Authors’ Instructions: Master Controller Prototype Design To Control Arm Manipulator 6 Dof Based On Arduino Microcontroller

Raihan Ramadhan1[[1]](#footnote-2), Setyawan Ajie Sukarno1[[2]](#footnote-3) and Susetyo Bagas Bhaskoro2[[3]](#footnote-4)

Department of Automation Engineering, Bandung Polytechnic of Manufacturing, Bandung,Indonesia.

[dhanraihan@gmail.com](mailto:dhanraihan@gmail.com), [setyawanajies@gmail.com](mailto:setyawanajies@gmail.com), [bagas@ae.polman-bandung.ac.id](mailto:bagas@ae.polman-bandung.ac.id)

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

Abstract: To be able to control a manipulator requires a remote controller that can be controlled by the user manually. This design focuses on making a controlling device called the Master controller where this controlling device is a manipulator controller on an underwater robot or Underwater-Remotly Operated Vehicle (ROV). In this design, the controller used is Arduino Mega with a potentiometer sensor as input for movement of 6 moving stepper motors. The test results provide an achievement that the robot can receive data by a pendant controller smoothly. And knowing the response to the movement of the 6 DoF arm manipulator based on the data of the movement and speed of the stepper motor. The results of the stepper motor accuracy with the input of angular movement by the master controller get an average error of 0.287%.

# 1 Introduction

The development of robot technology is progressing rapidly because of the sophistication of its technology. Therefore, many heavy and risky jobs that were originally done by humans are now carried out by robots. To control the robot there is a robot movement control program that functions to control all the actuators it has with the purpose created according to its design. According to the “Robot Institute of America,” 1979, “A robot is defined as a reprogrammable, multifunctional manipulator designed to move material, parts, tools, or specialized devices through various programmed motions for the performance of a variety of tasks.”[1]

One of the tasks that can be part of the prevention of work accidents is remote control of robots in field conditions that are impossible for humans to do. If a robot can be implemented and replace humans in carrying out this task, the efficiency of preventing work accidents can be

minimized. In addition, the robot is also expected to be operated easily and simply.

The robot can be programmed to perform movements according to the reference obtained from the sensors it has, therefore we need a controller that is able to control the robot. The master controller is a robot controller pendant that is moved by hand according to the desired movement. In the actual implementation, the master controller is used to control the arm manipulator on the underwater robot or Underwater-Remotly Operated Vehicle (ROV).

The master controller has approximately 7 sensors attached to it, each of which controls an actuator located on the joint arm manipulator. On the master controller there are also many buttons that are intended for various features such as scaling movement, start, stop, translational movement, and others. [2]

# 2 RESEARCH METHODOLOGY

**2.1 Arduino UART Serial Communication**

In a controller, the data inputted by the user to move a robot will be sent by a data communication system. Generally, all Arduino boards have at least 1 serial port which is commonly known as the USART or UART (Universal Asynchronous Receiver-Transmitter) type.

The serial communication process uses two pins, namely the RX pin to receive data and the TX pin to transmit data. That's the reason why sometimes someone mentions the term Arduino RX TX communication. Usually the RX pin on Arduino is pin 0 and the TX pin is pin 1.

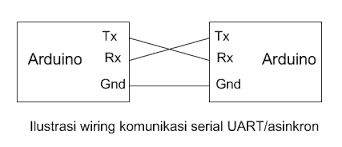


Figure 1 : UART serial communication wiring illustration

Data communication is done by connecting the Arduino master tx pin to the Arduino slave rx pin and the Arduino master rx pin to the Arduino slave tx pin. The picture is as in the image below:

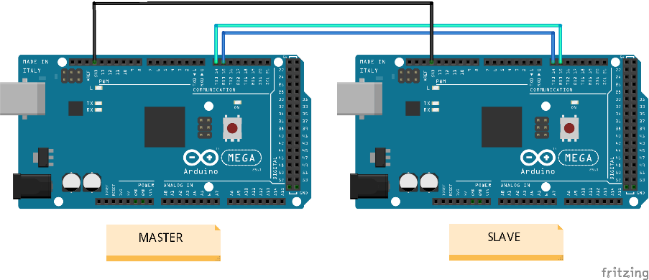


Figure 2 : Arduino serial data communication wiring overview

**2.2 Hardware Design**

The Master controller hardware design on the electrical part is built on various input components, while the input components consist of a potentiometer, joystick module, push button, and toggle switch. As for the output in the form of an LCD and a data that is sent serially. The data generated by the input component is then processed by the Arduino (master). Then the data will be displayed through the LCD as well as sent via UART serial communication.

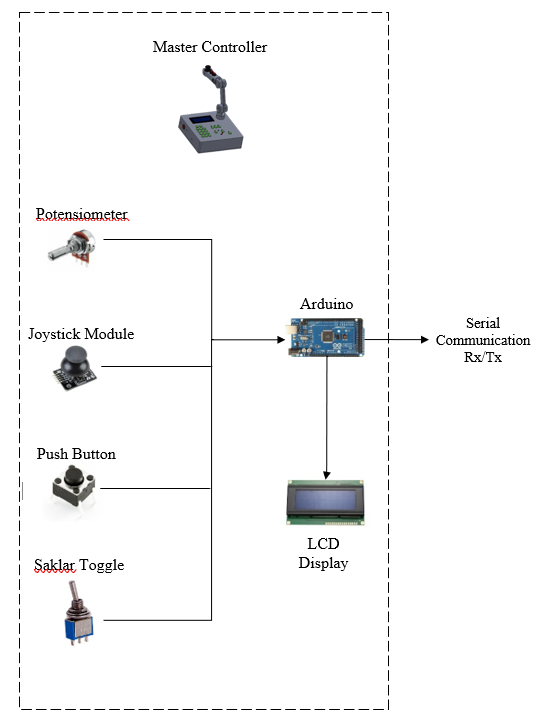


Figure 3 : System overview A

The data sent is in the form of angle data, this data will be received by Arduino (slave) and will be converted into digital pulses for then the motor driver will drive each stepper motor actuator located at each joint arm of the manipulator/robot arm.

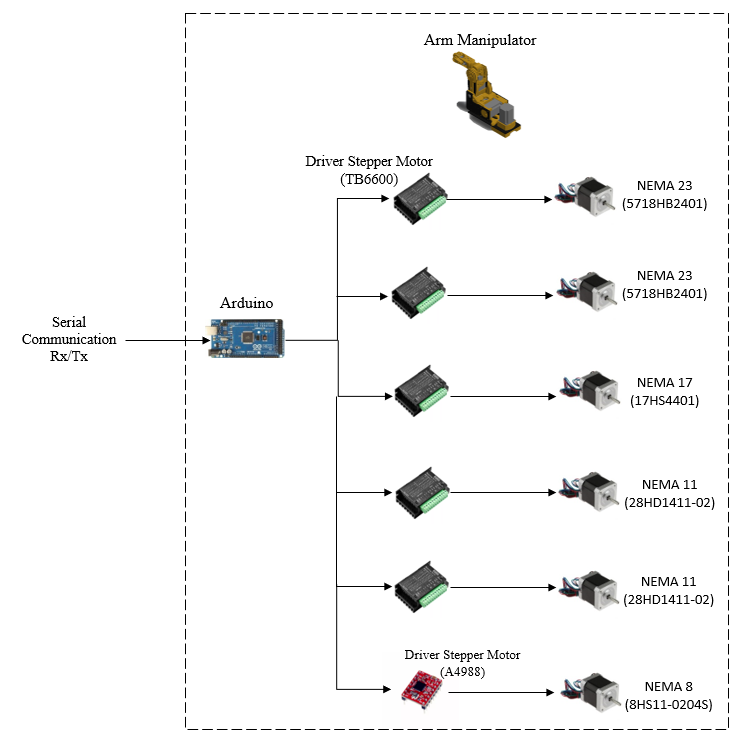


Figure 4 : System overview B

To be able to integrate all components, a mechanical design for making a master controller was made using Solidworks software. Physically there are two parts to the Hardware Master controller design, namely:

1. *Panel master*

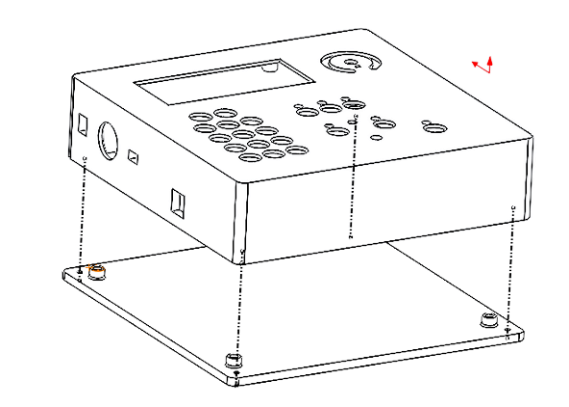
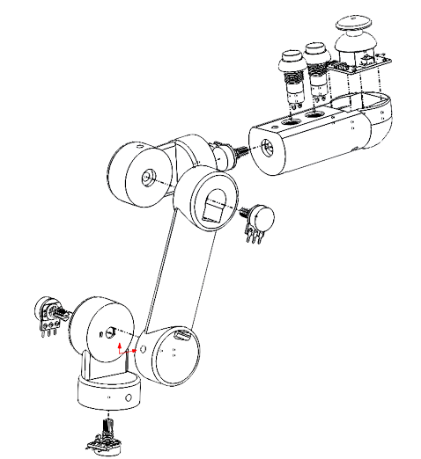




Figure 4 : Panel master Structure

The Master Panel functions as a casing panel, placing the display of information on the movement of the arm manipulator, and placing the push button keyboard. It is also a place for the installation of the entire electrical circuit and also the Arm master controller.

1. *Master Arm*



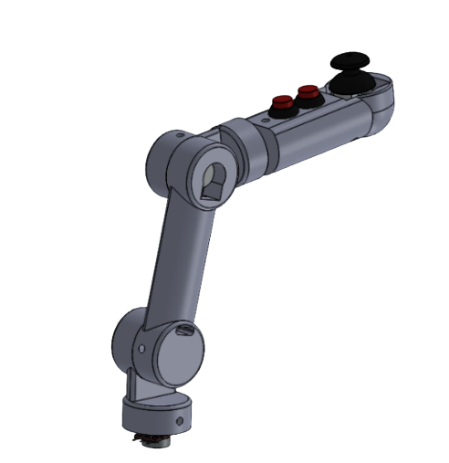


Figure 5 : Master Arm Structure

Master Arm functions as a robot arm manipulator controller in Joystick mode. This design is built by following the maximum resolution angle that can be provided by the linear potentiometer sensor, which is 270°, which means that each Join on the arm master can rotate 0 - 270°.

**2.2 Software Design**

Software design or Arduino program syntax, is the working principle of the tool that is displayed with a flowchart.

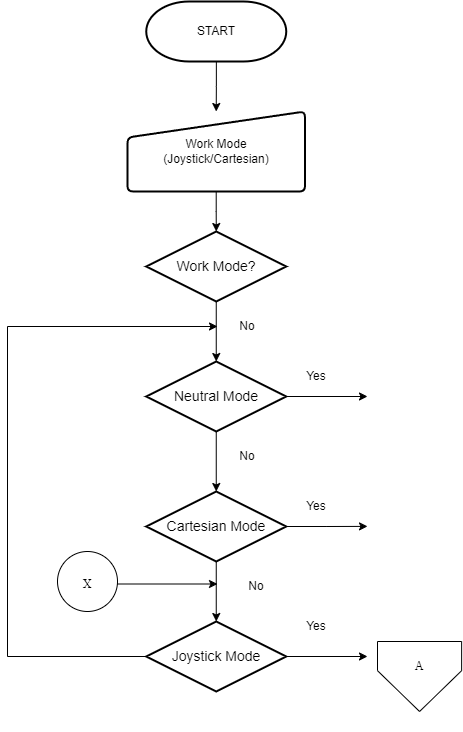


Figure 6 : General system flow

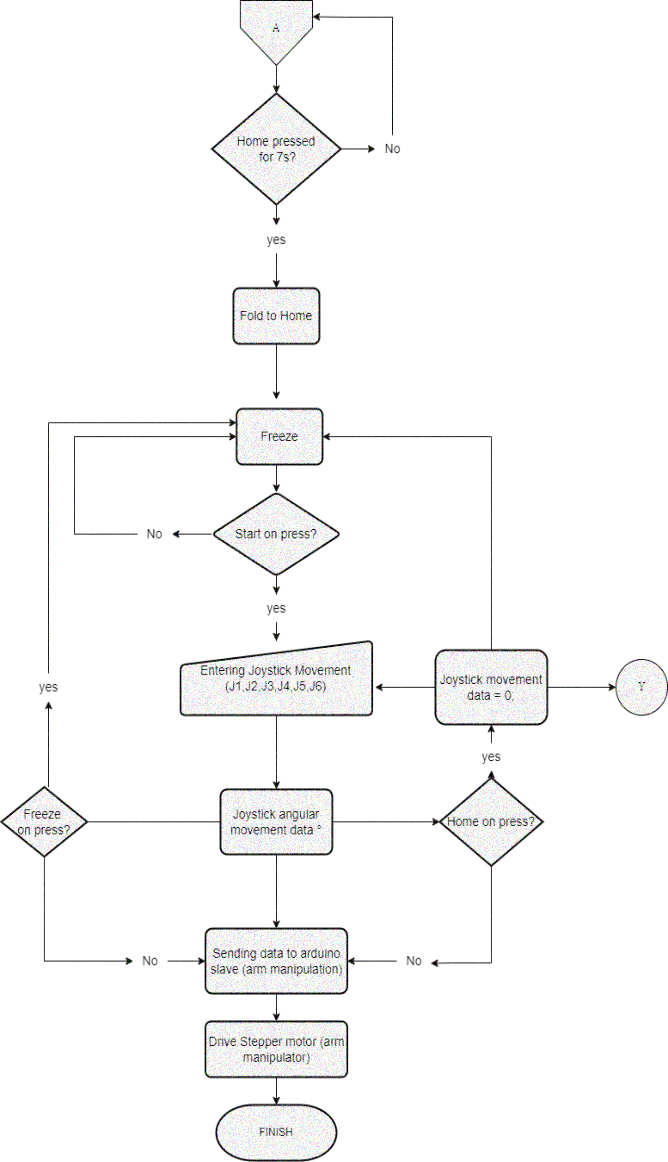


Figure 7 : General system flow A

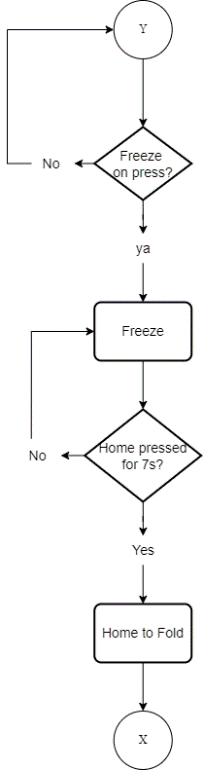


Figure 8 : General system flow Y

The diagrams in figure 6, figure 7 and figure 8. explain how the manipulator control system by the master controller in Joystick mode. The toggle switch must be set to select the joystick mode. The manipulator is initially in a state of folding. When the home button is pressed for 7 seconds, the robot arm will move to the steady/home position. To move the manipulator the start button needs to be pressed, so that the master arm will generate degree data according to the input from the operator. The data of the degree of each joint and the position of the end-effector of the robot are displayed on the available LCD display. If the freeze button is pressed, the master arm will not generate degree data and will not move the robot arm no matter how the master arm is moved by the operator. The robot will move to the home position or the position when all joints are 0 when the home button is pressed. When all joints are 0 and the home button is pressed for 7 seconds, the manipulator will move to the fold position.

**3 Result**

**3.1 Results of the Physical Form of the Tool**

Figure 9 : Master Controller

The results of this design produce a controller called the Master controller to control a manipulator at a certain distance with Arduino UART serial communication. The master controller is made to resemble the Master control product on the ROV robot sold in the market with the shape and features contained in it. The shape of the results of this design can be seen in the image below.

**3.1 Testing Master Controller Movement Input Against Joint Arm Manipulator**

Tests are carried out to see how the arm manipulator reacts to the input movement of each joint from the master controller and its accuracy at the specified angle. The value of the angle of movement will be inputted and the result will be measured using a manual measuring tool with the name of the actual angle. As for measuring the accuracy of the results of position readings, comparisons are made as follows:



Figure 10 : Formula to find an error

The test was repeated 4 times with various angle input values.

Table 1: Testing the input movement of the master controller against joint 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test | *Joint* | Master Controller input degree (°) | Actual  Degree (°) |  |
| Error (%) |
| 1 | 1 | 90 | 89.75 | 0.2778 |
| 2 | 45 | 45.15 | 0.333 |
| 3 | -30 | -30.5 | 1.6667 |
| 4 | -60 | -60 | 0 |
| Average of Error | | | | 0.5739667 |

Table 2: Testing the input movement of the master controller against joint 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test | *Joint* | Master Controller input degree (°) | Actual  Degree (°) |  |
| Error (%) |
| 1 | 2 | 90 | 89.75 | 0.2778 |
| 2 | 45 | 45.15 | 0.333 |
| 3 | -30 | -30.5 | 1.6667 |
| 4 | -60 | -60 | 0 |
| Average of Error | | | | 0.5739667 |

Table 3: Testing the input movement of the master controller against joint 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test | *Joint* | Master Controller input degree (°) | Actual  Degree (°) |  |
| Error (%) |
| 1 | 3 | 90 | 89.75 | 0.2778 |
| 2 | 45 | 45.15 | 0.333 |
| 3 | -30 | -30.5 | 1.6667 |
| 4 | -60 | -60 | 0 |
| Average of Error | | | | 0.5739667 |

Table 4: Testing the input movement of the master controller against joint 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test | *Joint* | Master Controller input degree (°) | Actual  Degree (°) |  |
| Error (%) |
| 1 | 4 | 90 | 89.75 | 0.2778 |
| 2 | 45 | 45.15 | 0.333 |
| 3 | -30 | -30.5 | 1.6667 |
| 4 | -60 | -60 | 0 |
| Average of Error | | | | 0.5739667 |

Table 5: Testing the input movement of the master controller against joint 5

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test | *Joint* | Master Controller input degree (°) | Actual  Degree (°) |  |
| Error (%) |
| 1 | 5 | 90 | 89.75 | 0.2778 |
| 2 | 45 | 45.15 | 0.333 |
| 3 | -30 | -30.5 | 1.6667 |
| 4 | -60 | -60 | 0 |
| Average of Error | | | | 0.5739667 |

Table 6: Testing the input movement of the master controller against joint 6

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test | *Joint* | Master Controller input degree (°) | Actual  Degree (°) |  |
| Error (%) |
| 1 | 6 | 90 | 89.75 | 0.2778 |
| 2 | 45 | 45.15 | 0.333 |
| 3 | -30 | -30.5 | 1.6667 |
| 4 | -60 | -60 | 0 |
| Average of Error | | | | 0.5739667 |

Table 7: Average error at each joint

|  |  |
| --- | --- |
| **Joint** | **Average Error** |
| 1 | 0.5739667 |
| 2 | 0.333275 |
| 3 | 0.4804 |
| 4 | 0.097275 |
| 5 | 0.16675 |
| 6 | 0.0754 |
| **Average Overall Error** | 0.2878445 % |

Based on the tests that have been carried out, the results show that the accuracy of the stepper motor based on the angular movement input by the given master controller is very good. Seen in the percent error obtained, each joint gets an error only in the range of zero point a few percent. Moreover, after being averaged, the overall joint error is 0.287%.

**3.2 Respond Test**

Response testing is done by providing input movement data from the master controller in the form of repetitive motion with the parameters of the delay time for sending serial data and the speed of the stepper motor. The results will be analyzed in the form of the number of movements executed and the response time to the movements.

Explanation of parameters in the table:

• Repeated movement data, is the Joystick master controller movement data in the form of one full movement in one joint.

• Serial data delay, is the delay time for each data transmission in units of time (ms).

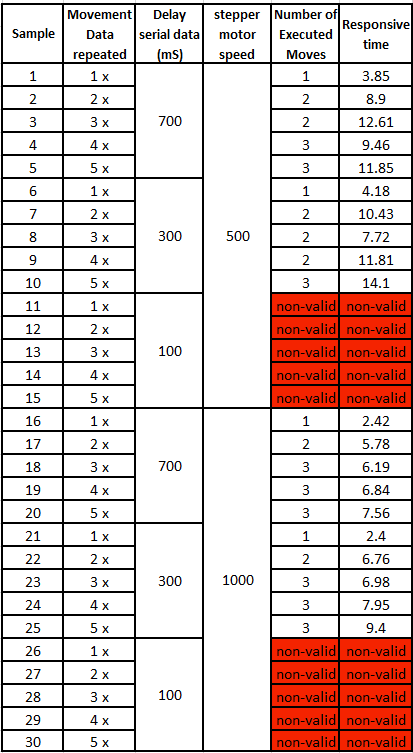
• The speed of the stepper motor, is the speed that is input using the function from the accelstepper() library

• The number of moves executed, is the movement executed by the arm manipulator against the given input data.

• Responsive time, is the amount of time obtained from movements executed by the arm manipulator.

1. Respond Test Table

Table 8: Respond Test Table



In the tests carried out, the results showed that the arm manipulator did not have a good response to data transmission by the master controller. Seen in the three motor speeds that are inputted, the movement data that is successfully executed only reaches 3x repetitive motion input, where the maximum movement data can only be executed three times repeatedly at a constant increase time.

From the data above, the arm manipulator cannot execute the incoming data input at a delay of 100ms of data transmission, then there is an increase in the responsiveness of movement at a delay of 300ms of data transmission. This is because the smaller the delay time, the more data will be sent and the faster the data will be sent, so the arm manipulator receives a lot of data but cannot execute it properly. Seen in the test above, the best data delay time is 700ms.

**4 Conclusions**

The conclusions of this study refer to several formulations of the problems that have been achieved.

1. In the design, there are two modes of movement, namely Joystick mode and Cartesian mode. In this final project, the master controller is able to control the 6 DoF arm manipulator with Joystick mode. With features that can be applied to the Master controller in this final project, namely Homing position, Fold position, Start and Freeze.

2. The communication used on the Master controller is using UART serial communication with rx/tx pins. 3. The arm manipulator does not respond well to data transmission by the master controller, one of the reasons is the selection of Arduino Mega as the controller. However, when referring to ROV, a fast response is not a priority because the operator can control the speed of the movement on a master controller.

4. The accuracy of the stepper motor applied to the 6 DoF arm manipulator for the given master controller movement input shows that the stepper motor has an accuracy with an average error of 0.287% for all joints.

**References**

1. Spong, M. W., Hutchinson, S., dan Vidyasagar, M.,”Robot Modeling and Control,” John Wiley & Sons, First Edition, ISBN-100-471-649, 2005.
2. Robert D. Christ dan Robert L. Wernli Sr. *The ROV Manual (A User Guide for Remotely Operated Vehicle Second Edition*). Amerika: Elsevier Ltd. 2014.

1.  https://orcid.org/0000-0000-0000-0000 [↑](#footnote-ref-2)
2.  https://orcid.org/0000-0000-0000-0000 [↑](#footnote-ref-3)
3.  https://orcid.org/0000-0000-0000-0000 [↑](#footnote-ref-4)