VOL. 21, NO. 1, 2022, 42-47
www.fke.utm.my/elektrika
ISSN 0128-4428

ELEKTRIKA
Journal of Electrical Engineering

Reverse Engineer 5 Degrees of Freedom Robot Arm using Programmable Logic Controller

Nurul Huda Abd Rahman[1*], **Zulfakar Aspar**[1], **Abdul-Malik H. Y. Saad**[1], **Izam Kamisian**[1] and **Ahmad Bukhari Abdullah**[2]

[1]School of Electrical Engineering, Faculty of Engineering, Universiti Teknologi Malaysia, 81310 UTM Skudai, Johor, Malaysia.
[2]Wise Path Sdn. Bhd., Taman Tampoi Utama, 81100 Johor Bahru, Johor, Malaysia.

[*]Corresponding author: nhuda57@graduate.utm.my

Abstract: Many types of robot arms have been developed in the market. Whenever a robot arm broke down, it is a challenge to do reverse engineer and repair the robot arm. This project has successfully reused the existing 5 degrees of freedom robot arm structure and electronic components while replacing the controller with a new Programmable Logic Controller (PLC) and LLD model. The project uses the existing DC servo motor, encoder, sensors, power supply, and interface board. Since the structure and most electronic components are already available, they are important to be analyzed and understood before they can be reused. Hence the reverse engineering activities are needed. In addition, a low-cost PLC is used to save cost. Thus, it is not possible to move all axes at the same time. Instead, the robot arm can only move one axis at a time. Additionally, the robot arm has been improved to be able to be programmed manually or automatically and repeat previously predetermined position sequences. In total, 90 rungs, 96 built-in commands, and custom functions have been used. The refurbished system is very stable and reliable.

Keywords: Robot Arm, PLC, microcontroller, Modbus.

Article History: received 8 February 2022; accepted 1 April 2022; published 20 April 2022.

1. INTRODUCTION

Robot arms have been used extensively in the industry to do specific works, avoid hazardous environments, produce consistent work, and much more. For instance, they are used in education, painting, car assembly, soldering, wire bonding, palm oil harvester [7] and etc. Robots can be categorized based on their structural properties which are Linear Robots (including Cartesian and gantry), Cylindrical Robots, Parallel Robots, Spherical Robots, SCARA Robots, and Articulated Robots [9]. A robotic arm can be said to be a typical example of an articulated robot. An important matter which should be considered is that the dimension of the configuration space increases with the number of joints. However, the operation speed is limited due to the different payloads at the manipulator and nonlinear environment as in [2] and [3].

A robot arm has the advantage to reach above or below obstacles and has the largest work area for the least floor space. However, a robot arm is difficult to program off-line as it has two or more ways to reach a point [2]. A pick and place robot arm has fixed axes such as using pneumatic to get high speed, and motor for higher torque. However, the freedom movement robot arm is more difficult due to more complex data to be read, processed, and updated at the same time. Many methods can be used to program a robot such as using a microprocessor or microcontroller in [2], [3], and [9], 5 DoF simulation in [4] and [5], PLC [10], FPGA [11], and [12], or direct from a PC [1]. Each type of implementation has its advantages and disadvantages such as total system cost, development time, size, power, and axis movement speed.

Details on the concept of reverse engineering works. And why it is important. If a developer going to design a new research, the developer will set the objectives, set the specification of what the system wants to achieve, identify what components with respective quality must be used, build the structure of the system, integrate a controller, develop the program, and run the test and verification. Finally, publish the result.

In reverse engineer, the system already exists. A developer cannot simply choose what components to use. The developer must use existing components with the existing and limited quality or specification. In many cases, there are no schematic available to assist the analysis activities. So, a new schematic should be drawn. But, the developer must understand the available capability, resolution, limitation, operational resources (controller) and etc. In other words, the developer has to work with what the developer have rather than what the developer want. And the general engineering concept is now in reverse order. Hence the name is reverse engineering.

The developer is also limited by the existing system architecture. For example, automotive engineers can design a new car from scratch. But, they are not

necessarily can do the troubleshooting and repairs.

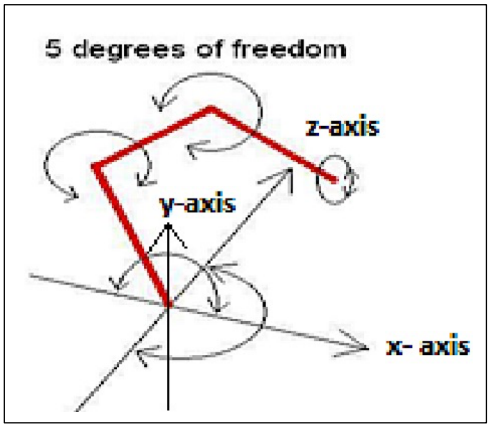


Figure 1. Position of servo motors and their rotation angles in 5 DoF robot arm [1]

In this research, Figure 1 shows the geometry representation of a 5 degrees of freedom (DOF) robot arm [1]. The 5 DOF robot arm consisting of the base, shoulder, elbow, pitch, and roll was reverse engineered and repaired using a Programmable Logic Controller or PLC. In addition, an ON and OFF gripper is also available to hold an object. The original controller was custom-made using a microprocessor-based system. Since it is impossible and not viable to hack the program inside the microprocessor, the controller is being replaced with a medium to low-cost PLC.

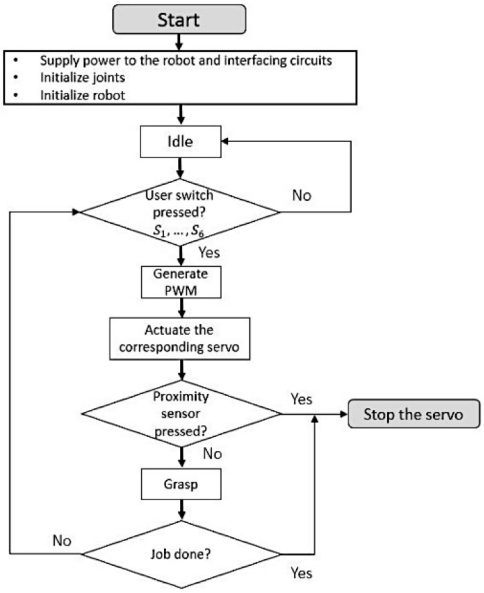


Figure 2. A general flowchart for controlling an axis [2]

A general idea to control an axis is by observing the motor movement as shown in Figure 2 [2]. The other axes which have similar DC motors will follow the same process flow. This also means if there are five axes, five flows must be observed at the same time, and an appropriate response must be given at the same time. The controller also has to do concurrent processing, which leads to the consumption of more processing power. However, too much concurrent processing is not possible when using a low-cost PLC due to the limitation of the PLC cyclic scan speed.

2. REVERSE ENGINEER AND REPAIR MECHANISM

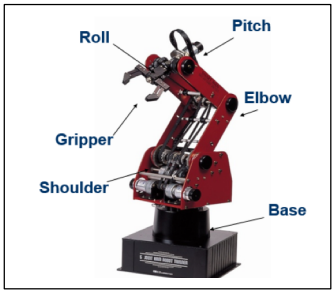
To professionally reverse engineer and repair a robot arm, the following procedures could be followed. This is the basic SOP (Standard Operating Procedure in reverse order of a normal engineering development.

1. Understand the robot arm structure: All design works are limited by the available structure. Instead of simulation to get the desired design, a hands-on approach and observation are required.
2. List all the current sensors, actuators and interfaces involved: The performance of the structure is limited by the sensors, actuators, interface, and controller being used.
3. Model the controller: Find the relationship between the robot arm structure with sensors and actuators. This involves a lot of trial errors to identify the manual operation without activating the available controller.
4. List the existing functions available when the robot arm is still working. This is based on the user's experience or requirement which is naturally limited by the hardware structure capability.
5. List any limitations or weaknesses of the current functions: Some of the limitations are known by the current users while other limitations can be identified by analyzing the structure, sensors, and actuators.
6. Is there anything that can be improved? By human nature, users want to do a lot of improvements such as increasing the speed, power, or range of operation or adding software on a computer to add more features.
7. Contingency plan: What to do if one of the steps cannot be done immediately?
   a) Replace part of the structure if necessary
   b) Build a new GUI and database if necessary
   c) Some of the sensors and actuators may not work properly. Replaced or build a subroutine to do error correction to the best outcome.
8. How to ensure all the objectives are achievable? There is no single model that can solve all problems. Be prepared to create alternative models if the current one does not satisfy the objectives. Based on budget, time, and equipment, finds the best optimum solution.

3. ROBOT STRUCTURE

Although a conclusion may review the main points of the paper, do not replicate the abstract as the conclusion. A conclusion might elaborate on the importance of the work or suggest applications and extensions.

The actual 5 DOF robot arm with its main components' movement specification is shown in Figure 3. Each axis is controlled by a DC servo motor and an encoder. Since the position of the servo motor cannot be tracked automatically by the encoder, the value of each encoder must be stored in the memory to keep track of the servo motor movement.



Component	Degree of movement	Encoder reading
Base	310-degree movement	from -60 to 60
Shoulder	+130 degree movement	from 0 to 440
Elbow	+130 degree movement	from -420 to 520
Pitch	+130 Degree movement	from 0 to 250
Roll	360-degree movement	from 0 to 300
Gripper	only open or close, 55mm wide	

Figure 3. The actual 5 DOF robot arm on the left and movement specification on the right

In theory, each axis should be able to move on its own axis. However, a limiter was forced upon the robot arm between the shoulder and elbow. Both axes can only be at a maximum of 180 degrees or horizontal lines to each other. The pitch can be lower than 90 degrees to the shoulder, but it cannot be more than 180 degrees. Hence, if the pitch wants to move the elbow higher, the shoulder must be moved first. The solution is provided in the LLD Model section. The roll has no encoder. Instead, a timer is used to ensure the roll axis move to the specific position in a controlled manner.

Development time to do the reverse engineering and repair was critical in the project. Hence, the team would not have sufficient time to build a proven and stable printed circuit board and interfaces. As a result, a PLC was chosen in this project due to its stability, robustness, and readily available all the necessary interfacing which are digital inputs, digital outputs, high-speed counter pulse inputs, voltage level interface, and communication to a PC.

3. LADDER LOGIC DIAGRAM (LLD) MODEL

The main controller was developed using a Triangle Logic PLC model T100MD1616+ with a compiler version known as TL41. The PLC contains 16 inputs and 16 outputs, which are all used in this project, and four unused ADC channels. In addition to the internal counter, the PLC also has two dedicated high-speed counters which are connected to the encoders. Since there are five encoders to be monitored, one of the high-speed counters is shared with two axes while the other high-speed counter is used for three axes. The sharing is possible by storing the encoder data in temporary storage or memory in the PLC. These activities are done in several custom functions written in BASIC language as in Figures 4 and 5. The custom function is triggered by the input sensors which are mimicked by the rung's input network. The summary of the LLD model is shown in the result section.

```
Custom function #3:
DM[7]=HSCPV[2]+DM[7]; Get Shoulder down current coordinate, save in data memory [7].
Custom function #4:
DM[8]=HSCPV[]+DM[8]; Get Shoulder up current coordinate, save in data memory [8].
Custom function #5:
DM[9]=DM[7]-DM[8]; Get the actual coordinate for the shoulder axis.
```

Figure 4. The shoulder exact location is using high-speed counter no. 2

```
Custom function #15:
DM[1]= DM[1]-HSCPV[2]; Get Roll forward current coordinate, save in data memory [1].
Custom function #16:
DM[2]= DM[2]-HSCPV[2]; Get Roll reverse current coordinate, save in data memory [2].
Custom function #17:
DM[12]=DM[1]-DM[2]; Get the actual coordinate for roll axis.DM[9]=DM[7]-DM[8]; Get the actual coordinate for the shoulder axis.
```

Figure 5. Shoulder and roll axes are sharing the same high-speed counter 2

This method can ensure that each axis can be traced all the time. However, the memory read and write are slow processes. If all axes want to move at the same time, the data in the memory may be skipped due to insufficient cyclic scan time to do the data movement. The Triangle Logic PLC model T100MD1616+ cyclic scan is 2ms [8]. Each step takes 10 us to execute. For a complex application such as in this project, the cyclic scan is too slow especially after adding the 96 custom functions which involve extensive data processing.

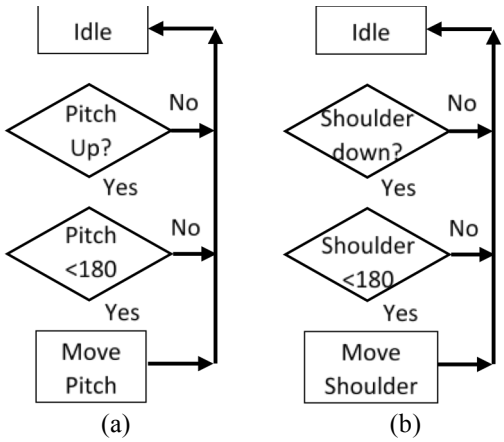


Figure 6. Flowchart to limit the pitch to shoulder angle, (a) when the pitch needs to move up and (b) down

To solve this problem, only one axis can be moved at a time. The weaknesses found in this project also lead to the research team's detailed study on PLC architecture which resulted in [6] where a high-performance processor was invented to solve the problem. To solve the limiter between the shoulder and elbow, the flowchart in Figure 6. (a) is used whenever the pitch is to be moved up while Figure 6. (b) is used whenever the shoulder is to move down. In actual, the flowchart is more complicated during automatic mode where the movement of both axes is done simultaneously. Hence, the automatic model must synchronize both models (Figures 6. (a) and (b)) at the same time while reaching the target position. For a more advanced implementation, Signal Interpreted Petri Net modeling can be used as in [13].

4. DEVELOPED CONTROL SYSTEM

The power supply, PLC, and interface are assembled inside the main controller box. The PLC is the main controller in this project. The controller box is under the robot arm base as shown in Figure 3. It has limited functions which are Power: Power On/Off, Home: Home position, and Play: Play the current automatic position which doesn't need to go through a computer. The Power sources are separated from the PLC and driver of DC servo motors. The communication between the PLC, the pendent, and a PC is done via RS 485 communication lines. The Modbus protocol is used during the data and command exchange. Both the pendent and PC can give commands to the PLC. However, the pendent must release the control to the PC for the PC to become the main operator.

5. PENDENT CONTROLLER

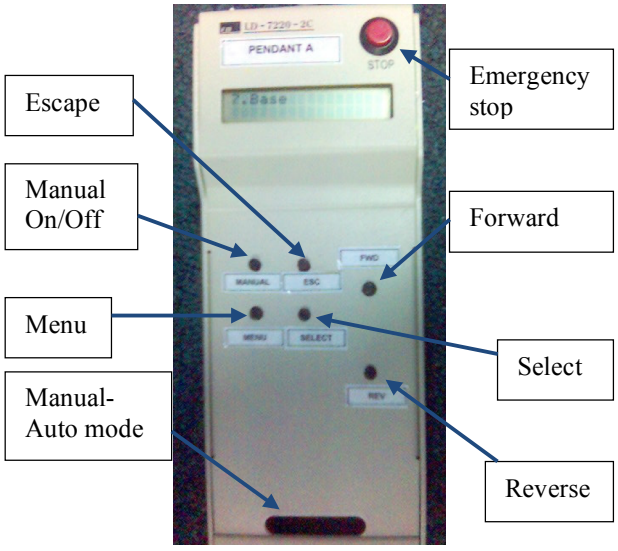


Figure 7. The pendent to control the robot arm by the operator

A pendent in Figure 7 is needed so that an operator can immediately control the robot arm without the need to go through the PC. A Microchip PIC 16F8XX microcontroller was used to build the pendent. An LCD is available to see the menu is being chosen. All controlling are done by choosing the buttons available in Figure 7. Before starting a pendent of the robot, users must change the "Manual – Auto" switch to manual mode. This will indicate the robot to run in manual mode.

Press the "manual on/off" button to activate the pendent. After that, press the "menu" button, and the LCD screen will show several modes that the user wishes to play. The modes are 1) Home, 2) Gripper, 3) Roll, 4) Pitch, 5) Elbow, 6) Shoulder, 7) Base. Use the forward/reverse button to choose the mode users wish to go. Press the "select" button to select mode. For example, if gripper mode is selected, the LCD display will be shown "gripper ready". This tells users that the mode is ready to use. Then users can use the forward and reverse buttons to go to the place they wanted. Coordinates will show in the program ROBOT_ARM. After finishing with the mode, the user can press the "escape" button to back to the previous status.

6. DEVELOPED GUI

For a more user-friendly operation, a PC is needed. The movement of each component can be controlled with the developed Graphical User Interface (GUI) shown in Figure 8 which must fulfill the robot arm movement for each component mentioned in the Robot Structure section. Since the servo motor and encoder have a high precision up to +0.5mm, the encoder reading value is used in the GUI as in Figure 8 instead of degree of movement so that the user can calibrate the angle on their own. The Gripper can only open or close. The maximum gap is 55mm.

Manual mode is also available to be used to move the robot to a specified coordinate manually. The Get data menu is used to get the current position of the axes and store them in the computer. Manually, the Insert menu can be used to insert the entire coordinates into the provided columns. The first column indicates the 1st coordinate that the robot will achieve, while the second column indicates the 2nd coordinate that the robot next will reach. In the process field, the value means that how many processes users want to repeat. For example, if process=1; the process will be executed for the 1st coordinate and then for the 2nd coordinate. When the robot finishes the process, the robot will automatically back to the Home position. If Process=2; the previous process is repeated two times before the robot will back to the home position.

Every time data (coordinates) had changed, users need to compile them first before sending them to Robot's controller. If the compilation process is completed without any error being found, the send button will appear (enable). Double click on the Send button to start downloading data from PC to controller. When the downloading process is completed, a popup message will be shown.

Figure 8. The developed GUI in the computer

When the popup message has disappeared, users can give a command to the robot controller to commence the routines. Users can double-click on the Play button to send the command to the robot controller. Within seconds, the robot routines will begin. In contrast, auto mode is used to move the robot arm to two positions from the home position automatically; the minimum is a single move, while the maximum is 9999 repetitions of sequence moves.

7. RESULTS

Since the number of Timer, Counter, and high-speed counter are limited in the PLC, memory storage was used to store the data temporarily so that the same modules can be used for different axes as in Figures 4 and 5. In total, 60 memory words were used for data calculation and manipulation purposes. The summary of the LLD model is in Table 1.

Built-in functions are functions that were built by the manufacturer which are RSctrl, Upctrl, MaRST, Clk, DIFU, Sequencer, and others. They are used together in the Custom Functions. The Custom Functions are the functions that are built by the PLC programmer. The BASIC language is used to build the custom functions. The signed Integer being used in this project are a, f, V, W, X, Y, and Z. Figure 9 shows a snapshot of the LLD model. The input sensor network is complicated to keep track of the position of pitch and to trigger the correct custom functions in the rung at the right timing.

The function in Figure 10 is used to set the pitch forward limit. High-Speed Counter 1 will count until value DM [11]-270, custom function #97 execute and stop the pitch forward instantly. The Pitch limit switch will be the pitch reverse limit. At the end of the project, the robot arm has been improved to be able to be programmed manually or repeat previously predetermined position sequences.

Table 1. LLD model resource utilization

No.	Resource	Quantity
a)	Number of rungs	90
b)	Number of custom function	96 (from 1 to 52 lines BASIC language programming)
c)	The built-in function being used:	6
d)	Digital Input	16
e)	Digital Output	16
f)	Relays	118
g)	Counter and comparator	24
h)	ADC being used	0
i)	Timer	18
j)	High-speed counter	2 (to read digital encoder)
k)	Rungs without any function	36
l)	Fixed integer	7
m)	Memory store	60 words
n)	Sharing Highspeed counter 1	elbow, pitch
o)	Sharing Highspeed counter 2	shoulder, roll forward, and reverse

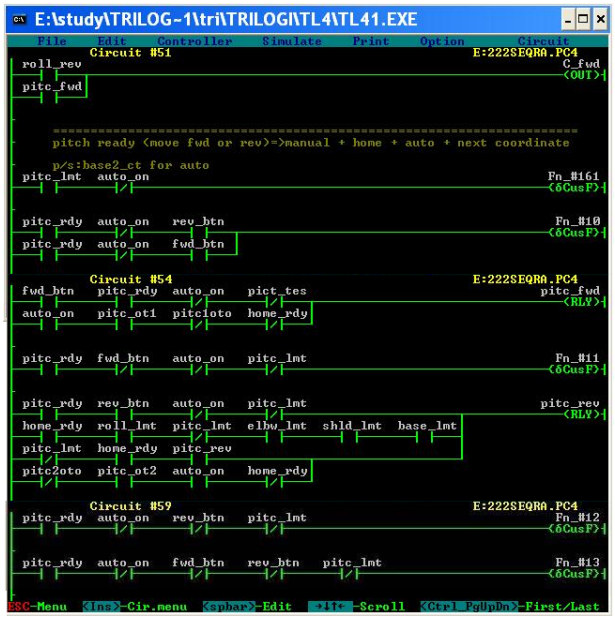


Figure 9. A rung example from 90 rungs

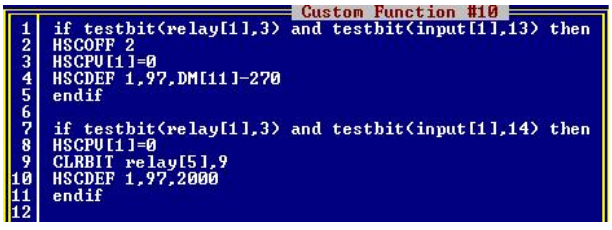


Figure 10. Custom Function #10 from 96 functions

Initial work tried to move all the axes at the same time but to no avail. This is due to the cyclic scan speed of the

PLC being used is too slow to monitor all the axes at the same time. Hence, the final developed PLC-based model chose to move one axis at a time. During verification, it was realized that there are two axes to be moved and tested at the same time which are the shoulder and pitch. This is due to a limiter being put in the original structure in such a way that it is not possible to have the pitch above the horizontal line of the shoulder. Hence, the program must always check whether the pitch is below the horizontal line of the shoulder before any movement of the pitch is allowed. Therefore the general servo motor and its position monitoring in Figure 2 must be modified as in Figure 6.

A software application as in Figure 8 was built on a PC to store the robot user requirement. This project chose to develop two different target movements on each axis which are the initial position and the target position. A user should also be able to control the movement manually by keying in the required destination. Once the user is satisfied with the target position, the software can record the position for future usage. Then, the position record can be saved in a file.

8. CONCLUSION

It is possible to use a medium or low-end PLC to control a robot arm's freedom movement. Although the movement is limited to one axis at a time, the system is very stable and reliable as inherited from any PLC-based design. The use of GUI on a PC makes the system more user-friendly with the option to manually control the robot arm or automatically play back the recorded positions.

ACKNOWLEDGMENT

This project is a collaboration with Wise Path Sdn. Bhd. and Atama Tech Sdn. Bhd. The headquarters of Wise Path Sdn. Bhd. was used to do all the works in two months. The researchers would also like to thank the Research Group of VECAD of School of Engineering, Faculty of Engineering, Universiti Teknologi Malaysia.

Researchers also like to thank Universiti Teknologi Malaysia for grant UTMER 20J87 as continuation for project high performance PLC and PLC modeling using Signal Interpreted Petri Net.

REFERENCES

[1] Nimisha Limaye and Siddharth Verma, "5 Degrees of Freedom Robotic Arm controlled using PS/2 Mouse", *International Journal of Engineering and Technical Research (IJETR)*, November 2014, ISSN: 2321-0869, Volume-2, Issue-11).

[2] Arian Faravar, "Design, Implementation, and Control of a Robotic Arm Using PIC 16F877A Microcontroller", Master Thesis, Dept. Computer Engineering, Eastern Mediterranean University February 2014, Gazimağusa, North Cyprus (Eastern Mediterranean University, Gazima, North Cyprus, 2014.

[3] Kabuka, R. M., Glaskowsky and N. P., Miranda, J., "Microcontroller-Based Architecture for Control of a Six Joints Robot Arm", *IEEE Transactions on Industrial Electronics*, 1988, pp. 217-221.

[4] Elias Eliot, Deepak B.B.V.L., Parhi D.R., and Srinivas J., "Design & Kinematic Analysis of an Articulated Robotic Manipulator (International Journal of Mechanical and Industrial Engineering", 2012.

[5] Amin A. Mohammed and M. Sunar, "Kinematics modeling of a 4-DOF robotic arm", *International Conference on Control, Automation and Robotics, IEEE Conference*, Singapore, 2015.

[6] Z Aspar and M Khalil-Hanim, "Modeling of a ladder logic processor for high-performance programmable logic controller", *Third Asia International Conference on Modelling & Simulation*, Indonesia, 2009, pp. 572-577.

[7] Mohd Hudzari Razali, Muhammad Aliuddin Bakar, and Muhammad Syukri Mohd Sabir, "Conceptual Development of Automated Harvester for Tall Oil Palm Tree", *Advances In Agricultural And Food Research Journal (AAFRJ)*, Malaysian Society of Agriculture and Food Engineers, 2020.

[8] Triangle International Research, "T100MD1616_Product_Info_Sheet", http://www.tri-plc.com/t100md1616.htm accessed on 2 Oct. 2021.

[9] Oridate Ademola Ayodeji, Design, "Simulation and Fabrication Of A 5-DoF Robotic Arm (With Implementation Of Inverse Kinematics", Bachelor Of Science Degree Thesis, Dept. Mechanical Engineering, Obafemi Awolowo University Ile-Ife, Nigeria, 2014.

[10] Daniel Kajzr, Leoš Beran and Václav Záda, "Development of a robotic arm suitable for demonstration of advanced control methods", *21st International Conference on System Theory, Control and Computing (ICSTCC 2017)*, IEEE, 2017.

[11] Cong Han, Hongbin Ma, Wenchao Zuo, Sunjie Chen and Xinghong Zhang, "A general 6-DOF industrial robot arm control system based on Linux and FPGA", *Chinese Control And Decision Conference (CCDC 2018*, IEEE, 2018.

[12] Xingqiang He, Zhengdong Wang, Haitao Fang, Kai He and Ruxu Du, "An embedded robot controller based on ARM and FPGA", *4th IEEE International Conference on Information Science and Technology*, IEEE, 2014.

[13] Zulfakar Aspar, Nasir Shaikh-Husin and M Khalil-Hani, "Algorithm to Convert Signal Interpreted Petri Net models to Programmable Logic Controller Ladder Logic Diagram Models", *Indonesian Journal of Electrical Engineering and Computer Science*, 2018.