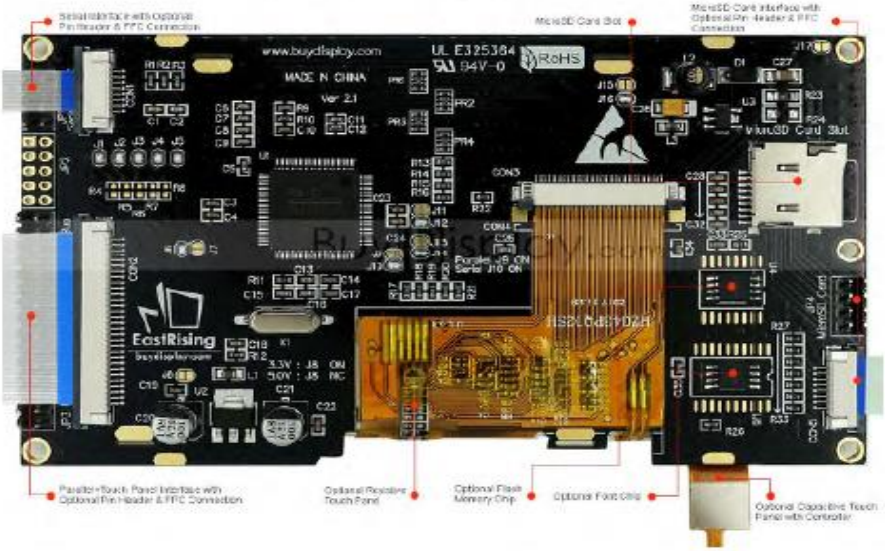
← ER-TFTM050-2 with No Touch Panel

ER-TFTM050-2 with Resistive Touch Panel →

← ER-TFTM050-2 with Capacitive Touch Panel

1.3 Controller Board Image

ER-TFTM050-2 Controller Board with Optional Capacitive & Resistive Touch Panel



2. SPECIFICATION

2.1 Display Specification

Item	Standard Value	Unit
Display Format	480 (RGB) x 272 Dots	--
Display Connector	FFC or Pin Header	--
Operating Temperature	-20 ~ +70	℃
Storage Temperature	-30 ~ +80	℃
Touch Panel Optional	Yes	--
Sunlight Readable	No	--

2.2 Mechanical Specification

Item	Standard Value	Unit
Diagonal Size	5	inch
Outline Dimension(PCB)	132.7 x 75.8	mm
Active Area	110.88 x 62.832	mm
Dot Pitch	0.231(W)X0.231(H)	mm
Gross Weight	0.172	kg

2.3 Electrical Specification

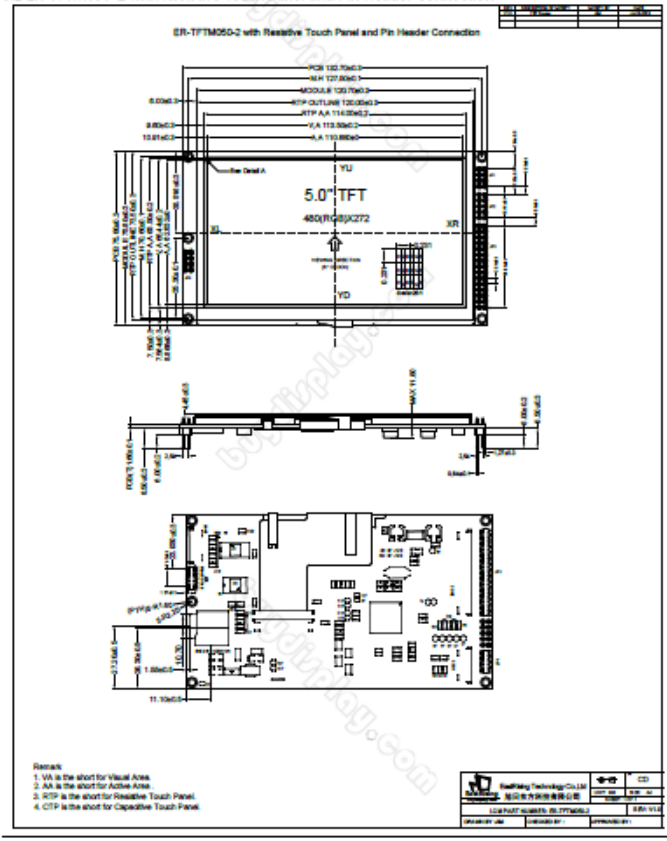
Item	Standard Value	Unit
IC Package	SMT	--
Controller	RA8875	--
Interface	8080/6800 8-bit/16-bit Parallel, 3-wire,4-wire SPI ,I2C	--
Response Time (Typ)	30	ms

2.4 Optical Specification

Item	Standard Value	Unit
LCD Type	TFT-LCD / Transmissive / Negative	--
Viewing Angle Range	Left:70, Right:70, Up:50, Down:50	deg
Colors	256/65K	--
Contrast Ratio (Typ)	500:1	--
Brightness (Typ)	300	cd/m2

8. OUTLINE DRAWING

3.1 ER-TFTM050-2 with Resistive Touch Panel and Pin Header Connection



4. ELECTRICAL SPEC

4.1 Pin Configuration-JP3/CON2 (8080/6800 8-bit/16-bit Parallel Interface)

Pin No	Symbol	Descriptions
1~2	VSS	Ground
3~4	VDD	Power Supply
5	E_/RD	Enable/Read Enable When MCU interface (I/F) is 8080 series, this pin is used as RD# signal (Data Read) , active low. When MCU I/F is 6800 series, this pin is used as EN signal (Enable), active high
6	R/W_/WR	Write/Read-Write When MCU interface is 8080 series, this pin is used as WR# signal (data write) , active low. When MCU interface is 6800 series, this pin is used as RW# signal (data Read/Write control) . Active high for read and active low for write.
7	/CS	Chip Select Input .Low active chip select pin.
8	RS	Command / Data Select Input The pin is used to select data/command cycle. RS = 0, data Read/Write cycle is selected. RS = 1, status read/command write cycle is selected. In 8080 interface, usually it connects to "A0" address pin.
9	WAIT	Wait Signal Output This is a WAIT# output to indicate the RA8875 is in busy state. The RA8875 can' t access MCU cycle when WAIT# pin is active. It is active low and could be used for MCU to poll busy status by connecting it to I/O port. I
10	INT	Interrupt Signal Output The interrupt output for MCU to indicate the status.
11	/RESET (NC)	Master synchronizes reset, Active Low. RC Reset circuit on board.
12	C86(NC)	MCU Interface Select 0: 8080 interface is selected 1: 6800 interface is selected Internally connected to the low level.
13	VSS	Ground
14	BL_CONTROL	Backlight control signal input. When using the internal PWM signal this pin floating.
15-30	DB0-DB15	Data Bus These are data bus for data transfer between MCU and RA8875. When setting register number and register data, DB[7:0] is used. When writing data to display RAM, DB[15:0] is used according to data bus mode setting. DB[15:8] will be input and should be pull-low or pull high. when 8-bit data bus mode is used.

4.2 Pin Configuration –JP1/CON1(3-wire,4-wire SPI Serial and I2C interface)

Pin No	Symbol	Descriptions
1~2	VSS	Ground
3~4	VDD	Power Supply
5	/SCS	SPI Chip Select. Chip select pin for 3-wire or 4-wire serial I/F.
6	SDO	3-wire SPI Data /4-wire SPI Data Output 4-wire SPI I/F: Data output for serial I/F. 3-wire SPI I/F: Bi-direction data for serial I/F IIC I/F: NC, if no use, please keep floating. If no use, please keep floating.
7	SDI	IIC data /4-wire SPI Data Input 4-wire SPI I/F: Data input for serial I/F. 3-wire SPI I/F: NC IIC I/F: Bi-direction data for serial I/F
8	SCLK	SPI Clock 3-wire, 4-wire Serial or IIC I/F clock.

Note: We disable serial and I2C interface by default. Please follow the below jump point description to enable.

4.3 Pin Configuration-JP2(Keyboard Interface)

Pin No	Symbol	Descriptions
1~5	KIN0-KIN4 GPI0-GPI4	Keypad Data Line or GPIs (General Purpose Input) Keypad data inputs (Default). They could be programmed as GPIs by register setting.
6~9	KOUT0-KOUT3 GPO0-GPO3	Keypad Strobe Line or GPOs (General Purpose Output) Keypad matrix strobe lines outputs with open-drain. (Default). They could be programmed as GPOs by register setting, if don' t use, please keep floating.
10	PWM2	PWM signal output

Note: The keyboard is disabled by default. Please leave J1 to J5 open and R4 to R8 is with 100KΩ if you want to enable.

4.4 Pin Configuration-JP3/CON5 (Micro SD Card Interface)

SD Mode		SPI Mode	
Pin No	Symbol	Pin No	Symbol
1	DATA2	1	NC
2	DATA3	2	/CS
3	CMD	3	DIN
4	CLK	4	SCLK
5	GND	5	GND
6	DATA0	6	DOUT
7	DATA1	7	NC
8	CARD DETECTION	8	CARD DETECTION

4.5 Jump Point Description

Function Description	Jump Method
Enable/Disable Keyboard	Disable Keyboard: J1~J5 Short,R4~R8 No Connection Enable Keyboard: J1~J5 Open,R4~R8=100KΩ
Power Supply Switch	Vdd=3.3V Power Supply : J8 Short Vdd=5V Power Supply : J8 Open
8080 Parallel Interface	J6,J9,J12,J13,J15 Short J7,J10,J11,J14,J16 Open
6800 Parallel Interface	J7,J9,J12,J13,J15 Short J6,J10,J11,J14,J16 Open
I2C Interface	J6,J10,J11,J14,J16 Short J7,J9,J12,J13,J15 Open
3-wire Serial Interface	J6,J10,J11,J13,J16 Short J7,J9,J12,J14,J15 Open
4-wire Serial Interface	J6,J10,J12,J14,J16 Short J7,J9,J11,J13,J15 Open
Backlight Control	J15 Short,J16 Open: Select Backlight Control Signal with External Input J15 Open,J16 Short: Select Backlight Control Signal with RA8875 'PWM'
Connect Fix Hole to Ground	J17 Shot: Connect Fix Hole to the Ground

Note: We leave J1 to,J5,J6,J9,J12,J13,J15 short and J7,J8,J10,J11,J14,J16,J17 open by default

4.6 Absolute Maximum Ratings

Item	Symbol	Min.	Typ.	Max.	Unit
Supply Voltage	VDD	-0.5	-	+5.5	V
Logic Signal Voltage	VDDIO	-0.5	-	+3.3	V
Operating Temperature	Top	-20	-	+70	℃
Storage Temperature	TST	-30	-	+80	℃
Humidity	RH	-		90%(Max60℃)	RH

4.7 Electrical Characteristics

Item	Symbol	Min.	Typ.	Max.	Unit
Power supply voltage(*1)	VDD	3.0	3.3	3.6	V
		4.8	5.0	5.2	V
Logic signal I/O voltage	VDDIO	3.0	3.3	3.6	V
Input voltage 'H' level	VIH	0.8VDDIO	-	VDDIO	V
Input voltage 'L' level	VIL	GND	-	0.2VDDIO	V
Output voltage 'H' level	VOH	VDDIO-0.4	-	VDDIO	V
Output voltage 'L' level	VCL	GND	-	GND+0.4	V
Schmitt-Trigger Input (*2)					
Input voltage 'H' level	VIH	0.7VDDIO	-	VDDIO	V
Input voltage 'L' level	VIL	GND	-	0.3VDDIO	V
Input Leakage Current 1 (*3)	VIH	--	+	+2	uA
Input Leakage Current 2 (*3)	VIL	--		-2	uA
Module Current	IDD(3.3V)	--	--	450	mA
	IDD(5.0V)	--	--	280	mA
Sleep Mode (*3)	ISLP	--	320	--	uA

Note1: Short J8 if VDD=3.3V

Note2: Signals RD#, WR#, CS#, RS, RST# are inputs of Schmitt-trigger.

Note3: Oscillator Clock = 20MHz, System Clock = 20~60MHz,VSYNC = 45~65Hz, TA=25 ℃

5. INSPECTION CRITERIA

5.1 Acceptable Quality Level

Each lot should satisfy the quality level defined as follows

Partition	AQL	Definition
A. Major	0.4%	Functional defective as product
B. Minor	1.5%	Satisfy all functions as product but not satisfy cosmetic standard

5.2 Definition of Lot

One lot means the delivery quantity to customer at one time.

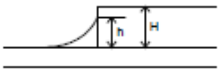
5.3 Condition of Cosmetic Inspection

- INSPECTION AND TEST
  - -FUNCTION TEST
  - -APPEARANCE INSPECTION
  - -PACKING SPECIFICTION

- INSPECTION CONDITION
  - - Put under the lamp (20W) at a distance 100mm from
  - - Tilt upright 45 degree by the front (back) to inspect LCD appearance.

- AQL INSPECTION LEVEL
  - - SAMPLING METHOD: MIL-STD-105D
  - - SAMPLING PLAN: SINGLE
  - - MAJOR DEFECT: 0.4% (MAJOR)
  - - MINOR DEFECT: 1.5% (MINOR)
  - - GENERAL LEVEL: II/NORMAL

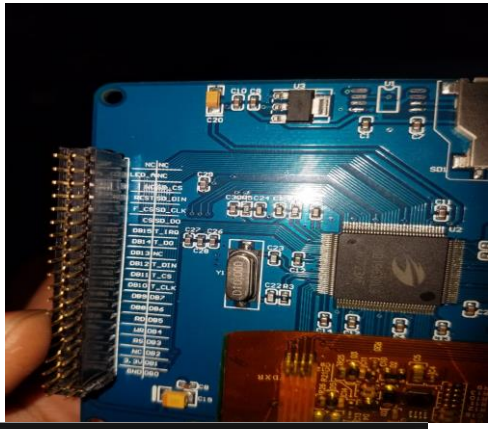
5.4 Module Cosmetic Criteria

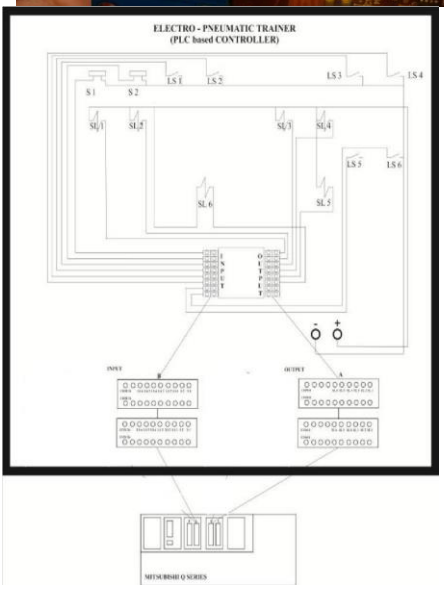
No.	Item	Judgment Criterion	Partition
1	Difference in Spec.	None allowed	Major
2	Pattern Peeling	No substrate pattern peeling and floating	Major
3	Soldering Defects	No soldering missing	Major
		No soldering bridge	Major
		No cold soldering	Minor
4	Resist Flaw on Substrate	Invisible copper foil(∉ 0.5mm or more)on substrate pattern	Minor
5	Accretion of Metallic	No soldering dust	Minor
	Foreign Matter	No accretion of metallic foreign matters(Not exceed ∉ 0.2mm)	
6	Stain	No stain to spoil cosmetic badly	Minor
7	Plate Discoloring	No plate fading, rusting and discoloring	Minor
8	Solder Amount 1.Lead Parts	a. Soldering side of PCB Solder to form a' Filet' all around t Solder should not hide the lead form  b.Components side (In case of 'Through Hole PCB') Solder to reach the Components side of PCB 	Minor
	2.Flat Packages	Either 'toe' (A) or 'heal' (B) of the lead to be covered by Filet'  Lead form to be assume over solder.	Minor
	3.Chips	(3/2) H≥h≥(1/2)H 	Minor

9	Backlight Defects	1.Light fails or flickers.(Major) 2. Color and luminance do not correspond to specifications. (Major) 3.Exceeds standards for display' s blemishes, foreign matter, dark lines or scratches.(Minor)	See list ←
10	PCB Defects	Oxidation or contamination on connectors.* 2. Wrong parts, missing parts, or parts not in specification.* 3.Jumpers set incorrectly.(Minor) 4.Solder(if any)on bezel, LED pad, zebra pad, or screw hole pad is not smooth.(Minor) *Minor if display functions correctly. Major if the display fails.	See list ←
11	Soldering Defects	1. Unmelted solder paste. 2. Cold solder joints, missing solder connections, or oxidation.* 3. Solder bridges causing short circuits.* 4. Residue or solder balls. 5. Solder flux is black or brown. *Minor if display functions correctly. Major if the display fails.	Minor

--- Halaman ini sengaja dikosongkan ---

LAMPIRAN C
DOKUMENTASI






ELECTRO-PNEUMATIC TRAINER
PLC
SWAN

RIWAYAT PENULIS

Nama : Bayu Atma Wirandana
TTL : Surabaya 28 mei 1997
Jenis Kelamin : Laki - Laki
Agama : Islam
Alamat Asal : Perum Bumi Gedangan Indah A-48,Sidoarjo
Telp/HP : 085856055751
E-mail : *atma512@gmail.com*

RIWAYAT PENDIDIKAN

- *2003 – 2009 : SDN Tembok Dukuh 1 Surabaya*
- *2009 – 2012 : SMPN 5 Sidoarjo*
- *2012 – 2015 : SMAN 3 Sidoarjo*
- *2015 – Sekarang : Departemen Teknik Elektro Otomasi Institut Teknologi Sepuluh Nopember.*

PENGALAMAN KERJA

- *Kerja Praktek PT Aneka Gas Industri*
- *Kerja Praktek PT. MSW BARU (Juli 2017 – Agustus 2017)*

PENGALAMAN ORGANISASI

- *Staff ITS Billiard 16/17*
- *Staff Ahli ITS Billiard 17/18*

--- Halaman ini sengaja dikosongkan ---