**RANCANG BANGUN PROTOTIPE *MASTER CONTROLLER***

**UNTUK MENGENDALIKAN *ARM MANIPULATOR* 6 DOF**

**BERBASIS MIKROKONTROLER *ARDUINO***

**PROYEK AKHIR**

Disusun sebagai salah satu syarat untuk

menyelesaikan pendidikan Diploma III

**Oleh:**

Raihan Ramadhan

219341043



**PROGRAM STUDI TEKNIK MEKATRONIKA**

**JURUSAN TEKNIK OTOMASI MANUFAKTUR DAN MEKATRONIKA**

**POLITEKNIK MANUFAKTUR BANDUNG**

**2022**

**LEMBAR PENGESAHAN**

**RANCANG BANGUN PROTOTIPE *MASTER CONTROLLER***

**UNTUK MENGENDALIKAN *ARM MANIPULATOR* 6 DOF**

**BERBASIS MIKROKONTROLER *ARDUINO***

**Oleh:**

Raihan Ramadhan

219341043

Program Studi Teknik Mekatronika

Jurusan Teknik Otomasi Manufaktur dan Mekatronika

Politeknik Manufaktur Bandung

Menyetujui

Tim Pembimbing

|  |  |
| --- | --- |
| Pembimbing I,  **Dr. Setyawan Ajie Sukarno, S.S.T., M.T., M.Sc.Eng**  **NIP. 198004282008101001** | Pembimbing II,  **Dr.Susetyo Bagas Bhaskoro, S.S.T., M.T.**  **NIP. 198706222015041002** |

# 

# ABSTRAK

Seiring dengan kemajuan otomatisasi teknologi maka perkembangan akan kebutuhan robot *arm manipulator* sangat diperlukan. Pada penggunaannya *arm manipulator* banyak diperlukan pada bidang industri, kedokteran dan juga pada robot bawah laut atau *Underwater-Remotly Operated Vehicle* (ROV). Untuk dapat mengendalikan sebuah manipulator diperlukan suatu pengendali jarak jauh yang dikendalikan oleh pengguna secara manual. Projek ini berjudul rancang bangun prototipe *master controller* untuk mengendalikan *arm manipulator* 6 DoF berbasis mikrokontroler Arduino, dimana perangkat kontroler menggunakan Arduino Mega, dan sensor potensiometer sebagai input pergerakan pada 6 buah motor stepper penggerak. Hasil pengujian memberikan capaian bahwa robot dapat menerima data oleh sebuah pendant pengendali secara lancar. Kemudian respons pada pergerakkan *arm manipulator* 6 DoF dicari berdasarkan data pergerakkan dan kecepatan motor stepper. Hasil yang didapatkan menunjukkan bahwa respons pengiriman data pada kecepatan di bawah 300mS tidak dapat dieksekusi secara baik oleh arm manipulator. Selanjutnya pada pengujian akurasi, motor stepper diberikan input data pergerakan sudut oleh *master controller* dan mendapatkan rata-rata error sebesar 0.287%.

**Kata Kunci:** *Underwater ROV (Remotly Operated Vehicle), Master controller, Arduino*

# *ABSTRACT*

*Along with advances in technology automation, the development of the need for arm manipulator robots is very necessary. In its use, arm manipulators are widely needed in the fields of industry, medicine and also in underwater robots or Underwater-Remotly Operated Vehicles (ROV). To be able to control a manipulator requires a remote controller that is controlled by the user manually. This project is entitled the design of a master controller prototype to control a 6 DoF arm manipulator based on an Arduino microcontroller, where the controller device uses an Arduino Mega, and a potentiometer sensor as input for the movement of 6 driving stepper motors. The test results provide an achievement that the robot can receive data by a pendant controller smoothly. Then the response to the movement of the 6 DoF arm manipulator is sought based on the movement data and the speed of the stepper motor. The results obtained indicate that the response of sending data at speeds below 300mS cannot be executed properly by the arm manipulator. Furthermore, in the accuracy test, the stepper motor is given input of angular movement data by the master controller and gets an average error of 0.287%.*

***Keywords:*** *Underwater ROV (Remotly Operated Vehicle), Master controller, Arduino*

# KATA PENGANTAR

Segala puji bagi Allah yang hanya kepadaNya kami memuji, memohon pertolongan, dan mohon keampunan. Kami berlindung kepadaNya dari kekejian diri dan kejahatan amalan kami. Barang siapa yang diberi petunjuk oleh Allah maka tidak ada yang dapat menyesatkan, dan barang siapa yang tersesat dari jalanNya maka tidak ada yang dapat memberinya petunjuk. Dan aku bersaksi bahwa tiada sembahan yang berhak disembah melainkan Allah saja, yang tiada sekutu bagi-Nya. Dan aku bersaksi bahwa Muhammad adalah hamba-Nya dan Rasul-Nya.

Atas petunjukan dan pertolongan-Nya, Alhamdlillah penulis dapat menyelesaikan proyek akhir ini dengan judul: “RANCANG BANGUN PROTOTIPE *MASTER CONTROLLER* UNTUK MENGENDALIKAN *ARM MANIPULATOR* 6 DOF BERBASIS MIKROKONTROLER *ARDUINO*”.

Proyek akhir dibuat dalam rangka memenuhi salah satu syarat menyelesaikan pendidikan Diploma III pada Program Studi Teknik Mekatronika di Politeknik Manufaktur Bandung.

Terselesaikannya proyek akhir ini tidak terlepas dari bantuan banyak pihak, sehingga pada kesempatan ini dengan segala kerendahan hati dan penuh rasa hormat penulis menghaturkan terima kasih yang sebesar-besamya bagi semua pihak yang telah memberikan bantuan moril maupun materil baik langsung maupun tidak langsung dalam penyusunan skripsi ini hingga selesai, terutama kepada yang saya hormati:

1. Direktur Politeknik Manufaktur Bandung, Bapak Mohammad Nurdin, S.T., M.A.B.
2. Ketua Jurusan Teknik Otomasi Manufaktur dan Mekatronika, Bapak Ismail Rokhim, ST., MT.
3. Ketua Program Studi D-III Teknik Mekatronika, Ibu Siti Aminah, S.T., M.T
4. Kedua Pembimbing proyek akhir Bapak Dr. Setyawan Ajie Sukarno, S.S.T., M.T., M.Sc.Eng dan Bapak Dr. Susetyo Bagas Bhaskoro, S.S.T., M.T.
5. Panitia proyek akhir Ibu Hilda Khoirunnisa.S.Tr.T., M.Sc.Eng, dan Bapak Afaf Fadhil Rifa’i, S.T., M.T.
6. Teristimewa kepada Orang Tua serta keluarga besar penulis yang selalu mendoakan, memberikan motivasi dan pengorbanannya baik dari segi moril, materi kepada penulis sehingga penulis dapat menyelesaikan proyek akhir ini.
7. Rekan kelompok 9 Fachri Maulana Fulchan, Riska Wulandari serta rekan kelas AEB 2019 dan rekan – rekan mahasiswa AE 2019 yang saling mendukung dan menyemangati dalam proses penyelesaian proyek akhir ini

Akhir kata penulis menyadari bahwa dalam penulisan proyek akhir ini masih jauh dari kesempurnaan. Karena itu, penulis memohon saran dan kritik yang sifatnya membangun demi kesempurnaannya dan semoga bermanfaat bagi kita semua. Aamiiiin Ya Robbal Alamin.

Bandung, 20 Juli 2022

Penulis

# DAFTAR ISI

[ABSTRAK i](#_Toc110324667)

[*ABSTRACT* ii](#_Toc110324668)

[KATA PENGANTAR iii](#_Toc110324669)

[DAFTAR ISI v](#_Toc110324670)

[DAFTAR GAMBAR vii](#_Toc110324671)

[DAFTAR TABEL ix](#_Toc110324672)

[BAB I 1](#_Toc110324673)

[PENDAHULUAN 1](#_Toc110324674)

[1.1 Tujuan 1](#_Toc110324675)

[1.2 Teknologi 1](#_Toc110324676)

[1.3 Pemilihan Komponen dan Bahan 2](#_Toc110324677)

[1.4 Batasan Pembahasan 4](#_Toc110324678)

[BAB II 5](#_Toc110324679)

[LAPORAN TEKNIK 5](#_Toc110324680)

[2.1 Landasan Teori 5](#_Toc110324681)

[2.1.1. Latar Belakang Proyek 5](#_Toc110324682)

[2.1.2. *Master Controller* 6](#_Toc110324683)

[2.1.3. Lengan Robot (*Arm Manipulator*) 7](#_Toc110324684)

[2.1.4. Kinematik 8](#_Toc110324685)

[2.1.5. *Arduino* Mega 2560 9](#_Toc110324686)

[2.1.6. Motor Stepper 10](#_Toc110324687)

[2.1.7. Potensiometer 12](#_Toc110324688)

[2.1.8. Komunikasi Serial UART Arduino 14](#_Toc110324689)

[2.2 Metodologi Penelitian 16](#_Toc110324690)

[2.3 Gambaran Umum Sistem 17](#_Toc110324691)

[2.4 Perangkat Mekanik 18](#_Toc110324692)

[2.4.1. Konstruksi Arm Manipulator 6 DoF dan *Master Controller* 18](#_Toc110324693)

[2.4.2. Ukuran dan Area Kerja *Arm Manipulator 6 DoF* 23](#_Toc110324694)

[2.4.3. Diagram Alir Sistem 25](#_Toc110324695)

[2.4.4. Diagram Alir Sistem Secara Umum 25](#_Toc110324696)

[2.5 Gambaran Sub Sistem 27](#_Toc110324697)

[2.5.1. Komunikasi Data dan Algoritma *Master controller* 28](#_Toc110324698)

[2.5.2. Mekanik *Master controller* 36](#_Toc110324699)

[2.5.3. Elektrik *Master controller* 38](#_Toc110324700)

[2.6 Hasil Pengujian 42](#_Toc110324701)

[2.6.1. Pengujian Algoritma 42](#_Toc110324702)

[2.6.2. Pengujian Respond 47](#_Toc110324703)

[*2.6.3.* Pengujian input pergerakkan *master controller* terhadap *joint arm manipulator* 48](#_Toc110324704)

[BAB III 52](#_Toc110324705)

[KESIMPULAN 52](#_Toc110324706)

[3.1 Kesimpulan 52](#_Toc110324707)

[3.2 Saran 52](#_Toc110324708)

[DAFTAR PUSTAKA 53](#_Toc110324709)

[LAMPIRAN 54](#_Toc110324710)

# DAFTAR GAMBAR

[**Gambar 2. 1** ROV dengan *arm manipulator* 6](#_Toc111736209)

[**Gambar 2. 2** *Master controller* 6](#_Toc111736210)

[**Gambar 2. 3** Lengan robot ROV 7](#_Toc111736211)

[**Gambar 2. 4** Arduino Mega 2560 9](#_Toc111736212)

[**Gambar 2. 5** Arduino Mega 2560 Pin-Out 10](#_Toc111736213)

[**Gambar 2. 6** Rangkaian motor *stepper* Unipolar 11](#_Toc111736214)

[**Gambar 2. 7** Rangkaian motor *stepper* Bipolar 12](#_Toc111736215)

[**Gambar 2. 8** Komponen Potensiometer 13](#_Toc111736216)

[**Gambar 2. 9** Ilustrasi wiring komunikasi serial antara Arduino 15](#_Toc111736217)

[**Gambar 2. 10** Alur sistem perancangan 16](#_Toc111736218)

[**Gambar 2. 11** Desain *Arm Manipulator* dan *Master* *controller* 17](#_Toc111736219)

[**Gambar 2. 12** Gambaran umum sistem 18](#_Toc111736220)

[**Gambar 2. 13** *Kinematics Diagram Arm Manipulator 6 DoF* 19](#_Toc111736221)

[**Gambar 2. 14** *Joint* Pada *Arm Manipulator* 20](#_Toc111736222)

[**Gambar 2. 15** Dimensi *Master controller* 21](#_Toc111736223)

[**Gambar 2. 16** Panel *Master controller* dan *Joystick Master controller* 22](#_Toc111736224)

[**Gambar 2. 17** Dimensi area kerja *arm manipulator* 6 DoF 23](#_Toc111736225)

[**Gambar 2. 18** Alur sistem secara umum 25](#_Toc111736226)

[**Gambar 2. 19** Alur sistem secara umum A 26](#_Toc111736227)

[**Gambar 2. 20** Tampilan hasil perancangan *master controller* 27](#_Toc111736228)

[**Gambar 2. 21** Gambaran wiring komunikasi data serial Arduino 28](#_Toc111736229)

[**Gambar 2. 22** Data pengiriman *Master controller* 29](#_Toc111736230)

[**Gambar 2. 23** Data yang diterima oleh slave arduino dan hasil parsing data 30](#_Toc111736231)

[**Gambar 2. 24** Konstruksi *Joystick master controller* 37](#_Toc111736232)

[**Gambar 2. 25** Konstruksi *Panel master controller* 38](#_Toc111736233)

[**Gambar 2. 26** Resistor PULL DOWN 39](#_Toc111736234)

[**Gambar 2. 27** Rangkaian *Motherboard master controller* 40](#_Toc111736235)

[**Gambar 2. 28** *Design* PCB *Motherboard* 40](#_Toc111736236)

[**Gambar 2. 29** *Keyboard Push button* 42](#_Toc111736237)

[**Gambar 2. 30** Tampilan awal mode Joystick 43](#_Toc111736238)

[**Gambar 2. 31** Tampilan mode Joystick FOLD to HOME position 43](#_Toc111736239)

[**Gambar 2. 32** Tampilan mode joystick saat tombol *start*ditekan 44](#_Toc111736240)

[**Gambar 2. 33** Tampilan mode Joystick saat tombol *Home* ditekan 45](#_Toc111736241)

[**Gambar 2. 34** Tampilan mode Joystick saat tombol *Freeze* ditekan 45](#_Toc111736242)

[**Gambar 2. 35** Tampilan awal mode Kartesian 46](#_Toc111736243)

[**Gambar 2. 36** Tampilan awal mode Neutral 46](#_Toc111736244)

[**Gambar 2. 37** Rumus mencari persentase *error* 49](#_Toc111736245)

[**Gambar 2. 38** Sampel pengambilan data 51](#_Toc111736246)

# DAFTAR TABEL

[**Tabel 2. 1** Perbandingan pemilihan sensor 2](#_Toc111736289)

[**Tabel 2. 2** Perbedaan Invers Kinematics dan Forward Kinematics 9](#_Toc111736290)

[**Tabel 2. 3** Spesifikasi Arduino Mega 2560 10](#_Toc111736291)

[**Tabel 2. 4** Perbandingan skala putaran terhadap nilai resistansi Potensiometer Linear 13](#_Toc111736292)

[**Tabel 2. 5** Perbandingan skala putaran terhadap nilai resistansi Potensiometer Logaritmis 14](#_Toc111736293)

[**Tabel 2. 6** Perbandingan skala putaran terhadap nilai resistansi Potensiometer Anti Logaritmis 14](#_Toc111736294)

[**Tabel 2. 7** Sudut maksimal dan minimal pergerakan antara *joint* 24](#_Toc111736295)

[**Tabel 2. 8** Tabel pengujian respons 47](#_Toc111736296)

[**Tabel 2. 9** Pengujian input pergerakkan *master controller* terhadap *joint* 1 49](#_Toc111736297)

[**Tabel 2. 10** Pengujian input pergerakkan *master controller* terhadap *joint* 2 49](#_Toc111736298)

[**Tabel 2. 11** Pengujian input pergerakkan *master controller* terhadap *joint* 3 50](#_Toc111736299)

[**Tabel 2. 12** Pengujian input pergerakkan *master controller* terhadap *joint* 4 50](#_Toc111736300)

[**Tabel 2. 13** Pengujian input pergerakkan *master controller* terhadap *joint* 5 50](#_Toc111736301)

[**Tabel 2. 14** Pengujian input pergerakkan *master controller* terhadap *joint* 6 50](#_Toc111736302)

[**Tabel 2. 15** Rata-rata *error* pada tiap *joint* 51](#_Toc111736303)

# BAB I

# PENDAHULUAN

## Tujuan

Tujuan dari penulisan karya tulis proyek akhir ini adalah sebagai berikut:

1. Memahami struktur dan pergerakan dari lengan robot 6 *degree of freedom*
2. Lengan robot mampu bekerja mengikuti inputan data yang dikirimkan oleh *master* *controller* pada mode pengendalian *Joystick.*
3. Membuat prototipe rancangan sistem *master* *controller*.
4. Membuat algoritma dan pengiriman data oleh *master* *controller* untuk menggerakkan lengan robot.
5. Informasi data pergerakan robot ditampilkan dalam *display* LCD.

## Teknologi

Teknologi yang digunakan para proyek akhir ini adalah:

1. Interfacing : LCD Display 20x4
2. Proses : *Microcontroller* – Arduino Mega 2560

Perangkat keras:

1. Motor Stepper : a. Nema 8 (8HS11-0204S)

b. Nema 11 (28HD1411-02)

c. Nema 17 (17HS4401)

d. Nema 23 (5718HB2401)

1. Driver Motor Stepper : a. A4988 Driver Stepper Module

b. TB6600 Driver Stepper Module

1. Input Digital : a. Push Button

b. Saklar Toggle

1. Output Digital : a. LED
2. Sensor : a. Potensiometer 100K Ohm

Perangkat Lunak:

1. Arduino IDE
2. Solidworks 2020
3. Eagle PCB Design

## Pemilihan Komponen dan Bahan

Pemilihan komponen yang dipakai, dipilih dengan pertimbangan-pertimbangan dan Batasan pembahan yang di tentukan Adapun pemilihan komponen pada proyek akhir ini adalah:

1. Potensiometer linear sebagai sensor

Untuk memilih suatu sensor yang sesuai digunakan pada tiap joint *master arm* harus mempunyai kriteria yang dibutuhkan yaitu:

1. Bagaimana resolusi yang dihasilkan komponen?
2. Berapa ukuran dimensi pada sensor?
3. Apakah komponen terdapat bebas di pasaran?
4. Berapa besaran sudut rotasi putaran pada sensor?

Untuk menjawabnya dibuatlah tabel perbandingan sensor Potensiometer dengan sensor Encoder rotary sebagai perbandingan yang dapat menjadi acuan pemilihin sensor. Keduanya dipilih dengan pertimbangan fungsi, harga, dan ketersediaan komponen di pasarsan.

**Tabel 2. 1** Perbandingan pemilihan sensor

|  |  |  |
| --- | --- | --- |
| Parameter | Potensiometer Linear | Modul Rotary Encoder |
| Resolusi | 0 – 1023 | 0 – 30 |
| Dimensi | 16.9 mm | 26 mm |
| Maks. sudut rotasi | 270° | 360° |
| Mudah ditemukan di pasaran | *Yes* | *Yes* |

Berdasarkan tabel di atas dapat disimpulkan bahwa:

1. Sensor potensiomer mempunyai resolusi yang lebih besar dibandingkan dengan sensor Encoder. Resolusi akan memperngaruhi pergerakkan lengan robot sehingga point ini menjadi acuan penting penggunaan potensiometer.
2. Pada ukuran dimensi sensor potensiometer dan sensor encoder terlihat bahwa sensor potensiometer memiliki dimensi yang lebih kecil di bandingkan dengan modul *encoder rotary*. Pada pembutan *master controller* dimensi yang kecil diperlukan agar proporsional dengan lengan operator.
3. Sensor potensiometer dan modul sensor *encoder rotary* umumnya mudah ditemukan di pasaran. Tetapi untuk *encoder rotary* dengan spesifikasi resolusi yang lebih tinggi sulit ditemukan di pasaran dengan harga yang relative mahal dan dimensi yang adapun lebih besar.
4. Sensor *encoder rotary* mempunyai keungulan khusus yaitu sudut rotasi putar yang mencapai 360° higga tanpa batas, dibandingankan dengan potensiometer linear yang hanya mencapai 270° saja.

Berdasarkan data diatas dan kebutuhan robot yang dibuat harus mampu untuk memberikkan pergerakkan yang baik dan lancar dengan sudut yang memang dibatasi agar penggunaan arm manipulator dapat terfokus pada suatu area kerjanya maka dipilihlah potensiometer linear sebagai sensor pembacaan untuk gerakan lengan robot proyek akhir ini, pembuatan *arm master* memaksimalkan sudut putaran pada sensor potensiometer linear.

1. *Arduino Mega* sebagai Controller

*Arduino Mega* digunakan karena memiliki spesifikasi tertinggi dibandingkan jenis *Arduino* lainnya. Dengan julmah I/O yang cukup untuk menampung *input* dan *output* yang diperlukan. Dan dengan *software* IDE Arduino mempermudah penggunannya karena *platform* ini menjadi banyak pilihan dan acuan bagi banyak praktisi.

1. Display LCD 20x4

Penggunaan LCD 20x4 dipakai karena dapat digunakan I2C sebagai komunikasi serial dua arah pada mikrokontroler Arduino. Penggunaan I2C ini dapat memudahkan dan meminimalisir penguunaan pin pada Arduino. Dengan ukuran LCD yang relative kecil dan dapat memuat informasi yang di perlukan oleh *master controller* serta pemprogramannya yang mudah membuat LCD 20x4 dipilih sebagai *display master controller*.

1. Pengiriman Data Serial UART

Pengiriman data serial UART (*Universal Asynchronous Receiver-Transmitter)* dengan pin *rx/tx* dipilih karena perngiriman data ini mudah untuk diaplikasikan. Pada kenyataannya untuk mengendalikan robot ROV diperlukan pengiriman data melalui sebuah kabel, hal itu dikarenakan untuk pengendalian robot di bawah laut mengharuskan komunikasi data yang stabil dengan jarak antara pengendali dengan robotnya yang saling berjauhan.[1]

1. Bahan

*Master controller* dibuat dengan menggunakan 3D printer Ender 3 V2, Adapun bahan yang digunakan adalah jenis bahan *Polylactic acid*. *Polylactic acid* (PLA) adalah bahan yang berpotensi menggantikan polimer tersebut. PLA merupakan salah satu biopolimer yang diproduksi dari fermentasi gula pada bahan yang dapat diperbaharui seperti tebu dan jagung. Oleh karena itu PLA sangat aman digunakan dan ramah lingkungan. Keunggulan PLA jika dibandingkan dengan biopolimer lainnya adalah memiliki kemampuan proses thermal yang lebih baik jika dibandingkan dengan bahan lain seperti *Polyhydroxyl alkanoate* (PHA), *Polyethylene glycol* (PEG), dan *Polycaprolactone* (PCL), memiliki kekuatan tarik tinggi, dan dapat diproses menggunakan peralatan.

## Batasan Penelitian

Fokus penelitian yang diuraikan pada karya tulis ini dibatasi. Pembatasan penelitian pada karya tulis ini adalah sebagai berikut:

1. Lengan robot menggunakan desain robot yang bersifat *open source* pada laman *SkyentificGit/SmallRobotArm - GitHub* .
2. *Master controller* dan *arm manipulator* (lengan robot) terhubung dengan komunikasi serial.
3. Robot digerakkan secara *Forward Kinematics* (masukan derajat menjadi keluaran posisi).
4. *Arm manipulator* digerakkan dengan menggunakan mode *Joystick Master* *controller.* Pada perancangannya Master controller dibuat dengan dua mode pergerakkan.
5. Robot tidak mempunyai fitur *teaching.*

# BAB II

# LAPORAN TEKNIK

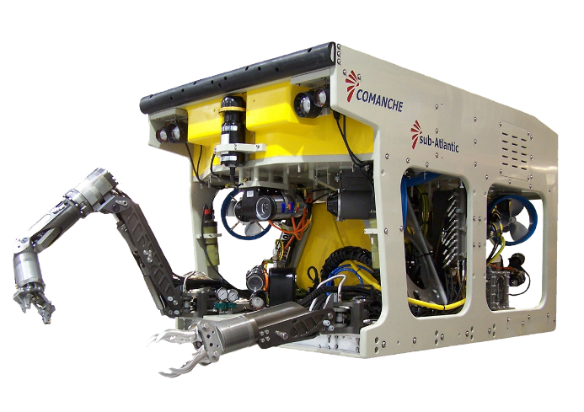
* 1. Landasan Teori

### Latar Belakang Proyek

Perkembangan teknologi robot mengalami suatu kemajuan yang pesat. Dengan kecangihan teknologi yang dimilikinya, maka akan semakin banyak pekerjaan-pekerjaan berat dan berisiko yang sebelumnya dilakukan oleh manusia menjadi dilakukan oleh robot. Pada sektor industri penggunaan robot telah terbukti meningkatkan konsistensi kualitas dan efisiensi pencegahan terjadinya kecelakaan kerja.

Untuk mengendalikan robot terdapat suatu program pengendali gerakan robot yang berfungsi mengendalikan seluruh aktuator yang dimilikinya dengan tujuan yang diciptakan sesuai dengan perancangannya. Menurut Robot Institute of America, robot adalah mesin multifungsi yang dapat diprogram ulang, dirancang untuk memindahkan material, komponen, alat, atau perangkat khusus, melalui gerakan terprogram yang variabel untuk pengerjaan berbagai tugas.[2] Salah satu tugas yang dapat menjadi bagian dari pencegahan terjadinya kecelakaan kerja adalah pengendalian robot jarak jauh pada kondisi medan yang tidak memungkinkan dikerjakan oleh manusia. Apabila suatu robot dapat diimplementasikan dan menggantikan manusia dalam melaksanakan tugas ini, maka efisiensi pencegahan terjadinya kecelakaan kerja dapat diminimalisir. Selain itu, robot juga diharapkan dapat dioperasikan dengan mudah dan sederhana.

Robot dapat diprogram untuk melakukan gerakan sesuai dengan referensi yang didapat dari sensor-sensor yang dimilikinya, Karenanya diperlukan suatu pengendali yang mampu untuk mengendalikan robot tersebut. *Master controller* merupakan suatu *pendant* pengendali robot yang digerakkan dengan mempergunakan tangan sesuai dengan gerakan yang diinginkan. Pada implementasi sesungguhnya *master controller* digunakan pada pengendalian *arm manipulator* pada robot bawah laut atau *Underwater-Remotly Operated Vehicle* (ROV).[3]



**Gambar 2. 1** ROV dengan *arm manipulator*

*Master controller* memiliki kurang lebih 7 sensor yang terpasang padanya yang masing-masing mengendalikan aktuator yang berada pada *joint arm manipulator.* Pada *master controller* pun terdapat banyak tombol yang diperuntukkan untuk berbagai fitur seperti *scalling* pergerakan*, start, stop,* pergerakan translasi, dan lain-lain.



**Gambar 2. 2** *Master controller*

Dalam rangka sedikit sumbangsih untuk kemajuan teknologi bawah laut Indonesia dan dengan tujuan pembelajaran bagi penulis serta generasi selanjutnya. Dirancang lah proyek akhir ini yang merupakan lengan robot 6 DoF yang dikendalikan menggunakan mode Joystick Dikontrol menggunakan dua buah *Arduino Mega* yang dikomunikasikan secara serial dengan tampilan *interface* berupa LCD untuk menampilkan posisi sudut dari lengan robot.

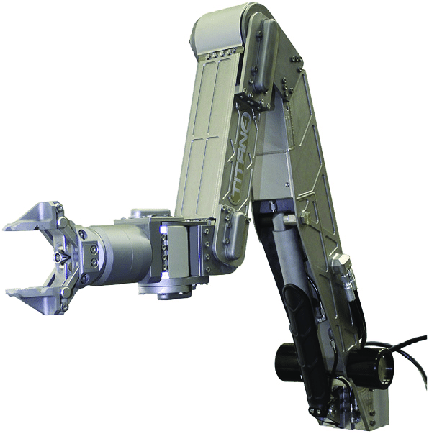
### *Master Controller*

*Master controller* merupakan replika tangan berukuran kecil yang difungsikan untuk menggerakan manipulator. Setiap *joint* dari manipulator atau DoF, digerakkan oleh satu sensor yang berada pada *master arm*. Dengan menggunakan controller ini, dapat membuat pergerakan dari manipulator menjadi simpel dan intuitif. Pergerakan dari masing – masing *joint* pun dapat di batasi sehingga manipulator tidak menjangkau area atau konfigurasi sistem yang dapat menimbulkan masalah. Fitur lain yang dimiliki controller ini adalah individual *joint* freeze, *joint* diagnostics, error checking, dan lain sebagainya.[3]

### Lengan Robot (*Arm Manipulator*)

*Arm Manipulator* adalah kontruksi robot yang memiliki bentuk hanya sebuah lengan saja. Robot ini memiliki aktuator berupa motor untuk menggerakan seluruh bagian robot tersebut. Robot manipulator ini sudah banyak dipakai di bidang industri terutama dalam bagian *packing* suatu pabrik. Selain untuk menghemat waktu robot ini digunakan karena hasil yang diperoleh juga lebih baik daripada hasil yang dikerjakan oleh manusia.

Lengan robot banyak digunakan di industri yang membutuhkan ketepatan tinggi dan pekerjaan yang repetitif. Namun dalam pengaplikasian di ROV (*Remotely Operated Vehicle*), lengan robot lebih ditekankan pada mobilitas nya dibandingkan ketepatannya dikarenakan lingkungan pengaplikasiannya yang dinamis. Untuk ketepatan, nantinya akan dipantau oleh operator yang mengoperasikan lengan robot.



**Gambar 2. 3** Lengan robot ROV

Beberapa istilah dalam arm manipulator yaitu:

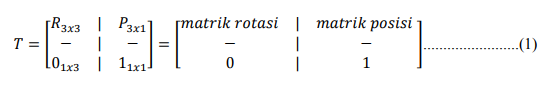
1. Joint (Sendi) yaitu koneksi antar *link* yang dapat menentukan pergerakan.
2. Link merupakan bagian-bagian kerangka yang kaku yang dihubungkan secara bersamaan sehinga membentuk suatu rangkaian *kinematic.*
3. Derajat kebebasan ( Degree Of Freedom atau DoF ) merupakan setiap Gerakan linier atau putaran sepanjang atau sekitar pada sebuah sumbu (axis).

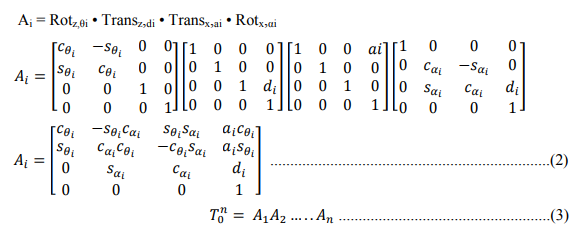
### Kinematik

Kinematik merupakan pembelajaran pergerakan tubuh robot tanpa memperhitungkan gaya, torsi, maupun momen tertentu yang menyebabkan pergerakan.

1. Forward Kinematik

Koordinat posisi dan orientasi dari *end effector* merupakan hal yang penting dalam pergerakan manipulator. Orientasi menyatakan rotasi *end effector* terhadap *base frame* sedangkan koordinat posisi menyatakan letak *end effector* (x,y,z) terhadap *base frame*. Ciri khas dari penyelesaian menggunakan persamaan *forward kinematics* adalah hanya mempunyai satu solusi penyelesaian yang menyatakan hubungan antara *joint* terkait atau *end effector* terhadap *base frame* sehingga dapat diketahui koordinat posisi dan orientasi dari *end effector*.

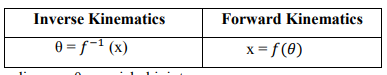


 Forward kinematic dinyatakan dalam matriks transformasi homogen seperti pada persamaan (1) dimana merupakan matriks rotasi dan merupakan matriks posisi yang menyatakn pergerakan *end effector* terhadap *base frame*. Langkah untuk mencari formard kinematic pertama kali adalah mencari terlebih dahulu parameter tiap *joint* dengan aturan Denavit Hartenberg (DH). Parameter *joint* ini kemudian dibentuk menjadi matriks transformasi homogen tiap *joint* seperti pada persamaan (2), untuk selanjutnya dibentuk menjadi matriks transformasi seperti pada persamaan (3) yang menyatakan hubungan antar *joint*.

1. Inverse Kinematik

Inverse kinematics digunakan untuk mencari kombinasi sudut yang harus diberikan pada *arm manipulator* agar *end effector* bergerak menuju koordinat posisi yang ditentukan. Hal ini berbeda dengan konsep *forward kinematics*, dimana sudut dari tiap *joint* sudah ditentukan untuk mencari koordina posisi dari *end effector*, perbedaan keduanya dapat dilihat pada Tabel 2.2. Ada beberapa cara untuk menyelesaikan *inverse kinematics*, salah satunya dalah dengan metode aljabar. Ciri khas dari penyelesaian menggunakan persamaan *inverse kinematics* adalah mempunyai lebih dari satu solusi penyelesaian, karena dimungkinkan memiliki lebih dari satu kombinasi sudut tiap *joiunt* untuk mendapatkan koordinat posisi dari *end effector*.

**Tabel 2. 2** Perbedaan Invers Kinematics dan Forward Kinematics

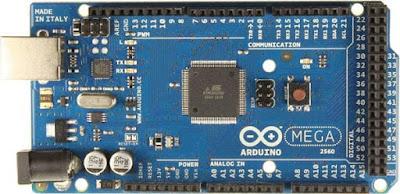


Dimana : θ = variabel *joint*

x = posisi dan orientasi end effector

### *Arduino* Mega 2560

Arduino Mega 2560 adalah salah satu dari sekian banyaknya jenis papan sirkuit yang dibuat oleh perusahaan resmi Arduino LLC. Papan sirkuit dengan *chip* mikrokontrolerAtmega2560 ini memiliki jumlah pin paling banyak diantara semua jenis Arduino lainnya. Arduino Mega cocok digunakan untuk membuat projek yang relative besar, dikarenakan kapasitas memori yang lebih besar dibandingkan Arduino jenis lain.[4]



**Gambar 2. 4** Arduino Mega 2560

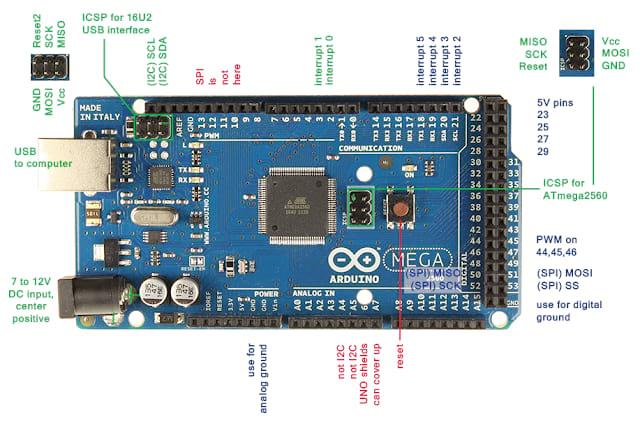
Pertimbangan untuk memilih Arduino Mega di dalam projek adalah sebagai berikut :

* Pin I/O yang cukup untuk mengintegrasikan komponen input dan output.
* Bahasa yang mudah diemengerti dan open source.
* Kapasitas memori Arduino yang cukup besar.

Berikut spesifikasi lanjut dari Arduino Mega 2560 :

**Tabel 2. 3** Spesifikasi Arduino Mega 2560

|  |  |
| --- | --- |
| Jenis Mikrokontroler | Atmega2560 |
| Tegangan Operasional | 5 Volt |
| Tegangan Rekomendasi | 7-12 Volt |
| Batas Tegangan | 6-20 Volt |
| Pin Input/Output Digital | 54 |
| Pin PWM | 15 |
| Pin Input Analog | 16 |
| Arus Untuk Pin Digital | 40 mA |
| Arus Untuk Pin 3,3 V | 50 mA |
| Memori Flash | 256 KB (8 KB untuk *bootloader*) |
| SRAM | 8 KB |
| EEPROM | 4 KB |
| Clock Speed | 16 MHz |
| Panjang | 10,1 cm |
| Lebar | 5,3 cm |
| Berat | 37 gram |



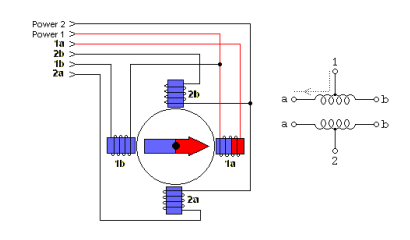
**Gambar 2. 5** Arduino Mega 2560 Pin-Out

### Motor Stepper

Motor *stepper* adalah perangkat elektromekanis yang bekerja dengan mengubah pulsa elektronis menjadi gerakan mekanis diskrit. Motor stepper merupakan motor listrik yang dikendalikan dengan pulsa-pulsa digital, bukan dengan memberikan tegangan yang terus-menerus. Motor stepper bergerak berdasarkan urutan pulsa yang diberikan kepada motor. Karena itu, untuk menggerakkannya diperlukan pengendali motor stepper yang membangkitkan pulsa-pulsa periodik.

.Untuk menggerakkan setiap step pada stepper ada beberapa metode yang dapat dilakukan, diantaranya yaitu full step dan half step. Metode full step berarti motor stepper berputar sesuai dengan spesifikasi derajat per stepnya, sedangkan metode half step berarti motor stepper berputar setengah derajat per step dari spesifikasi motor stepper tersebut.

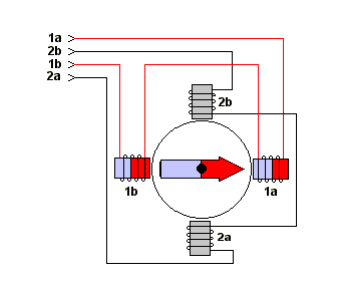
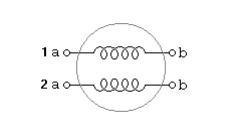
* + 1. **Motor Stepper Unipolar**

****

**Gambar 2. 6** Rangkaian motor *stepper* Unipolar

Motor stepper dengan tipe unipolar merupakan motor stepper yang memiliki 2 buah lilitan yang masing-masing lilitan di tengah-tengahnya diberikan sebuah tap. Rangkaian pengendali motor stepper unipolar lebih mudah dibuat karena hanya memerlukan satu signal On Off dengan menggunakan switch / transistor pada masing – maisng lilitannya. Motor ini mempunyai step sebesar 90 derajat dan mempunyai dua buah lilitan yang didistribusikan berseberangan 180 derajat di antara kutub pada stator. Sedangkan pada rotornya menggunakan satu pasang magnet permanen. Untuk menjalankan dan menghentikan motor ini cukup dengan menerapkan pulsa digital yang hanya terdiri atas tegangan positif dan nol (*ground*) pada salah satu terminal lilitan (*wound*) motor sementara terminal lainnya dicatu dengan tegangan positif konstan pada bagian tengah (*center tap*) dari lilitan.

1. **Motor Stepper Bipolar**



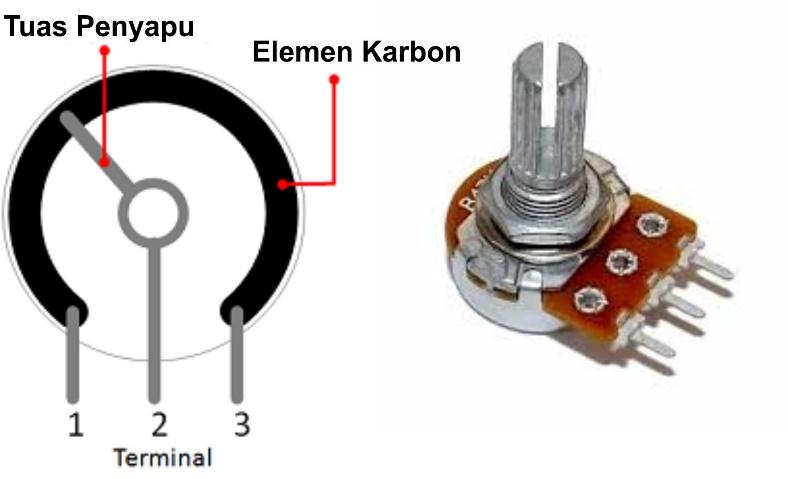
**Gambar 2. 7** Rangkaian motor *stepper* Bipolar

Bipolar Stepper motor hampir sama atau mirip dengan Unipolar stepper motor, namun pada motor stepper jenis bipolar tidak menggunakan sambungan center tap. Oleh karena itu Motor stepper jenis ini memerlukan jenis rangkaian kontrol yang berbeda. Pada motor stepper dengan lilitan bipolar, diperlukan sinyal pulsa yang berubah-ubah dari positif ke negative dan sebaliknya.Jadi pada setiap terminal lilitan (A&B) harus dihubungkan dengan sinyal yang mengayun dari positif ke negatif dan sebaliknya.Motor stepper bipolar memiliki keungulan dibandingkan dengan motor stepper unipolar dalam hal torsi yang lebih besar untuk ukuran yang sama.

### Potensiometer

Potensiometer adalah salah satu jenis resistor variable yang nilai resistansinya dapat diubah melalui tindakan mekanis dari penggunanya. Tindakan mekanis tersebut bisa berupa diputar atau digeser, baik itu diputar dengan tangan atau menggunakan alat berupa obeng trimmer.[5]

Potensiometer memiliki 3 kaki, dimana 2 kakinya terhubung ke sebuah lempengan karbon yang dijadikan elemen utama untuk mengubah nilai hambatan. Selanjutnya 1 kakinya yang lain menempel pada elemen karbon dan terhubung ke sebuah tuas penyapu yang bisa digerakkan melalui putaran atau pergeseran tadi.



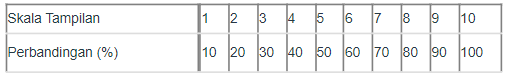
**Gambar 2. 8** Komponen Potensiometer

Dilihat dari perbandingan putaran tuas terhadap perubahan nilai, sebuah potensiometer dibedakan menjadi beberapa jenis yakni potensiometer linear, logaritmis dan anti logaritmis.

1. Potensiometer Linear

Potensiometer linear adalah potensiometer dengan perbandingan putaran terhadap resistansinya yang bergerak linear. Artinya perbandingan skala putaran terhadap nilai resistansi bergerak secara selaras dan sesuai. Rotasi tunggal pada potensiometer jenis ini sekitar 270 derajat atau ¾ putaran penuh. Untuk lebih jelas perhatikan table berikut ini :

**Tabel 2. 4** Perbandingan skala putaran terhadap nilai resistansi Potensiometer Linear

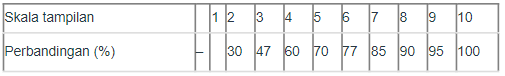


1. Potensiometer Logaritmis

Potensiometer logaritmis adalah potensiometer dengan perbandingan putaran terhadap resistansi yang bergerak mengikuti aturan logaritma, Potensiometer jenis ini banyak dipakai untuk keperluan audio seperti volume dan pengatur nada bass, middle, dan treble.

Skala yang di log kan adalah 0 sampai 10. Misalnya pada skala 2 akan diperoleh perbandingan sebesar log perbandingan sebesari logaritma dari 2 yaitu kira-kira sebesar 30%. Jika dibuat grafik akan terlihat grafik potensiometer logaritmis berupa garis lengkung dengan parabola menghadap keatas.

**Tabel 2. 5** Perbandingan skala putaran terhadap nilai resistansi Potensiometer Logaritmis

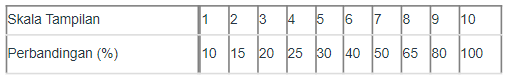


1. Potensiometer Anti Logaritmis (Reverse Log)

Potensiometer anti logaritmis adalah potensiometer dengan perbandingan putaran terhadap resistansi yang bergerak mengikuto aturan anti logaritma skala. Potensiometer ini banyak dipakai pada aplikasi penguat gitar.

Skala yang di antilog kan adalah 0 sampai 1 atau skala tampil dibagi 10. Misalnya pada skala 2 akan diperoleh perbandingan sebesar anti logaritma dari 0.2 yaitu kira-kira sebesar 10%. Jika dibuat grafik akan terlihat grafik potensiometer logaritmis berupa garis lengkung dengan parabola menghadap ke bawah.

**Tabel 2. 6** Perbandingan skala putaran terhadap nilai resistansi Potensiometer Anti Logaritmis

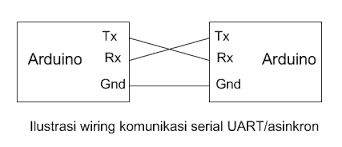


Dalam rancang bangun, *Arm manipulator* dengan *master controller* ini digunakan sensor Potensiometer jenis linear yang terpasang di masing-masing *joint master controller*. Pertimbangan untuk memakai Potensiometer linear adalah pergerakan robot nantinya harus linear dengan fluktuasi resistansi yang nantinya diberikan oleh potensiometer

### Komunikasi Serial UART Arduino

Secara sederhana, komunikasi serial adalah sebuah komunikasi yang terjadi dengan mengirimkan data per-bit secara berurutan dan bergantian. Dengan adanya komunikasi serial, maka Arduino bisa melakukan pembagian tugas dengan Arduino lainnya untuk mengendalikan sistem agar pengolahan data menjadi ringan.

Umumnya semua *board* Arduino telah memiliki minimal 1 *port* serial yang biasa dikenal dengan jenis USART atau UART (*Universal Asynchronous Receiver-Transmitter)*. Proses komunikasi serial menggunakan dua pin, yaitu pin RX untuk menerima data dan pin TX untuk mengirimkan data. Itulah alasan mengapa terkadang ada yang menyebut istilah komunikasi RX TX Arduino. Biasanya pin RX di Arduino adalah pin 0 dan pin TX adalah pin 1.

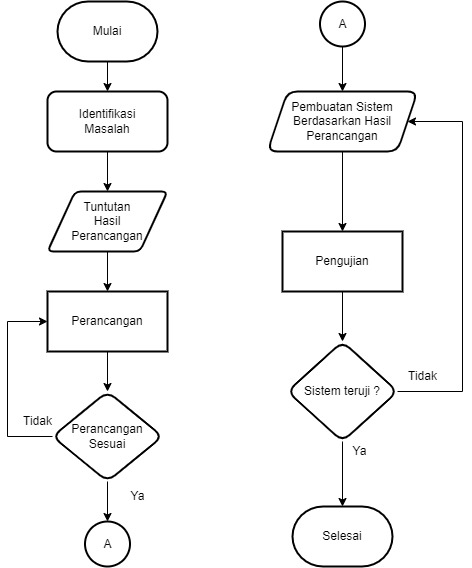


**Gambar 2. 9** Ilustrasi wiring komunikasi serial antara Arduino

Level elektrikal pada tegangan Arduino menggunakan tegangan digital dengan logika 0V untuk logika “0” dan 5V untuk logika “1” atau sering pula disebut level TTL ( Transistor-transistor Logic). Jika *device* pairing komunikasi serial berada pada level yang sama, maka antar pin serial serial dapat langsung di hubungkan. Sedangkan jika *device* pairing berada pada level serial yang berbeda, maka harus menggunakan rangkaian tambahan untuk menyesuaikan level tegangan agar kedua device dapat berkomunikasi.

Pengiriman data pada komunikasi serial ditentukan oleh *baudrate*. *Baudrate* adalah kecepatan transfer data dalam bit per satuan detik atau *bit per second (bps)*. Beberapa pilihan *baudrate* pada Arduino yaitu 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800 38400, 57600 dan 115200.[6]

* 1. Metodologi Penelitian

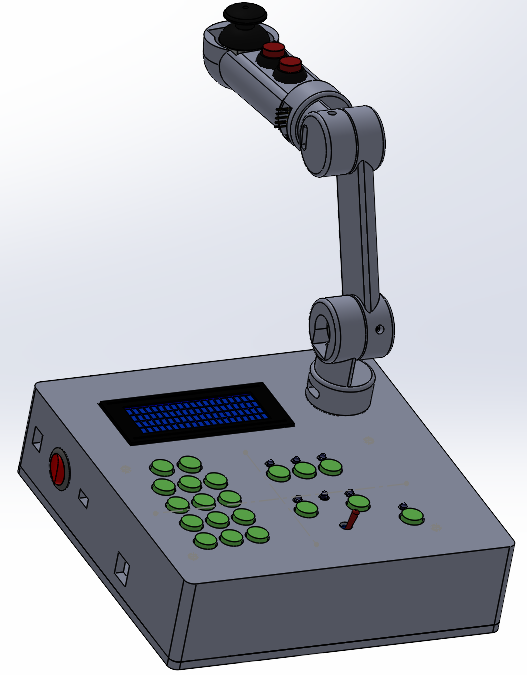
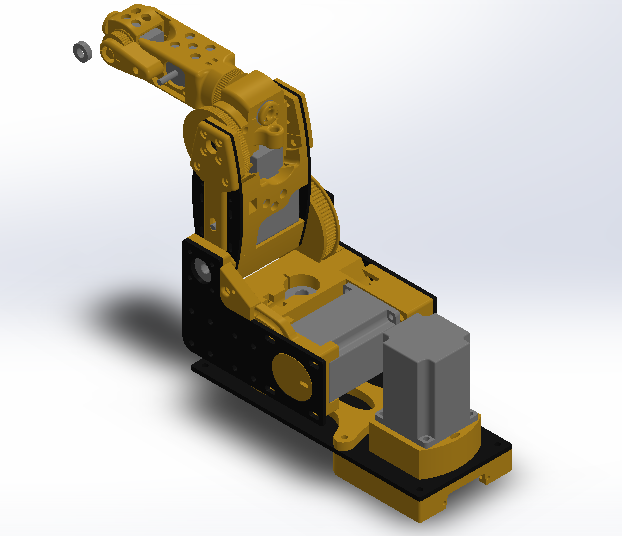


**Gambar 2. 10** Alur sistem perancangan

Gambar 2.10 di atas menggambarkan diagram alur sistem perancangan. Sebelum membuat setiap bagian pada robot, perlu dilakukan perencanaan pembuatan. Identifikasi Masalah meliputi fungsi keseluruhan robot, kemudian membuat daftar tuntunan yang harus dicapai lalu membuat rancangan sistem. Jika perancangan serta simulasi sudah dapat dipastikan, robot pun akan mulai ke tahap pembuatan. Perencanaan bertujuan untuk memudahkan langkah pembuatan robot ini.

* 1. Gambaran Umum Sistem

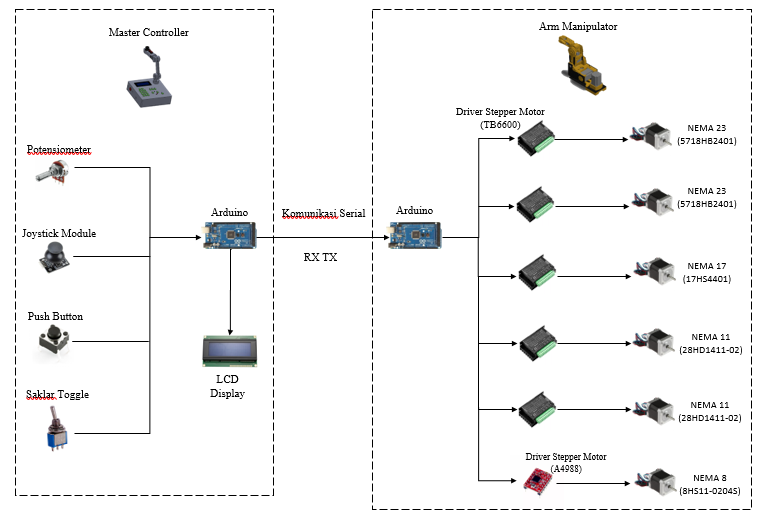
Secara hardware, sistem proyek akhir ini terdiri dari *arm manipulator* dan *master controller*. *Arm manipulator* merupakan lengan robot yang memiliki 6 axis pergerakan (*joint*) dan 1 gripper, sistem arm manipulator ini sering disebut dengan arm manipulator 7 functions. *Arm manipulator* akan dikendalikan oleh sebuah pengendali dengen mode dan fitur fitur yang dapat dipakai pada penggunaannya. *Master controller* adalah sebuah pendant yang mengendalikan pergerakan dari *arm manipulator*. Sistem ini dirancang agar *master controller* dapat mengendalikan *arm manipulator* dari pergerakkan *joint* yang dihasilkan dari pembacaan *joystick* yang terpasang pada *master controller*. Setiap *joint* pada *manipulator* akan digerakkan oleh masing – masing satu sensor yang terpasang pada *master controller.*

****

**Gambar 2. 11** Desain *Arm Manipulator* dan *Master* *controller*

Secara teknis lengan robot dikendalikan oleh *master* *controller* dan data hasil pembacaan berupa besaran sudut akan ditampilkan pada display LCD sekaligus mengirimkan data secara serial kepada Slave Controller Arduino mega untuk pemprosessan data selanjutnya.

*User* menggerakkan lengan robot secara manual dengan menggerakkan joystick yang terdapat pada *master controller*. Kemudian pergerakkan yang terbaca oleh potensiometer berupa data diolah oleh program dan ditampilkan kedalam bentuk sudut.



**Gambar 2. 12** Gambaran umum sistem

Terlihat pada Gambar 2.12 diatas master controller dibangun atas berbagai input komponen dan sensor, adapun komponen input terdiri dari Potensiometer, modul Joystick, Push button, dan saklar Toggle. Data yang dihasilkan oleh komponen input tersebut kemudian akan di proses oleh Arduino (master) sebagai controller. Lalu data akan di tampilkan melalui LCD sekaligus dikirim melalui komunikasi serial UART.

Data yang dikirim berupa sudut akan diterima oleh Arduino (*slave*) dan akan diubah menjadi pulsa digital untuk kemudian driver motor akan menggerakkan tiap-tiap aktuator motor stepper yang berada pada setiap joint *arm manipulator/*lengan robot.

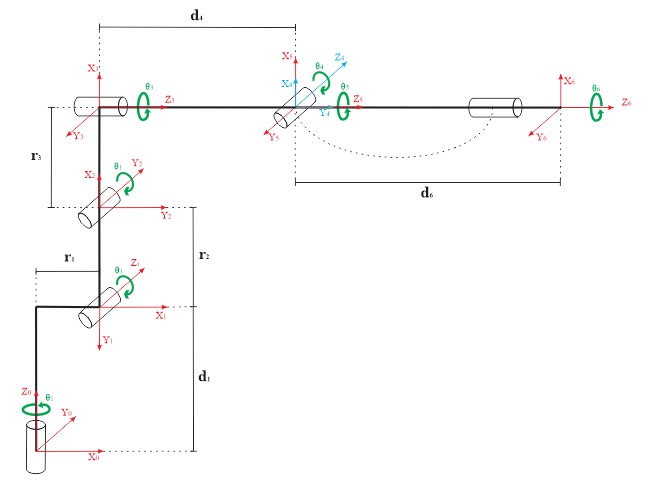
* 1. Perangkat Mekanik

1. Konstruksi *Arm Manipulator* 6 DoF dan *Master Controller*

*Arm manipulator* dan *Master controller* ini terbangun dari sistem kontrol, perangkat elektronik, dan perangkat mekanik yang masing-masing entitas tersebut saling berkesinambungan menjalankan fungsi setiap bagiannya. Sebagian besar konstruksi rangka dari *arm manipulator* dan *master controller* di manufaktur oleh 3D printer dengan bahan PLA. Berikut perangkat – perangkat mekanik pada alat ini:

1. *Arm Manipulator*

*Manipulator* terdiri dari *links* dan *joints*. Rangka dari *manipulator* 6 DoF ini dapat tergambarkan pada kinematic diagram seperti ini:



**Gambar 2. 13** *Kinematics Diagram Arm Manipulator 6 DoF*

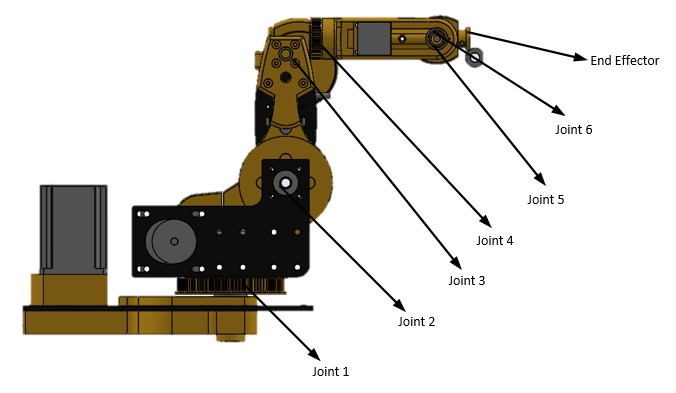
Panjang link:

r1 = 47 mm d1 = 133 mm

r2 = 110 mm d4 = 117.5 mm

r3 = 26 mm d6 = 28 mm

Motor stepper pun dipasang pada masing – masing *joint* sebagai sumber pergerakan. Berikut merupakan spesifikasi dari masing – masing motor stepper disetiap *joint*:



**Gambar 2. 14** *Joint* Pada *Arm Manipulator*

*Joint* 1 dan 2

Jenis : Nema 23 5718HB2401 bipolar stepper motor

Dimensi : 54x54x55 mm

*Shaft diameter* : 6.35 mm

*Step angle* : 1.8o

*Holding torque* : 1.3 N.m

*Phase current* : 3.5 A

*Joint* 3

Jenis : Nema 17 17HS4401 Bipolar stepper motor

Dimensi : 42x42x40 mm

*Shaft diameter / length* : 5 mm x 22 mm

*Step angle* : 1.8o

*Holding torque* : 4 Kg.cm

*Phase current* : 1.7A

*Joint* 4 dan 5

Jenis : Nema 11 28HD1411-02

Dimensi : 28x28x28 mm

*Shaft diameter / length* : 5 mm x 18 mm

*Step angle* : 1.8o

*Holding torque* : 6.5 N.cm

*Phase current* : 0.8 A

*Joint* 6

Jenis : Nema 8 8HS11-0204S

Dimensi : 20x20x28 mm

*Shaft diameter / length* : 4 mm x 10 mm

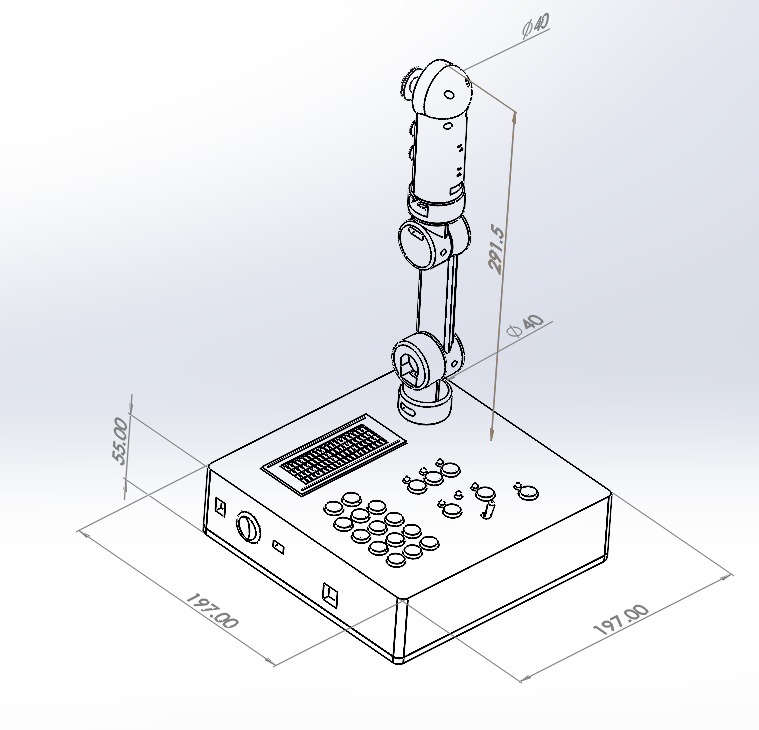
*Step angle* : 1.8o

*Holding torque* : 1.6 N.cm

*Phase current* : 0.2 A

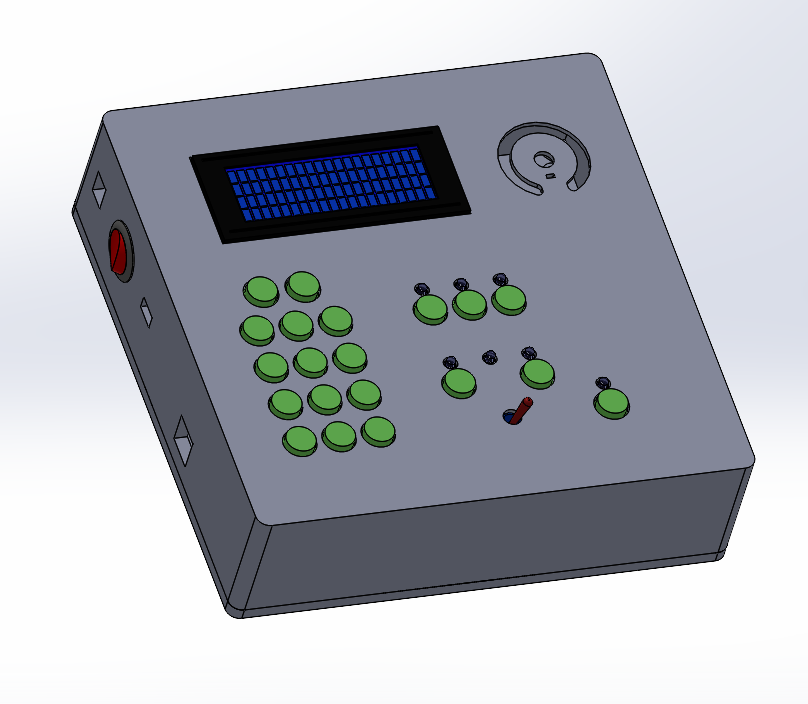
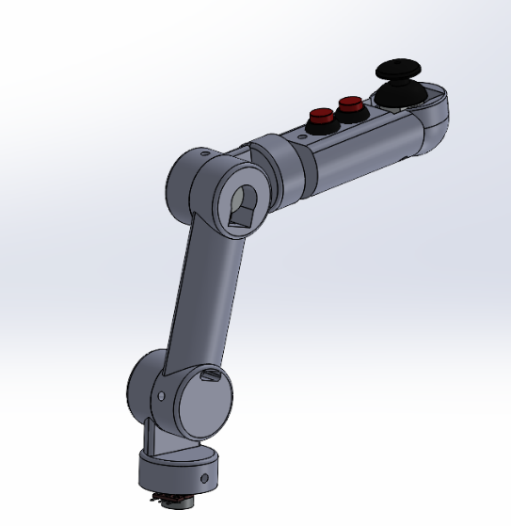
1. *Master controller*

*Master controller* memiliki dimensi yang di rancang agar *master arm* dapat dioperasikan dengan memaksimalkan sudut perputaran pada potensiometer. Adapaun dimensi pada *master controller* adalah seperti pada gambar ini:

**

**Gambar 2. 15** Dimensi *Master controller*

*Master controller* terdiri dari dua bagian, yaitu bagian *Panel Master* yang berfungsi sebagai *casing* *panel*, penempatan *display* informasi pergerakan *arm manipulator,* dan penempatan keyboard push button. Kemudian bagian selanjutnya yaitu *Joystick* *Master* yang berfungsi sebagai pengendali lengan robot pada mode Joystick.

**Gambar 2. 16** Panel *Master controller* dan *Joystick Master controller*

*Panel Master controller*

Dimensi : 197x197x55 mm

Thickness : 5 mm

*Material* : PLA ( *Polilactic acid* )

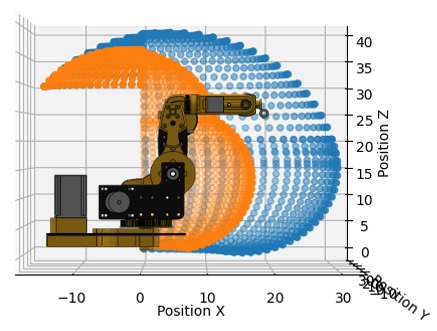
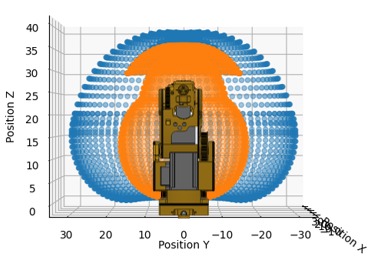
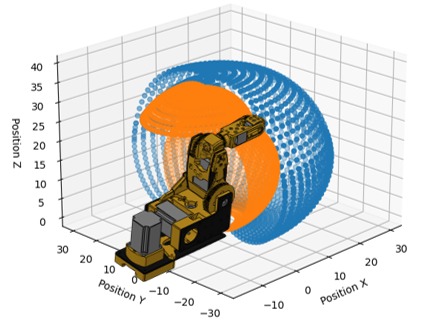
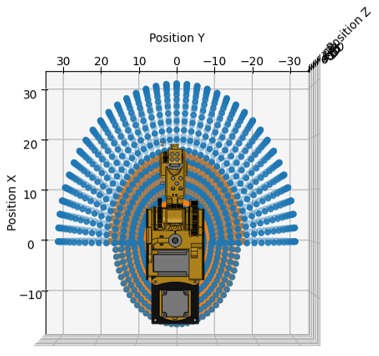
*Joystick Master* *controller*

Diameter : 40 mm

*Total length of joint* : 300 mm

*Material* : PLA ( *Polilactic acid* )

1. Ukuran dan Area Kerja *Arm Manipulator 6 DoF*

**Gambar 2. 17** Dimensi area kerja *arm manipulator* 6 DoF

Gambar 2.17 menjelaskan area kerja dari arm manipulator. Area kerja dari manipulator berbentuk seperempat bola dengan area yang berwarna biru merupakan capaian maksimal manipulator sedangkan area yang berwarna jingga merupakan capaian minimum nya.

Pergerakkan putaran sudut antar *joint* pada *arm manipulator* dibatasi dengan sudut maksimal dan minimal robot, ini bertujuan agar pergerakkan robot lebih terfokus pada area kerja yang ditentukan dan menghindari *arm manipulator* terjadi *crash.* Adapun sudut pergerakan antara *joint* seperti pada table dibawah ini:

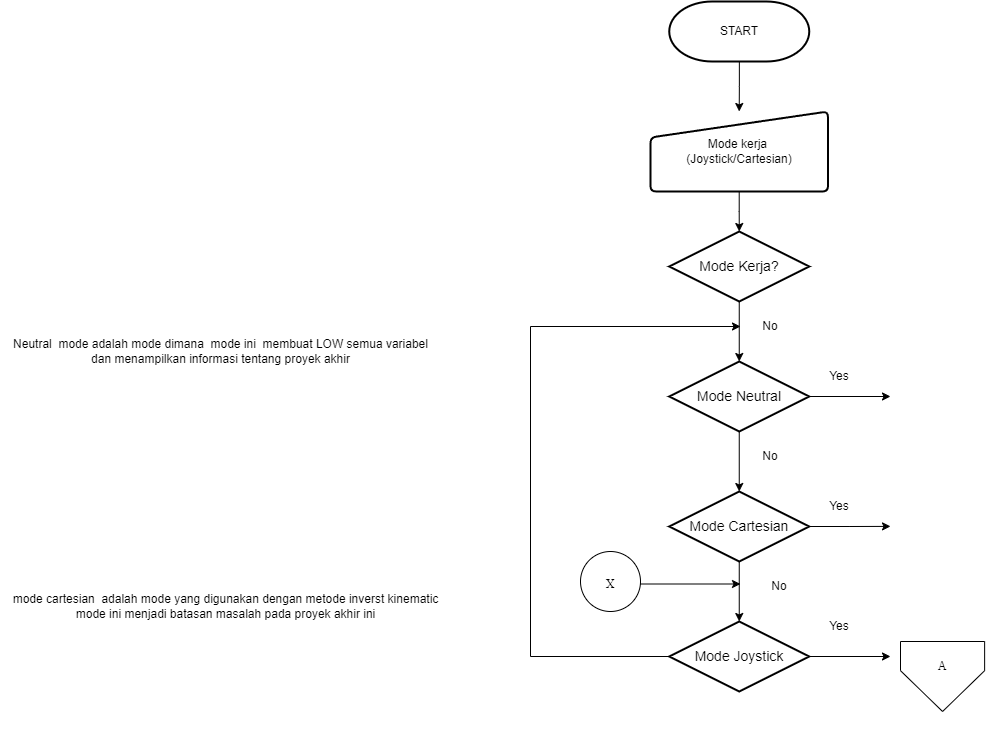
**Tabel 2. 7** Sudut maksimal dan minimal pergerakan antara *joint*

|  |  |
| --- | --- |
|  | *Joint* 1  Sudut Maksimal : 90°  Sudut Minimal : -90°  Jangkauan sudut : 90o ≤ θ1 ≤ -90o |
|  | *Joint* 2  Sudut Maksimal : 90°  Sudut Minimal : -45°  Jangkauan sudut : -45o ≤ θ2 ≤ 90o |
|  | *Joint* 3  Sudut Maksimal : 45°  Sudut Minimal : -90°  Jangkauan sudut : -90o ≤ θ3 ≤ 45o |
|  | *Joint* 4  Sudut Maksimal : 180°  Sudut Minimal :-180°  Jangkauan sudut : -180 o ≤ θ4 ≤ 180o |
|  | *Joint* 5  Sudut Maksimal : 90°  Sudut Minimal : -90°  Jangkauan sudut : -90 o ≤ θ5 ≤ 90 o |
|  | *Joint* 6  Sudut Maksimal :inf°  Sudut Minimal : -inf°  Jangkauan sudut : -inf o ≤ θ6 ≤ inf o |

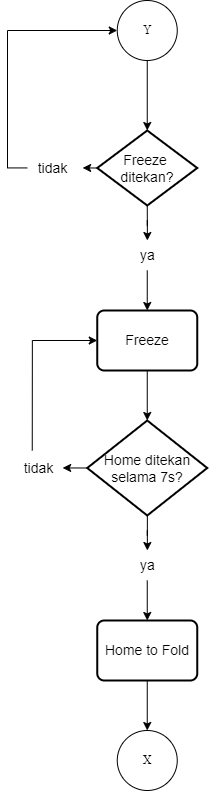
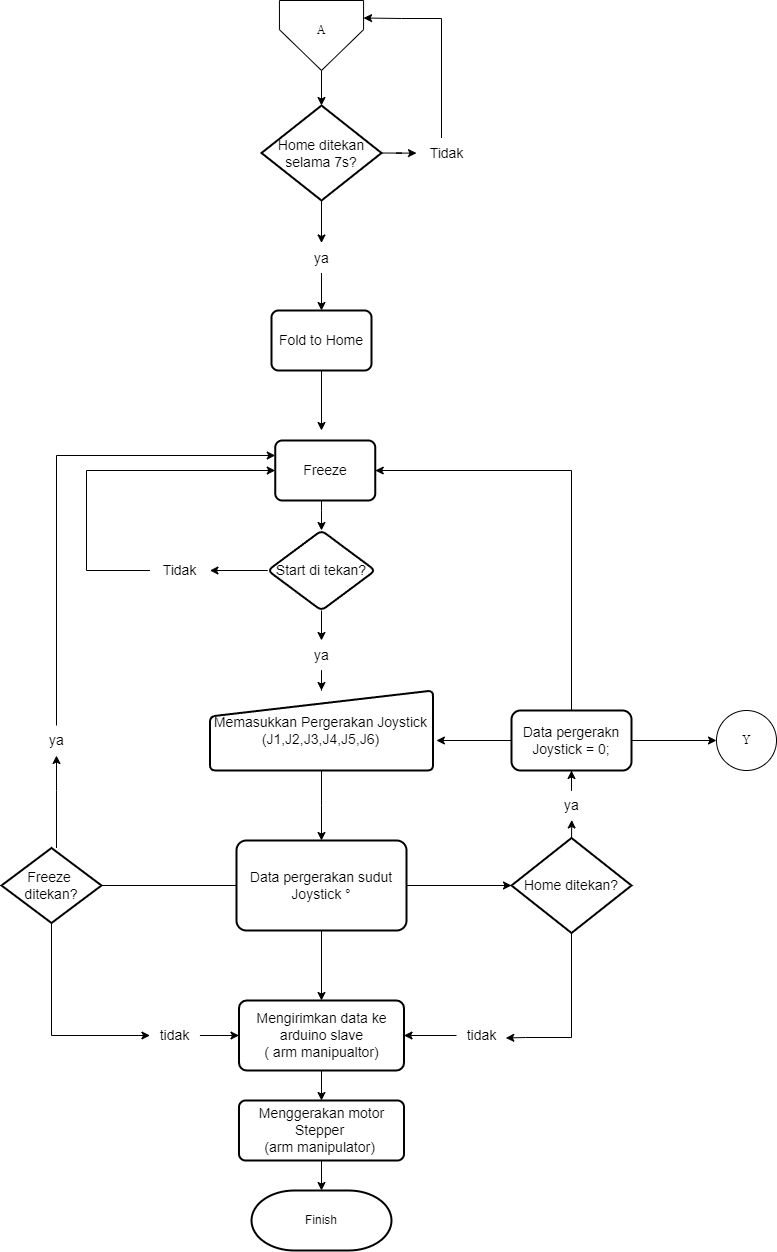
1. Diagram Alir Sistem

Untuk mulai menjalankan sebuah *Master controller, User* terlebih dahulu memilih mode pergerakkan yang akan dipakai. Terdapat dua mode pergerakkan yaitu mode Cartesian dan mode Joystick. Mode Neutral merupakan mode tambahan dimana pada mode ini semua variable menjadi berlogika LOW. Pada diagram alir sistem di bawah ini akan menjelaskan bagaimana mode Joystick dapat bekerja untuk mengendalikan suatu *arm manipulator.*

1. Diagram Alir Sistem Secara Umum



**Gambar 2. 18** Alur sistem secara umum



**Gambar 2. 19** Alur sistem secara umum A

Diagram pada Gambar 2.18 dan Gambar 2.19 menjelaskan bagaimana sistem pengendalian manipulator oleh master controller pada mode joystick. Saklar toggle harus diatur untuk memilih mode joystick. Manipulator mula-mula berada pada kondisi fold atau melipat. Ketika tombol home ditekan selama 7 detik, maka lengan robot akan bergerak ke posisi steady/home (keadaan dimana robot tergambar pada kinematic diagram). Untuk menggerakan manipulator tombol start perlu ditekan, sehingga master arm akan menghasilkan data derajat sesuai dengan masukkan dari operator. Data derajat masing – masing joint dan posisi *end-effector* dari robot tertampil pada display LCD yang tersedia. Jika tombol freeze ditekan, master arm tidak akan menghasilkan data derajat dan tidak akan menggerakan lengan robot bagaimanapun master arm digerakkan sedemikian rupa oleh operator. Robot akan bergerak ke posisi home atau posisi ketika semua joint bernilai 0 adalah ketika tombol home ditekan. Pada saat semua joint bernilai 0 dan tombol home ditekan selama 7 detik maka manipulator akan bergerak ke posisi melipat / fold.

* 1. Gambaran Sub Sistem

Pada bagian ini akan dijelaskan perancangan sistem mekanik, elektrik dan algoritma serta program pengiriman data yang terjadi pada komunikasi serial *master controller* dengan *slave* Arduino. Adapun bentuk hasil dari pengerjaan *master controller* dapat dilihat pada gambar di bawah ini:



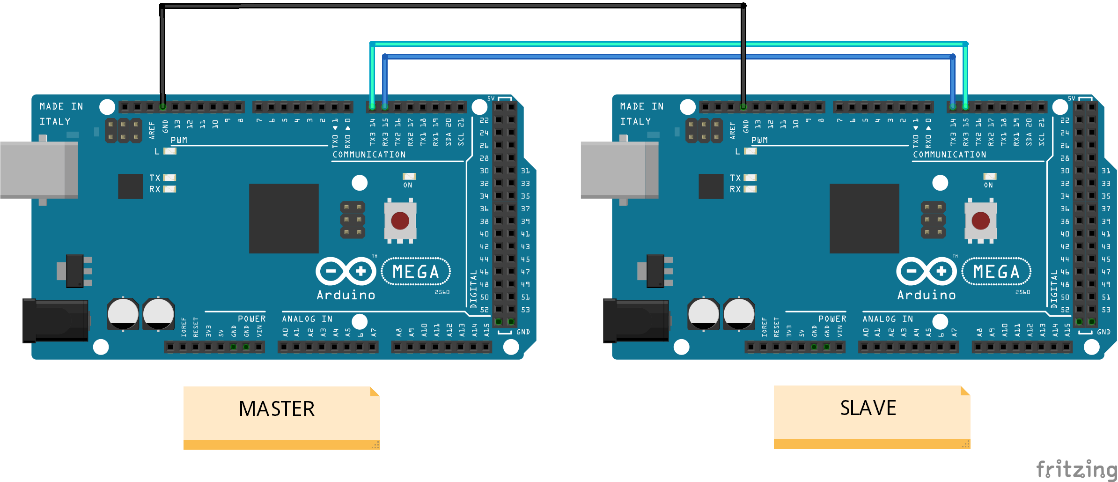
**Gambar 2. 20** Tampilan hasil perancangan *master controller*

Gambar 2.20 menunjukkan sebuah hasil rancangan prototype *Master controller* yang dibuat berdasarkan kebutuhan, fungsi, dan *study literatur* pada sebuah alat pengendali lengan robot pada ROV. Dengan kendala-kendala yang masih dapat ditemukan pada proses akhir pembuatan *master controller.*

1. Komunikasi Data dan Algoritma *Master controller*

Pada bagian ini penulis akan memaparkan proses penyelesaian subsistem Pengiriman data dan algoritma program mode Joystick pada *master controller*.

Komunikasi data dilakukan dengan menguhungkan pin tx *master* Arduino ke pin rx *slave* Arduino dan pin rx *master* Arduino ke pin tx *slave* Arduino. Adapun gambarannya seperti pada gambar di bawah ini:



**Gambar 2. 21** Gambaran wiring komunikasi data serial Arduino

1. Program untuk mengirimkan data ke slave Arduino pada *Master controller*

// Send Data to Slave

if (timeAfter - timeBefore >= 1000) {

// Assemble data dan kirim

String out = "#" + String(kode1) + "#" + String(steppJ1) + "#" + String(steppJ2) + "#" + String(steppJ3) + "#" + String(steppJ4) + "#" + String(yawJ5) + "#" + String(rollJ6) + "#" + String(gripCount) + "#" + String(fold) + "#$";

**Serial**.println (out);

**Serial3**.print (out);

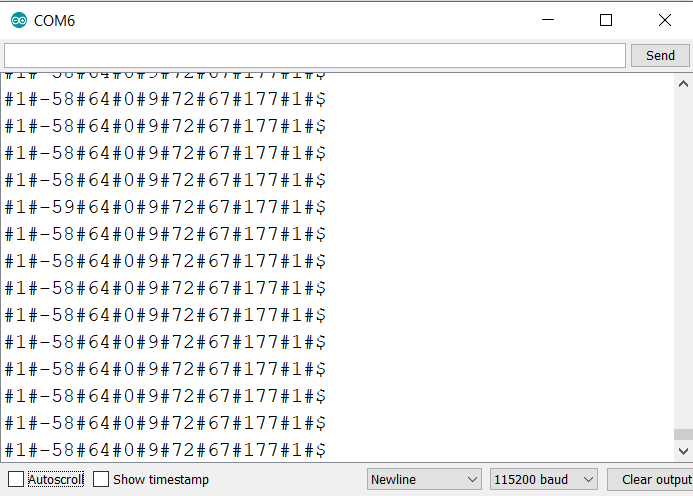
timeBefore = millis();

//Indikator pengiriman data (TX)

SendLEDIndicator();

}

Kemudian data yang terkirim akan berupa type data string yang di batasi oleh ‘#’ sebagai pembatas data yang masuk, dan diakhiri oleh ‘$’ sebagai indikator bahwa satu pembacaan penuh atau *looping* sudah terkirim



**Gambar 2. 22** Data pengiriman *Master controller*

1. Program untuk penerima data dan parsing slave Arduino

// Inisiasi variable komdat

String sData, data[12];

int dataInt[12];

bool parsing = false;

data yang dikirim oleh *master controller* sebanyak 9 data, tetapi pada penerima data di sediakan 12 data masukkan yang bertujuan untuk penambahan data. Kemudian data di olah dengan program :

void receiveEvent() {

// ===== KOMDAT

// Baca dari serial (ASCII), convert ke char, masukin ke data

while (**Serial3**.available() > 0) {

delay (10);

char c = **Serial3**.read();

sData += c;

if (c == '$') {

parsing = true;

}

}

//parsing data

if (parsing) {

// Serial.print ("DATA MASUK:");

// Serial.println(sData);

int q = 0;

for (int i = 0; i < sData.length(); i++) {

if (sData[i] == '#') {

q++;

data[q] = "";

}

else {

data[q] += sData[i];

}

}

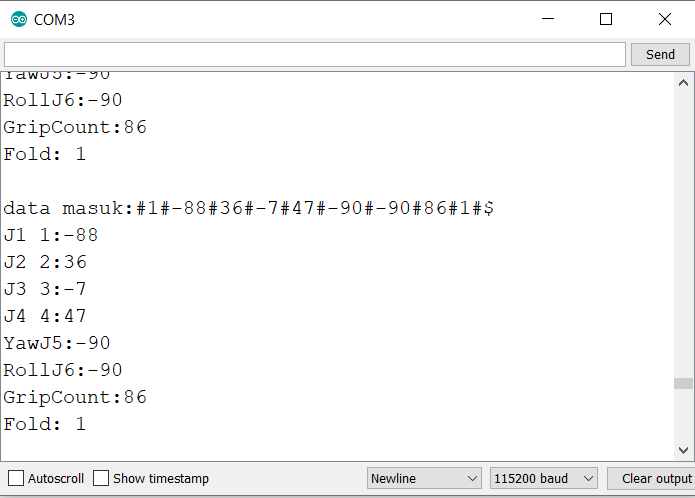
// Konversi string ke integer

for (int i = 0; i < 12; i++) {

dataInt[i] = data[i].toInt();

}

Pengolahan data yang di terima dilakukan dengan metode *parsing* dimana metode ini akan memisahkan data sesuai dengan kegunaan data tersebut. Data yang hasil pemisahan kemudian diubah ke dalam type data Integer untuk menggerkkan motor stepper.Program ini dibuat pada sebuah function dengan nama void *receiveEvent().*



**Gambar 2. 23** Data yang diterima oleh slave arduino dan hasil parsing data

1. Algoritma Pengiriman data pada Master controller

Perancangan *master controller* dibuat dengan beberapa mode *control,* salah satunya adalah mode Joystick. Pada mode Joystick sensor potensiometer dipakai sebagai pembacaan data untuk *Joint* 1, *Joint* 2, *Joint* 3,dan *Joint* 4. Untuk *Joint* 5 dan *Joint* 6, Pembacaan data diambil dari pergerakan axis X dan Y pada modul Joystick. Berikut adalah program pada mode Joystick:

/\* ====== JOYSTICK MODE ======= \*/

else if (modePergerakan == 1) {

// Ketika di posisi FOLD

if (fold == 0) {

// Determine Fold or Home Position

fold = determineFoldOrHome (pinButtonHome, fold);

LCDFolding();

startFreezeIndicator (0); // Red LED is On

// Send Data to Slave (Untuk pengiriman data fold)

if (timeAfter - timeBefore >= 1000) {

// Assemble data dan kirim

String out = "#" + String(kode1) + "#" + String(steppJ1) + "#" + String(steppJ2) + "#" + String(steppJ3) + "#" + String(steppJ4) + "#" + String(yawJ5) + "#" + String(rollJ6) + "#" + String(gripCount) + "#" + String(fold) + "#$";

**Serial**.println (out);

**Serial3**.print (out);

timeBefore = millis();

}

}

// Ketika sudah berada di posisi HOME

else if (fold == 1) {

// Start Mode or Freeze Mode

mode = determineModeStartFreeze(pinStart, pinFreeze, mode);

startFreezeIndicator(mode);

// Generate current state dari potensio

currentStatePot1 = generateStatePot(pinPot1, minAngleJ1, maxAngleJ1);

currentStatePot2 = generateStatePot(pinPot2, minAngleJ2, maxAngleJ2);

currentStatePot3 = generateStatePot(pinPot3, minAngleJ3, maxAngleJ3);

currentStatePot4 = generateStatePot(pinPot4, minAngleJ4, maxAngleJ4);;

// Read Joystick Yaw & Roll

joyYaw = readJoystick(pinJoyYaw);

joyRoll = readJoystick(pinJoyRoll);

Terlihat pada program untuk memulai pembacaan sensor-sensor yang terdapat pada *Joystick* *master controller,* terdapat kondisi FOLD = 1 dimana kondisi ini adalah kondisi untuk membuat arm manipulator berada pada posisi HOME, dibuat dengan Function bernama *determineFoldOrHome ()*

// ==== Determine Fold or Home Position ====

int determineFoldOrHome (int pinA, int fold) {

unsigned long currentMillisFold = 0;

bool buttonHome = digitalRead (pinA);

// ----- FOLD TO HOME

if (buttonHome == HIGH and fold == LOW and tanda == 0) {

tanda = 1;

previousMillisFold = millis();

}

else if (buttonHome == HIGH and fold == LOW and tanda == 1) {

currentMillisFold = millis();

if (currentMillisFold - previousMillisFold >= 7000) {

tanda = 0;

fold = 1;

}

}

else if (buttonHome == LOW and fold == LOW) {

tanda = 0;

previousMillisFold = 0;

}

// ---- HOME TO FOLD

if (buttonHome == HIGH and fold == HIGH and tanda == 0) {

tanda = 1;

previousMillisFold = millis();

}

else if (buttonHome == HIGH and fold == HIGH and tanda == 1) {

currentMillisFold = millis();

if (currentMillisFold - previousMillisFold >= 7000) {

tanda = 0;

fold = 0;

}

}

else if (buttonHome == LOW and fold == HIGH) {

tanda = 0;

previousMillisFold = 0;

}

return fold;

}

Setelah *arm manipulator* berada pada posisi HOME. Maka untuk memulai pembacaan sensor, User harus menekan tombol *Start* karena pada posisi HOME *arm manipulator* masih berada pada mode Freeze.

// START MODE

if (mode == 1) {

// Generate step for stepper (sebenernya diganti jadi generate degree sih)

steppJ1 = generateStep (1, currentStatePot1, steppJ1);

steppJ2 = generateStep (2, currentStatePot2, steppJ2);

steppJ3 = generateStep (3, currentStatePot3, steppJ3);

steppJ4 = generateStep (4, currentStatePot4, steppJ4);

yawJ5 = determineYawValue(joyYaw, yawJ5, minAngleJ5, maxAngleJ5);

rollJ6 = determineRollValue(joyRoll, rollJ6);

// Read Gripper Value

gripCount = determineGripperValue(pinButtonGripper, pinButtonRelease, gripCount);

// Home Positioning

homingPositionJoystick(pinButtonHome);

// LCD Mode Joystick

LCDJoystick(); // pake variable global

// Send Data to Slave

if (timeAfter - timeBefore >= 1000) {

// Assemble data dan kirim

String out = "#" + String(kode1) + "#" + String(steppJ1) + "#" + String(steppJ2) + "#" + String(steppJ3) + "#" + String(steppJ4) + "#" + String(yawJ5) + "#" + String(rollJ6) + "#" + String(gripCount) + "#" + String(fold) + "#$";

**Serial**.println (out);

**Serial3**.print (out);

timeBefore = millis();

//Indikator pengiriman data (TX)

SendLEDIndicator();

}

}

// FREEZE MODE

else if (mode == 0) {

// pergerakan pot diabaikan dan tidak bisa menggenerate step stepper

lastStatePot1 = currentStatePot1;

lastStatePot2 = currentStatePot2;

lastStatePot3 = currentStatePot3;

lastStatePot4 = currentStatePot4;

// Kembali ke Fold Position (ketika semua joint bernilai 0)

if (steppJ1 == 0 and steppJ2 == 0 and steppJ3 == 0 and steppJ4 == 0 and yawJ5 == 0 and rollJ6 == 0) {

fold = determineFoldOrHome(pinButtonHome, fold);

}

LCDJoystick();

**Serial**.println ("Freeze in Joystick mode");

// Send Data to Slave (Untuk pengiriman data fold)

if (timeAfter - timeBefore >= 1000) {

// Assemble data dan kirim

String out = "#" + String(kode1) + "#" + String(steppJ1) + "#" + String(steppJ2) + "#" + String(steppJ3) + "#" + String(steppJ4) + "#" + String(yawJ5) + "#" + String(rollJ6) + "#" + String(gripCount) + "#" + String(fold) + "#$";

**Serial**.println (out);

**Serial3**.print (out);

timeBefore = millis();

}

}

Kemudian setelah tombol *start* ditekan sensor akan di baca sebagai masukkan data. Untuk membuat hasil data yang baik diperlukan program pembacaan data dan *smoothing* data. Program tersebut di buat dengan sebuah function bernama *readPotensiometer* ()

// ==== Potensiometer Reading & Smoothing Functions =====

int readPotensiometer(int pinPot) {

int i;

int sval = 0;

int sampel = 16;

for (i = 0; i < sampel; i++) {

sval = sval + analogRead(pinPot);

}

sval = sval / sampel; //average (scale 0-1023)

// sval = sval / 4; //scale to 8 bits (0-255)

return sval;

}

int readPotensiometerInvert (int pinPot) {

int i;

int sval = 0;

int sampel = 16;

for (i = 0; i < sampel; i++) {

sval = sval + analogRead(pinPot);

}

sval = sval / sampel; //average (scale 0-1023)

sval = 1023 - sval;

return sval;

}

Selanjutnya data hasil pembacaan tersebut di *scalling* ke dalam bentuk satuan derajat dengan fungsi *mapping ()* dan dinamakan dengan function *generateStatePot ()*

// ==== Mapping Potensio to Joint Degree Range ======

// Mapping nilai potensiometer ke range yang sesuai dengan masing2 joint nya

int generateStatePot (int pinPot, int batasMin, int batasMax) {

int potensioVal = 0;

int stateVal = 0;

// Untuk joint 2 (karena elektriknya terbalik)

if (pinPot == 1) {

potensioVal = readPotensiometerInvert(pinPot);

}

else {

potensioVal = readPotensiometer(pinPot);

}

stateVal = map (potensioVal, 0, 1023, batasMin, batasMax);

return stateVal;

}

Hasil dari scalling kemudian dibuatkan function untuk hasil akhir pengolahan data pada sensor. Dibuatkan pada function bernama *generateStep()*

// ===== Generate Step Value from Potensio =====

int generateStep (int gerak, int currentStatePot, int stepp) {

// Potensio 1

if (gerak == 1) {

if (currentStatePot != lastStatePot1) {

if (currentStatePot > lastStatePot1) {

int val = 0;

int i;

val = currentStatePot - lastStatePot1;

// kalo mode start baru jalanin stepper

for (i = 0; i < val; i++) {

stepp++;

stepp = BatasMax (stepp, maxAngleJ1);

}

}

else if (currentStatePot < lastStatePot1) {

int val = 0;

int i;

val = lastStatePot1 - currentStatePot;

for (i = 0; i < val; i++) {

stepp--;

stepp = BatasMin (stepp, minAngleJ1);

}

}

}

lastStatePot1 = currentStatePot;

}

Program di atas berlaku untuk ke 4 *joint master controller* pada sensor potensiometer. Sedangkan pada Modul Joystick sebagai pergerakan untuk *joint* 5 dan 6 dibuatlah *function* dengan nama *readJoystick()* untuk pembacaan , *determineYawValue()* dan *determineRollValue()* untuk pengolahan hasil data.

// ==== Reading Analog Joystick Sensor =====

int readJoystick(int pinJoy) {

int i;

int sval = 0;

int sampel = 16;

for (i = 0; i < sampel; i++) {

sval = sval + analogRead(pinJoy);

}

sval = sval / sampel; //average

// sval = sval / 4;

return sval;

}

// ====== Generate Yaw Joystick to Step Value =====

int determineYawValue (int joyVal, int yaw, int minAngle, int maxAngle) {

unsigned long currentMillis = millis();

// Pertambahan

if (joyVal >= 650) {

if (currentMillis - previousMillisYaw >= 5) {

yaw ++;

yaw = BatasMax (yaw, maxAngleJ5);

previousMillisYaw = currentMillis;

}

}

// Pengurangan

else if (joyVal <= 400) {

if (currentMillis - previousMillisYaw >= 5) {

yaw --;

yaw = BatasMin (yaw, minAngleJ5);

previousMillisYaw = currentMillis;

}

}

return yaw;

}

Terakhir terdapat *Function* Umum dimana fungsi ini mengkondisikan mode start dan mode freeze *determineModeStartFreeze()* kemudian mode Homing untuk *arm manipulator* Kembali pada posisi awal yaitu posisi HOME, *homingPositionJoystick().* Adapaun yang terakhir fungsi umtuk menentukan batas maksimal dan minimal degree dengan fungsion bernama *BatasMax()* dan *BatasMin()*.

1. Fungsi pengkondisian mode Start dan Freeze :

// ==== Mode Start / Freeze Function ====

int determineModeStartFreeze (int pinButtonStart, int pinButtonFreeze, int mode) {

bool btnStartVal = digitalRead (pinButtonStart);

bool btnFreezeVal = digitalRead (pinButtonFreeze);

if (btnStartVal == HIGH && btnFreezeVal == LOW) {

mode = 1;

}

else if (btnStartVal == LOW && btnFreezeVal == HIGH) {

mode = 0;

}

return mode; }

1. Fungsi pengkondisian mode Homing Position :

// ==== Homing Position Joystick Function ====

void homingPositionJoystick (int pinHoming) {

bool buttonValue = digitalRead (pinHoming);

homingIndicator (buttonValue);

if (buttonValue == HIGH) {

steppJ1 = 0;

steppJ2 = 0;

steppJ3 = 0;

steppJ4 = 0;

yawJ5 = 0;

rollJ6 = 0;

}

}

// ==== Fungsi untuk menentukan batas maksimal degree ====

int BatasMax (int counter, int batas) {

if (counter >= batas) {

counter = batas;

}

else {

//pass

}

return counter;

}

1. Fungsi untuk menentukan batas maksimal dan minimal sudut:

// ==== Fungsi untuk menentukan batas minimal degree ====

int BatasMin (int counter, int batas) {

if (counter <= batas) {

counter = batas;

}

else {

//pass

}

return counter;

}

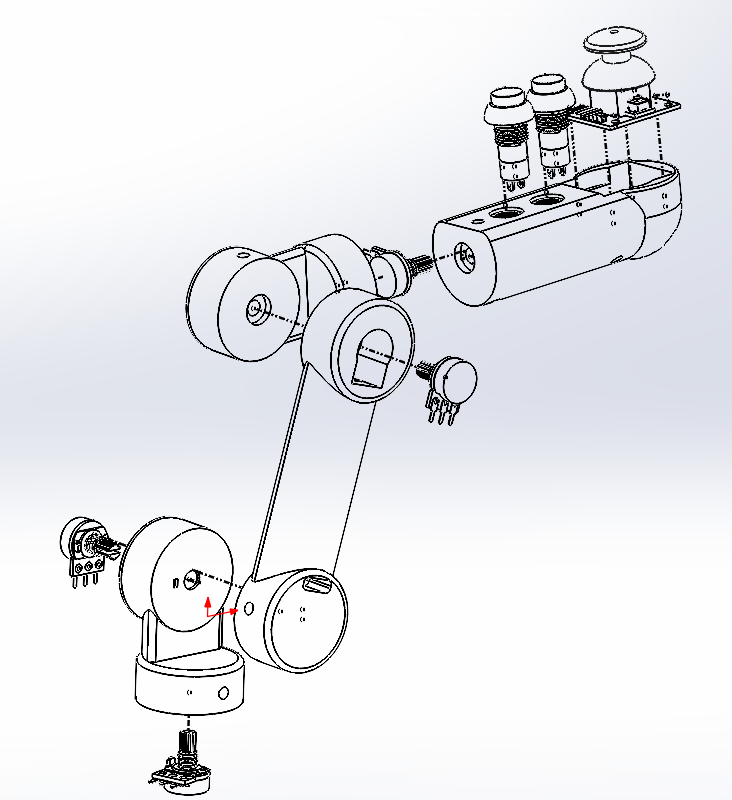
1. Mekanik *Master controller*

Proses mekanik pada *master controller* dapat ditemukan pada bagian Joystick *master controller.* Dimana *Joystick master controller* dibuat agar menyerupai alat dan prinsip kerja dari pengendali lengan robot ROV yang sesunggunya.

* + 1. Pembuatan *Joystick master controller*

Pada peroses pembuatan *Joystick master controller* sudut pergrakkan antara *joint* ditopang menggunakan potensiometer sebagai penggerak dan sekaligus sebagai sensor pembacaan gerakan *joint*. Diketahui sebuah potensiometer linear mempunyai sudut maksimal perputaran sebesar 270° hal inilah yang menjadi prinsip pergrakan pada *Joystick master controller.*

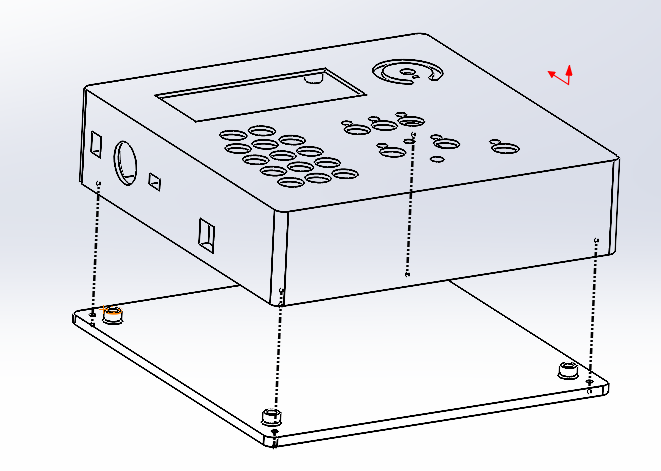
Pergerakan *joint* *Joystick master controller* memaksimalkan sudut tertinggi yang bisa dicapai oleh sensor Potensiometer, agar nantinya setiap *joint* pada *master controller* dapat di atur posisi pengambilan data awalnya. Karena dalam pengaplikasian ROV (*Remotely Operated Vehicle*), lengan robot lebih ditekankan pada mobilitas nya dibandingkan ketepatannya dikarenakan lingkungan pengaplikasiannya yang dinamis. Untuk ketepatan, nantinya akan dipantau oleh operator yang mengoperasikan lengan robot.



**Gambar 2. 24** Konstruksi *Joystick master controller*

Gambar 2.24 menunjukkan konstruksi pembuatan *Joystick master controller* menggunakan software Solidworks untuk setiap *part* pada *joint*. Setelah semua *part* sudah dibuat. Selanjutnya dilakukan proses assembly agar hasilnya dapat di lihat dan di simulasikan secara 3D.

* + 1. Pembuatan *Panel master controller*



**Gambar 2. 25** Konstruksi *Panel master controller*

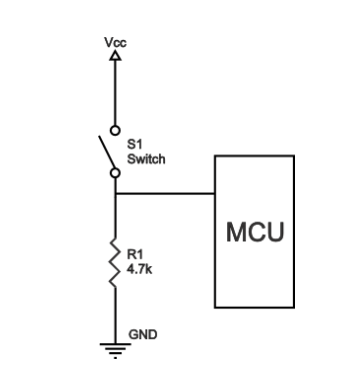
Gambar 2.25 menunjukkan konstruksi Panel master controller yang dibuat dengan software Solidworks. Panel ini merupakan tempat terpasangnya seluruh rangkaian elektrik dan juga Joystick master controller. Terlihat pada gambar di atas panel terdiri dari dua bagian part yang diikat dengan empat buah baut yang di ilustrasikan dengan garis putus.

1. Elektrik *Master controller*

Proses elektrik *master controller* dibuat dengan membagi dua buah rangkaian PCB yang natinya akan tersambung secara terpusat pada suatu rangkaian *Motherboard.* Rangkaian ini terdiri dari mikrokontroler sebagai proses pengolahan data juga pengiriman data. Mikrokontroler mendapatkan input dari beberapa sensor yaitu potensiometer, push button, saklar toggle dan sebuah modul Joystick. Data hasil pembacaan sensor diolah dan di tampilkan melalui sebuah *display* LCD 20x4. Bersamaan dengan itu data dikirimkan melalui komunikasi data serial *rx/tx* yang terhubung dengan sebuah s*lave* mikrokontroler.

Komponen sensor utama pada *master controller* ini berupa potensiometer dimana sensor ini menjadi sensor pembacaan data pada *joint* 1, *joint* 2, *joint* 3 dan juga *joint* 4. Sedangkan pada *joint* 5 dan *joint* 6 pembacaan data dihasilkan dari pergerakkan modul Joystick pada axis X dan Y.

Sedangkan sensor push button digunakan untuk pengoperasian kondisi pada lengan robot. Rangkaian sensor push button menggunakan resistor PULL\_DOWN dimana resistor dihubungkan dengan Ground, sehingga saat kondisi normal (tombol tidak ditekan) mikrokontroler akan terhubung dengan ground.



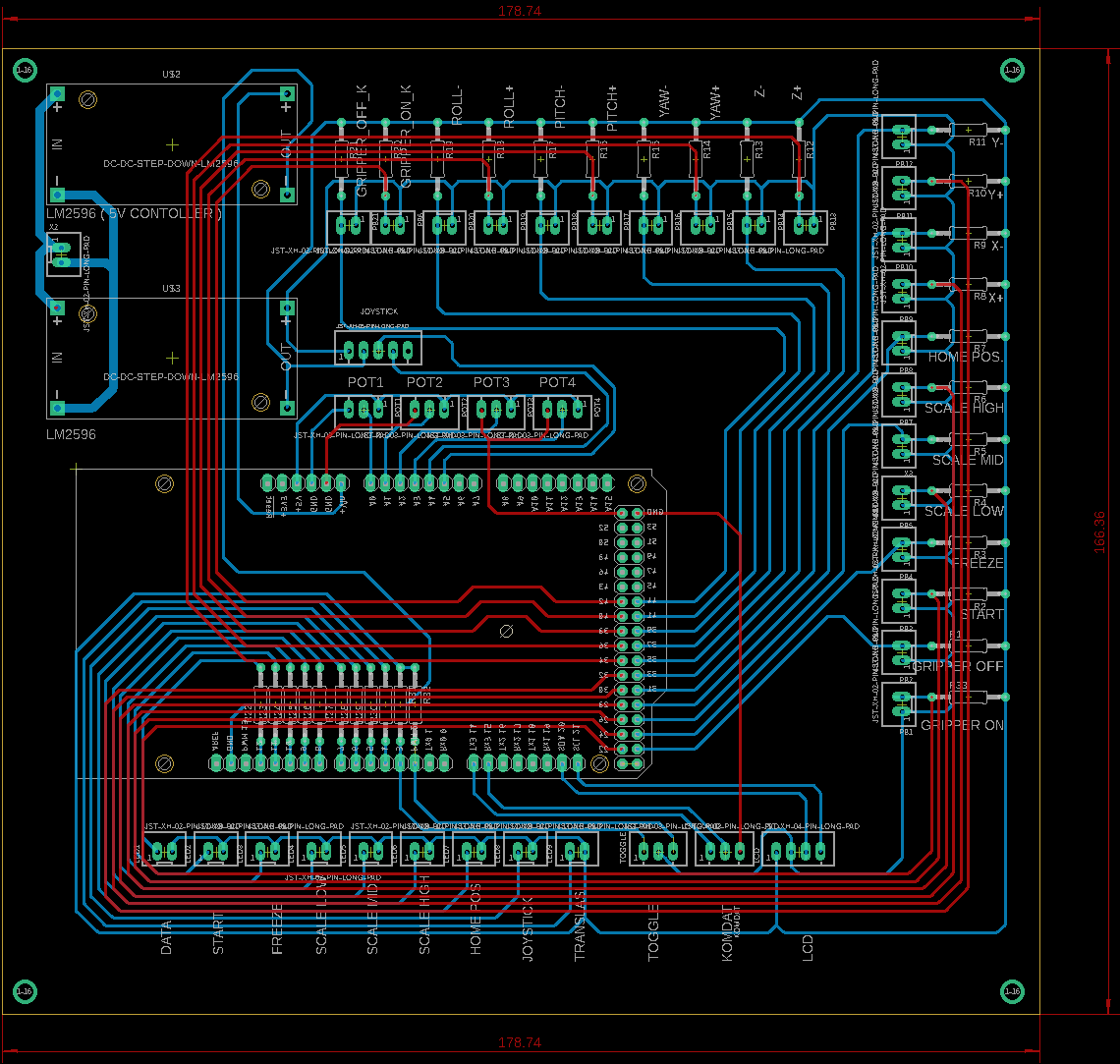
**Gambar 2. 26** Resistor PULL DOWN

 Resistor PULL\_DOWN harus memiliki resistansi yang lebih besar daripada impedansi rangkaian logika, atau mungkin akan menarik tegangan keseluruhan ke bawah terlalu banyak. Ini akan menyebabkan tegangan input pada pin tetap pada kondisi LOW.

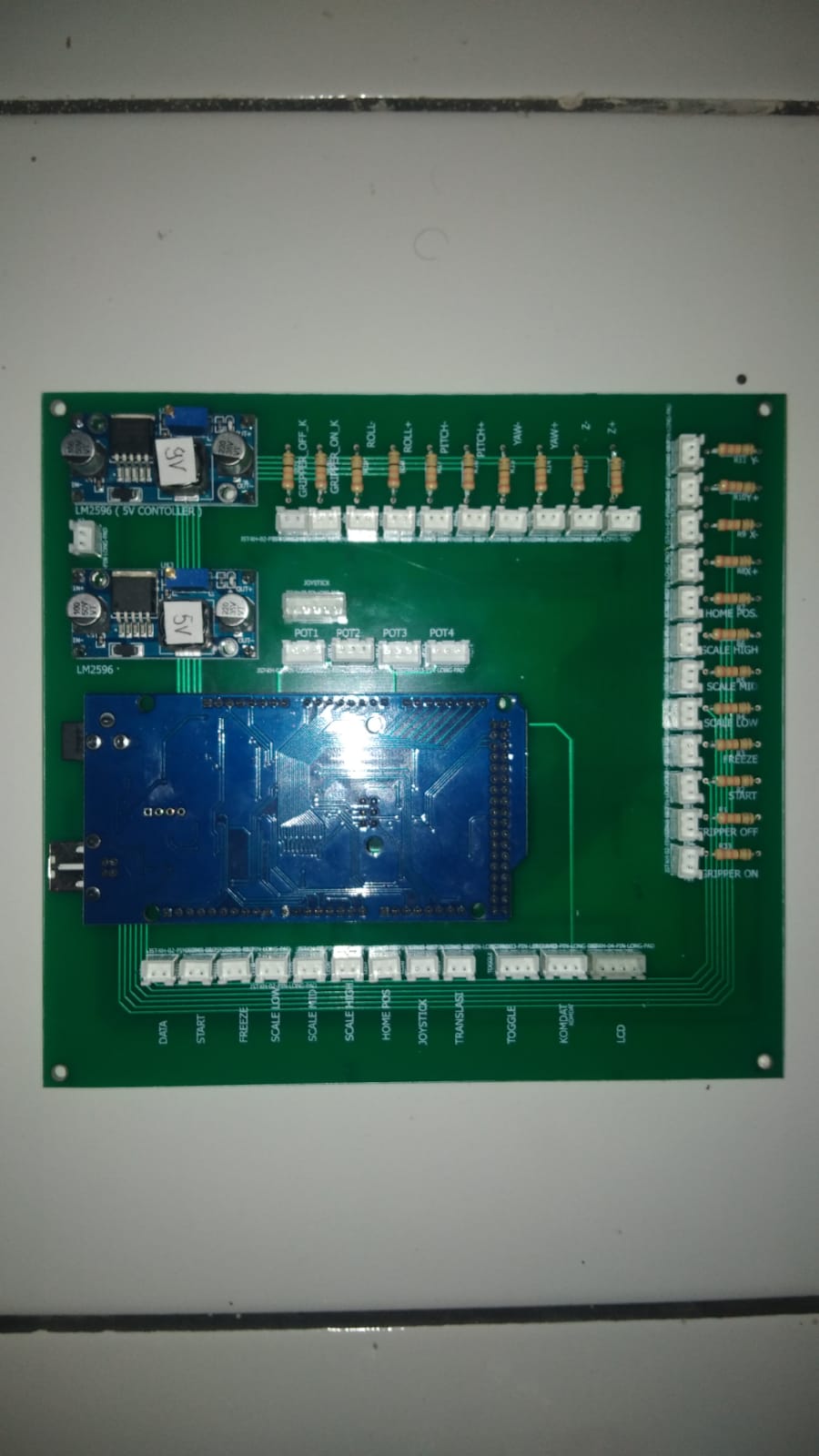
*Power supply* pada *master controller* menggunakan adaptor AC-DC *converter* dengan output tegangan 12 VDC dan arus 2A, Adaptor ini akan menjadi sumber utama dari *master controller* yang kemudian akan melalui dua buah LM2596 sebagai *step-down* tegangan5 VDC untuk memberi daya terpisah pada mikrokontroler dan berbagai inputan komponen yang di pakai.

* + 1. Rancangan rangkaian *Motherboard master controller*

Pembuatan rangkaian elektrik pada *master controller* dibuat secara terpusat pada suatu rangkaian *motherboard*. Rangkaian ini terdiri dari *microcontroller arduino*, sensor potensiometer, sensor *analog joystick*, *push buttons, pull down resistors* saklar toggle, *power supply unit for microcontroller, power supply unit for sensors*, LCD I2C *connections,* dan UART *communication to slave arduino*.



**Gambar 2. 27** Rangkaian *Motherboard master controller*

****

**Gambar 2. 28** *Design* PCB *Motherboard*

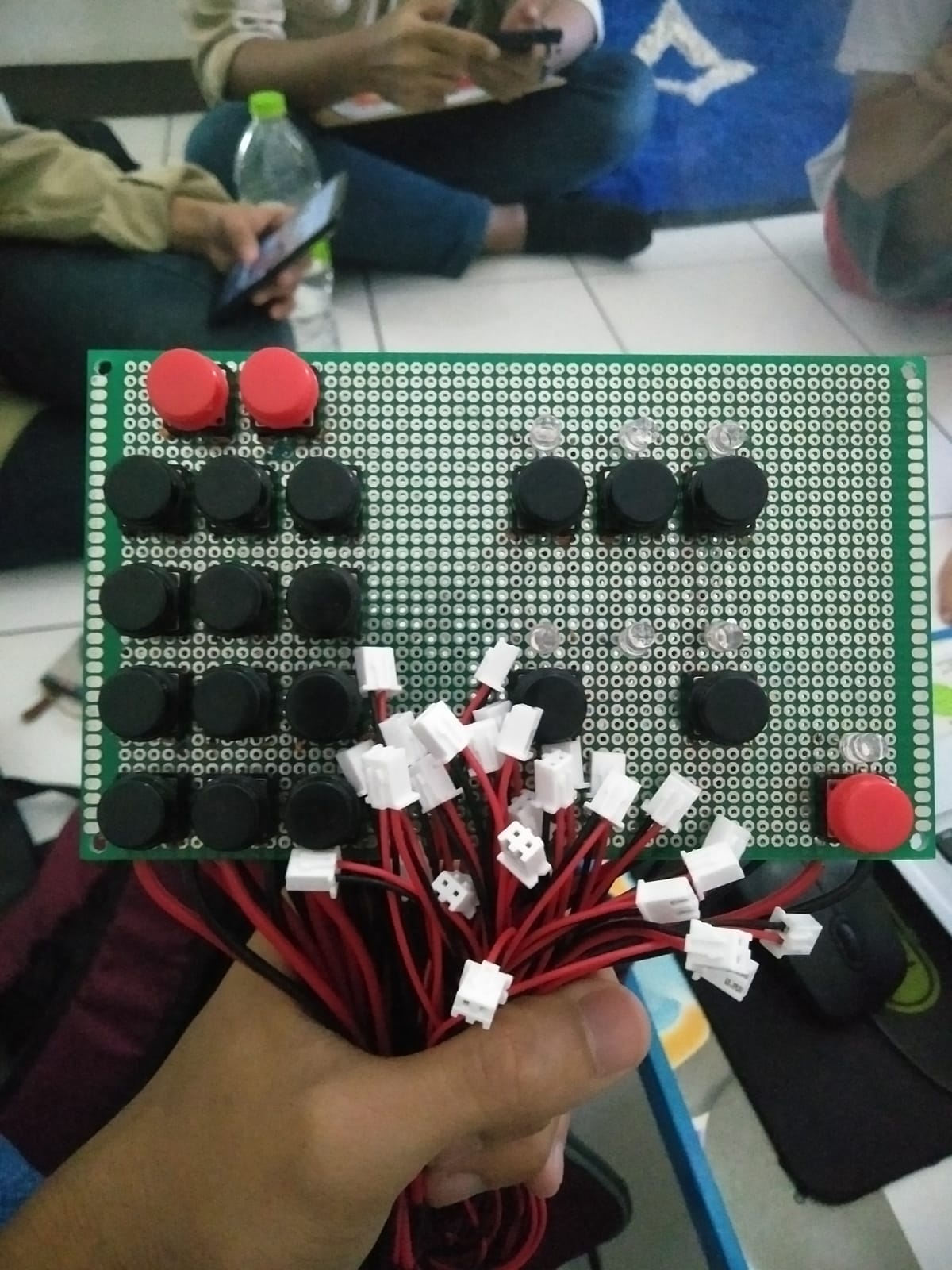
*Motherboard* ini menjadi tempat terpasangnya seluruh komponen sensor, modul dan mikrokontroler melalui sebuah pin *JST* untuk menghubungkan berbagai komponen yang terpasang agar lebih mudah untuk di *maintenance* dan memperkecil dimensi perancangan. Gambar 2.38 menunjukkan hasil desain PCB pada motherboard serta Gambar 2.27 menunjukkan *layout* rangkaian motherboard agar lebih mudah dipahami.

Pada *layout* rangkaian *motherboard* terlihat bahwa Arduino Mega menjadi tempat terhubungnya berbagai komponen *JST.* Pada tabel di bawah ini akan di tunjukkan I/O yang dihubungkan pada suatu komponen dengan keterangan fungsi dari pin-pin tersebut.

**Tabel 2. 8** Tabel penggunaan I/O Arduino Mega pada *Master controller*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Pin | Komponen | Keterangan | Pin | Komponen | Keterangan |
| A0 | Potensiometer | Join 1 | **D23** | Push Button | Gripper Close |
| A1 | Potensiometer | Join 2 | **D24** | Push Button | Start Button |
| A2 | Potensiometer | Join 3 | **D25** | Push Button | Freeze Button |
| A3 | Potensiometer | Join 4 | **D26** | Push Button | Speed Mode (LOW) |
| A4 | Modul Joystick | Joystick Yaw (Y) | **D27** | Push Button | Speed Mode (MID) |
| A5 | Modul Joystick | Joystick Roll (X) | **D28** | Push Button | Speed Mode (HIGH) |
| D2 | Toggle Switch | Right On (Joystick Mode ) | **D29** | Push Button | Home Position |
| D3 | Toggle Switch | Left On (Cartesian Mode ) | **D30** | Push Button | X+ |
| D4 | LED | ( not used ) | **D31** | Push Button | X- |
| D5 | LED | Joystick Mode | **D32** | Push Button | Y+ |
| D6 | LED | Home Position | **D33** | Push Button | Y- |
| D7 | LED | Speed Mode (HIGH) | **D34** | Push Button | Z+ |
| D8 | LED | Speed Mode (MID) | **D35** | Push Button | Z- |
| D9 | LED | Speed Mode (LOW) | **D36** | Push Button | Yaw+ |
| D10 | LED | Freeze | **D37** | Push Button | Yaw- |
| D11 | LED | Start | **D38** | Push Button | Pitch+ |
| D12 | LED | Data Delivery | **D39** | Push Button | Pitch- |
| D14 | TX | Data communication (to RX) | **D40** | Push Button | Roll+ |
| D15 | RX | Data communication (to TX) | **D41** | Push Button | Roll- |
| D20 | LCD 20x4 | SDA | **D42** | Push Button | Gripper Open in Cartesian Mode |
| D21 | LCD 20x4 | SCL | **D43** | Push Button | Gripper Close in Cartesian Mode |
| D22 | Push Button | Gripper Open |

1. Rancangan rangkaian *keyboard Push button*



**Gambar 2. 29** *Keyboard Push button*

Terlihat pada gambar 2.29 Rancangan rangkaian *keyboard push button* dibuat secara tersusun agar memudahkan pemasangan push button pada casing-nya. Rangkaian ini dibuat pada suatu papan PCB secara terpisah, dengan *motherboard* di sambungkan dengan pin *male* JST yang membuat rangkaian ini terhubung oleh pin *female* pada *motherboard* yang di sediakan.

* 1. Hasil Pengujian

Setelah menyelesaikan perancangan dan pembuatan perangkat keras robot maka dilakukan pengujian untuk mengetahui respons hasil pengiriman data pada *master controller.*

Pada pengujian respons, uji coba yang dilakukan yaitu mengetahui respons lengan robot dengan masukkan data dari *master controller* secara berulang pada satu waktu.

1. Pengujian Algoritma

Pengujian Algoritma dilakukan dengan untuk mengamati apakah terdapat algoritma yang salah pada saat pengoperasian *master controller.* Pengujian ini dapat dilakukan dengan cara melihat indikator LED dan output data yang di tampilkan pada LCD Ketika tombol operasi dijalankan.

1. Tampilan awal mode Joystick

****

**Gambar 2. 30** Tampilan awal mode Joystick

Gambar 2.30 menunjukkan tampilan pada mode *Joystick* *master controller,* mode ini diaktifkan dengan menggeser toggle switch kearah kanan. Kemudian display LCD akan menampilkan tulisan “The robot is in FOLD position please chage it to HOME position” dan indicator LED Freeze akan menyala.

Untuk merubah FOLD *position* ke HOME *position* adalah dengan cara menekan tombol Home selama 7 detik. Seketika *arm manipulator* akan bergerak menuju posisi home-nya dan *display* akan langsung berubah seperti pada gambar di bawah ini:



**Gambar 2. 31** Tampilan mode Joystick FOLD to HOME position

Pada posisi ini *arm manipulator* akan berada pada posisi Home. Untuk memulai pergerakkan tekan tombol ‘*start*’ kemudian akan tampil seperti pada gambar di bawah ini:



**Gambar 2. 32** Tampilan mode joystick saat tombol *start*ditekan

Terlihat kondisi pada saat di tekan tombol ‘*start*’ display akan menampilkan data pergerakkan sesuai denagn *joint* yang di gerakkan pada J*oystick master controller.* LED hijau akan menyala sebagai indikator tombol ‘*start*’ sudah ditekan.

Untuk mengembalikan posisi *arm manipulator* pada kondisi Home, maka tekan tombol Home sekali, kemudian akan muncul indicator LED berwarna biru. Dan *display* akan mengembalikan nilai posisi antar *joint* ke posisi Home atau posisi ‘0’. Dapat dilihat pada gambar di bawah ini:



**Gambar 2. 33** Tampilan mode Joystick saat tombol *Home* ditekan

Kemudian untuk membuat *arm manipulator* berhenti pada posisi pergerakkan tertentu tekan tombol ‘*freeze*’ dan robot akan terjeda pada posisi yang di tentukan,pada posisi ini *master arm* tidak dapat memberikan input pergerakkan. Kemudian Indikator LED akan menyala berwarna merah.



**Gambar 2. 34** Tampilan mode Joystick saat tombol *Freeze* ditekan

Pada posisi ‘Freeze’ ini robot dapat dikembalikan ke posisi FOLD dengan syarat value tiap *joint* adalah ‘0’. Atau tiap *joint* telah perpindah ke posisi Home terlebih dahulu.

1. Tampilan awal mode Kartesian



**Gambar 2. 35** Tampilan awal mode Kartesian

Gambar 2.35 menunjukkan tampilan pada mode Kartesian, mode ini diaktifkan dengan menggeser toggle switch kearah kiri. Kemudian display LCD akan menampilkan tulisan “MODE KARTESIAN IS ON DEVELOPMENT” yang artinya mode kartesian ini belum selesai dan menjadi saran pengembangan pada proyek akhir ini.

1. Tampilan awal mode Neutral



**Gambar 2. 36** Tampilan awal mode Neutral

Gambar 2.36 menunjukkan tampilan pada mode Neutral, mode ini diaktfikan dengan mengarahkan posisi *toggle switch* pada posisi tengah. Pada mode ini berisi informasi pengerjaan proyek akhir ini.

1. Pengujian Respons

Pengujian respons dilakukan dengan cara memberikan input data pergerakan dari *master controller* berupa gerakkan berulang dengan parameter waktu tunda pengiriman data serial dan kecepatan motor stepper. Hasil yang akan di analisis berupa jumlah gerkan yang dieksekusi dan waktu respons terhadap pergerakannya.

Penjelasan parameter dalam tabel:

* Data pergerakkan berulang, merupakan data pergerakkan *Joystick* *master controller* berupa satu pergerakkan penuh pada satu joint.
* Delay data serial, merupakan waktu tunda setiap pengiriman data dengan satuan waktu (ms).
* Kecepatan *stepper* motor, merupakan kecepatan yang di inputkan dengan menggunakan fungsi dari library *accelstepper()*
* Jumlah pergerakkan yang dieksekusi, merupakan gerakan yang di eksekusi oleh a*rm manipulator* terhadap inputan data yang diberikan.
* Waktu respons, merupakan banyaknya waktu yang diperoleh dari pergerakkan yang di eksekusi oleh arm manipulator.

1. Tabel pengujian respons

**Tabel 2. 9** Tabel pengujian respons

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sampel** | **Data Pergerakkan  berulang** | **Delay  data serial (mS)** | **Kecepetan *Steppe*r Motor** | **Jumlah Pergerakkan yang dieksekusi** | **Waktu respons (s)** |
|  |
| 1 | 1 x | 700 | 500 | 1 | 3.85 |  |
| 2 | 2 x | 2 | 8.9 |  |
| 3 | 3 x | 2 | 12.61 |  |
| 4 | 4 x | 3 | 9.46 |  |
| 5 | 5 x | 3 | 11.85 |  |
| 6 | 1 x | 300 | 1 | 4.18 |  |
| 7 | 2 x | 2 | 10.43 |  |
| 8 | 3 x | 2 | 7.72 |  |
| 9 | 4 x | 2 | 11.81 |  |
| 10 | 5 x | 3 | 14.1 |  |
| 11 | 1 x | 100 | non-valid | non-valid |  |
| 12 | 2 x | non-valid | non-valid |  |
| 13 | 3 x | non-valid | non-valid |  |
| 14 | 4 x | non-valid | non-valid |  |
| 15 | 5 x | non-valid | non-valid |  |
| 16 | 1 x | 700 | 1000 | 1 | 2.42 |  |
| 17 | 2 x | 2 | 5.78 |  |
| 18 | 3 x | 3 | 6.19 |  |
| 19 | 4 x | 3 | 6.84 |  |
| 20 | 5 x | 3 | 7.56 |  |
| 21 | 1 x | 300 | 1 | 2.4 |  |
| 22 | 2 x | 2 | 6.76 |  |
| 23 | 3 x | 3 | 6.98 |  |
| 24 | 4 x | 3 | 7.95 |  |
| 25 | 5 x | 3 | 9.4 |  |
| 26 | 1 x | 100 | non-valid | non-valid |  |
| 27 | 2 x | non-valid | non-valid |  |
| 28 | 3 x | non-valid | non-valid |  |
| 29 | 4 x | non-valid | non-valid |  |
| 30 | 5 x | non-valid | non-valid |  |

Pada pengujian yang dilakukan diperoleh hasil bahwa *arm manipulator* tidak memiliki respons yang baik terhadap pengiriman data oleh *master controller.* Terlihat pada ketiga kecepatan motor yang di inputkan data pergerakkan yang berhasil di eksekusi dengan baik hanya sampai pada inputan pergerakkan berulang 3x, dimana data pergerakkan maksimal hanya dapat di ekseksui tiga kali pergerakkan secara berulang pada waktu kenaikan yang konstan.

Dari data di atas *arm manipulator* tidak dapat mengeksekusi inputan data masuk pada waktu tunda pengiriman data 100ms kemudian terjadi kenaikan waktu responsif pergerakan pada waktu tunda pengiriman data 300ms. Ini dikarenakan semakin kecil waktu tunda akan semakin banyak data yang dikirimkan dan semakin cepat pengiriman datanya, sehingga *arm manipulator* menerima banyak data tetapi tidak dapat dieksukusi dengan baik. Terlihat pada pengujian di atas waktu tunda pengiriman data terbaik adalah 700ms.

1. Pengujian input pergerakkan *master controller* terhadap *joint arm manipulator*

Pengujian dilakukan untuk melihat bagaimana reaksi *arm manipulator* terhadap input pergerakkan tiap *joint* dari *master controller* dan akurasinya pada sudut yang di tentukan. Nilai sudut pergerakkan akan di inputkan dan hasilnya akan diukur dengan menggunakan alat ukur manual dengan nama sudut aktual.

Adapun untuk mengukur ketepatan hasil pembacaan posisi dilakukan perbandingan seperti persamaan berikut:



**Gambar 2. 37** Rumus mencari persentase *error*

Berdasarkan data pengujian yang ditunjukkan pada tabel dibawah. Pengujian dilakukan secara berulang sebanyak 4 kali dengan nilai inputan sudut yang variatif.

Penjelasan parameter dalam tabel:

* Joint, merupakan parameter joint yang di analisis
* Sudut masukan *Master Controller*, merupakan sudut yang di inputkan pada *master controller.*
* Sudut aktual, merupakan sudut yang dihasilkan dari *arm manipulator* terhadap parameter masukkan. Sudut ini diukur menggunakan alat ukur manual yaitu busur digital dengan resolusi 0.05° dan akurasi ± 0.03°.
* *Error,* merupakan perbandingan kesalahan antara sudut yang dihasilkan terhdap sudut aktual.

**Tabel 2. 10** Pengujian input pergerakkan *master controller* terhadap *joint* 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Percobaan | *Joint* | Sudut masukan Master Controller (°) | Sudut Aktual (°) | *Error* |
| (%) |
| 1 | 1 | 90 | 89.75 | 0.2778 |
| 2 | 45 | 45.15 | 0.333 |
| 3 | -30 | -30.5 | 1.6667 |
| 4 | -60 | -60 | 0 |
| Rata-Rata Error | | | | 0.5739667 |

**Tabel 2. 11** Pengujian input pergerakkan *master controller* terhadap *joint* 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Percobaan | Joint | Sudut masukan Master Controller (°) | Sudut Aktual (°) | Error |
| (%) |
| 1 | 2 | 90 | 90.05 | 0.0556 |
| 2 | 45 | 45.15 | 0.333 |
| 3 | -30 | -30.05 | 0.1667 |
| 4 | -45 | -45.35 | 0.7778 |
| Rata-Rata Error | | | | 0.333275 |

**Tabel 2. 12** Pengujian input pergerakkan *master controller* terhadap *joint* 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Percobaan | Joint | Sudut masukan Master Controller (°) | Sudut Aktual (°) | Error |
| (%) |
| 1 | 3 | 45 | 45.15 | 0.333 |
| 2 | 25 | 25.05 | 0.2 |
| 3 | -60 | -59.5 | 0.833 |
| 4 | -90 | -89.5 | 0.5556 |
| Rata-Rata Error | | | | 0.4804 |

**Tabel 2. 13** Pengujian input pergerakkan *master controller* terhadap *joint* 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Percobaan | Joint | Sudut masukan Master Controller (°) | Sudut Aktual (°) | Error |
| (%) |
| 1 | 4 | 180 | 180.05 | 0.0278 |
| 2 | 90 | 90.15 | 0.167 |
| 3 | -45 | -45.05 | 0.111 |
| 4 | -60 | -60.05 | 0.0833 |
| Rata-Rata Error | | | | 0.097275 |

**Tabel 2. 14** Pengujian input pergerakkan *master controller* terhadap *joint* 5

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Percobaan | Joint | Sudut masukan Master Controller (°) | Sudut Aktual (°) | Error |
| (%) |
| 1 | 5 | 90 | 90.15 | 0.1667 |
| 2 | 45 | 45 | 0 |
| 3 | -30 | -29.9 | 0.333 |
| 4 | -60 | -60.5 | 0.8333 |
| Rata-Rata Error | | | | 0.16675 |

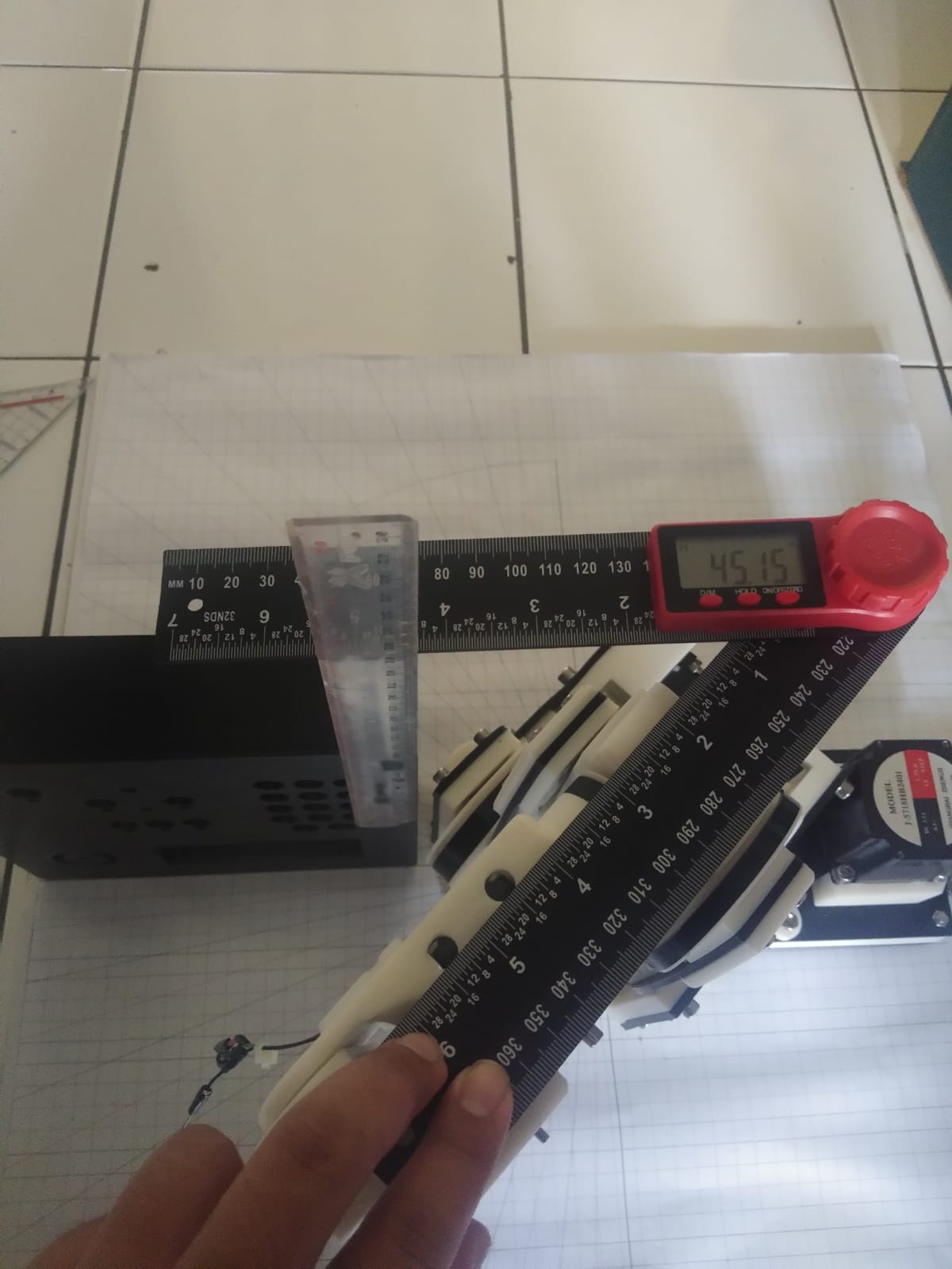
**Tabel 2. 15** Pengujian input pergerakkan *master controller* terhadap *joint* 6

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Percobaan | Joint | Sudut masukan Master Controller (°) | Sudut Aktual (°) | *Error* |
| (%) |
| 1 | 6 | 210 | 210.3 | 0.1429 |
| 2 | 180 | 180.15 | 0.0833 |
| 3 | -210 | -210.1 | 0.0476 |
| 4 | -180 | -180.05 | 0.0278 |
| Rata-Rata Error | | | | 0.0754 |

|  |  |
| --- | --- |
| **Joint** | **Rata-Rata Error** |
| 1 | 0.5739667 |
| 2 | 0.333275 |
| 3 | 0.4804 |
| 4 | 0.097275 |
| 5 | 0.16675 |
| 6 | 0.0754 |
| **Rata-Rata Error Keseluruhan** | 0.2878445 |

**Tabel 2. 16** Rata-rata *error* pada tiap *joint*

Berdasarkan pengujian yang telah dilakukan, didapatkan hasil bahwa akurasi motor stepper berdasarkan input pergerakkan sudut oleh *master controller* yang diberikan sangat baik. Terlihat pada persen error yang didapat, pada tiap *joint* mendapat error hanya pada kisaran nol koma sekian persen. Terlebih lagi setelah di rata-ratakan didapat hasil error keseluruhan *joint* yaitu pada 0.287%.

 ****

**Gambar 2. 38** Sampel pengambilan data

# BAB III

# KESIMPULAN

## Kesimpulan

Adapun hasil yang telah dicapai berdasarkan tujuan dan tuntutan secara umum dalam proyek akhir ini yaitu:

1. Pembuatan *arm manipulator* 6 DOF dapat dibangun dengan pengendali *master controller.* Pada perancangannya terdapat dua mode pergerakkan yaitu mode Joystick dan mode Kartesian. Pada proyek akhir ini *master controller* mampu mengendalikkan *arm manipulator* 6 DoF dengan mode Joystick. Dengan fitur yang mampu diterapkan untuk *Master controller* pada proyek akhir ini yaitu *Homing position, Fold position, Start* dan *Freeze*,
2. Komunikasi yang dipakai pada *Master controller* yaitu menggunakan komunikasi serial UART dengan pin *rx/tx.*
3. *Arm manipulator* tidak memiliki respons yang baik terhadap pengiriman data oleh *master controller,* salah satu alasannya karena pemilihan *Arduino Mega* sebagai *controller.* Tetapi jika mengacu pada ROV, respons yang cepat tidak diutamakan dikarenakan operator dapat mengatur sendiri kecepatan pergerakkannya pada sebuah *master controller.*
4. Akurasi motor stepper yang diaplikasikan pada *arm manipulator* 6 DoF terhadap Input pergerakkan *master controller* yang diberikan menunjukkan bahwa motor stepper memiliki akurasi dengan rata-rata error terhadap seluruh joint yaitu sebesar 0,287%.

## Saran

Berdasarkan pembahasan dan pengujian mengenai proyek akhir ini. Terdapat beberapa saran, diantaranya sebagai berikut:

* + 1. Dalam penelitian kedepannya di harapkan dapat merancang mode Kartesian dengan metode inverse kinematic agar robot dapat bergerak berdasarkan koordinat masukan yang diinginkan.
    2. Menggunakan *mikrokontroler* atau *mini computer* dengan *processor* yang lebih tinggi meminimalkan latensi pengiriman data.
    3. Pada perancangan *master controller* dan *arm manipulator, management* pengkabelan dapat teratasi dengan rapih dan tetap fungsional.

# DAFTAR PUSTAKA

[1] D. P. Putra, “Perbedaan Transmisi Data Serial dan Paralel,” pp. 1–10.

[2] M. W. Spong, S. Hutchinson, and M. Vidyasagar, “Robot modeling and control,” *IEEE Control Syst.*, vol. 26, no. 6, pp. 113–115, 2006, doi: 10.1109/MCS.2006.252815.

[3] R. D. Christ and R. L. Wernli, *The ROV manual: a user’s guide to remotely operated vehicles*. 2014.

[4] Aldy Razor, “Arduino Mega 2560: Pengertian, Harga, dan Spesifikasi,” *Aldyrazor.Com*, pp. 1–10, 2020.

[5] E. Purnomo, “Nulis- Potensiometer : Pengertian Potensiometer,” 2015. https://nulis-ilmu.com/potensiometer/

[6] A. Razor, “Komunikasi Serial Arduino dengan PC : Pengertian dan Program,” pp. 1–19, 2020.

# LAMPIRAN

Semua dokumentasi mekanik, elektrik dan program dihimpunkan ke dalam folder github yang berada pada tautan :

<https://github.com/FachriMFulchan/Arm_Manipulator_7_Functions>

Bentuk fisik pengambilan data *pergerakkan arm manipulator* :

|  |  |  |
| --- | --- | --- |
| Percobaan | Sudut masukan Master Controller (°) | Bentuk Fisik pengambilan data |
|
| 1 | 90 |  |
| 2 | 45 |  |
| 3 | -30 |  |
| 4 | -60 |  |  |