SUPERVISOR ENDORSEMENT

"I hereby declare that I have read through this report entitle "Petri Net Controller Design For Robotic Pick and Place System" and found that it has comply the partial fulfilment for awarding the degree of Bachelor of Electrical Engineering (Control, Instrumentation, and Automation)".

Signature : ……………………………………..

Supervisor's Name : DR. SAIFULZA BIN ALWI

Date : ……………………………..

PETRI NET CONTROLLER DESIGN FOR ROBOTIC PICK AND PLACE SYSTEM

MATHIALAGAN A/L VENGADESON

A report submitted in partial fulfilment of the requirements for the degree of

Bachelor in Electrical Engineering (Control, Instrumentation, and Automation)

Faculty of Electrical Engineering

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

JUNE 2016

DECLARATION

I declare that this report entitle “Petri Net Controller Design for Robotic Pick and Place System**”** is the result of my own research except as cited in the references. The report has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Signature : ....

Name : MATHIALAGAN A/L VENGADESON

Date : ……………………………………….

DEDICATION

To my beloved mother and father

ACKNOWLEDGEMENT

First of all, I would like to thank almighty God for the strength and blessings. I would like to express my deepest gratitude to my supervisor, DR. SAIFULZA BIN ALWI for guiding me throughout my final year project development. Knowledge and extra input given by them has highly motivated me to successfully complete this project. The information, suggestions and ideas given by them played huge role in developing a fully functional project.

I would also like to thank my parents for their moral support. Their support gives me the strength to endure this final year project and successfully complete it. Finally, I am also grateful and would like to thank my friends who had helped me directly or indirectly and their help had given me a lot of ideas on troubleshooting the problems that rises during the development of the project.

ABSTRACT

Design methods for sequence controllers are given priority in advancing industrial automation. The expanding unpredictability and fluctuating needs of modern discrete manufacturing system have tested the design methods such as the use of ladder logic diagrams (LLD's) for programmable logic controllers. Traditionally, LLD has lack flexibility, extremely complex to understand the diagram and less adaptability to the current changes. Even for the real time purpose, if there is any problem occurs in the system, it is hard to detect the cause of the problem via LLD. The analysis which has been made shows that robotic controllers have received high attention by academic researchers and industrial engineers in order to design flexible, reusable, and maintainable control software for robotic system. To be more specific, Petri net (PN) are becoming as a very important tool to produce an integrated solution for modelling, analysis, simulation, and control of robotic system. Petri Net is a collection of directed arcs connecting places and transitions. Places may hold tokens. This project identifies the basic operation of the pick and place robotic system, analyse PN simulation for logic control system, model pick and place robotic system using Boolean function and design controller for pick and place robotic system by using PN. The comparison between the LLD and PN made to show the difference in operation and behaviour. The LLD controller which is using currently unable to control the sequence alone, it requires KUKA programming language in order to control the sequence efficiently. The KUKA program for pick and place operation is designed. The control goal is to enforce a set of linear constraints on the marking behaviour or state of the Petri net. The result includes a design and techniques used to satisfy the controller properties such as liveness, reachability, reversibility, safeness and boundedness. The results that presented in this paper will help a) further implementation of PN based controllers for other machines and robots in industries, and b) convince researchers, industrial engineers and robot manufacturers that PN is an effective and worthy controller which is applicable in their industries.

ABSTRAK

Kaedah reka bentuk untuk pengawal urutan diberi keutamaan dalam memajukan industri automasi. Keperluan sistem pembuatan diskret moden telah menguji kaedah rekabentuk seperti penggunaan gambarajah mantik (LLD’s) tangga untuk alat-alat kawalan logik boleh aturcara. Secara tradisinya, LLD mempunyai kekurangan fleksibiliti, yang amat kompleks untuk memahami rajah dan kurang keupayaan menyesuaikan diri kepada perubahan semasa. Jika ada apa-apa masalah berlaku dalam sistem, ia adalah sukar untuk mengesan punca masalah melalui LLD. Jurutera untuk merekabentuk perisian kawalan yang fleksibel, boleh diguna semula dan maintainable robot sistem. Lebih spesifik, Petri Net (PN) menjadi sebagai satu alat yang amat penting untuk menghasilkan penyelesaian bersepadu bagi permodelan, analisis, simulasi, dan mengawal sistem robot. Projek ini operasi asas sistem robot memilih dan tempat mengenal pasti, menganalisis PN simulasi sistem kawalan logik, model memilih dan meletakkan sistem robotik dengan menggunakan Boolean pengawal fungsi dan reka bentuk untuk memilih dan menempatkan robot sistem dengan menggunakan PN. Perbandingan antara LLD dan PN dibuat untuk menunjukkan perbezaan dalam operasi dan tingkah laku. Pengawal LLD yang kini mampu mengawal jujukan yang semata-mata, ia memerlukan bahasa pengaturcaraan KUKA bagi mengawal jujukan yang cekap. Program KUKA untuk memilih dan tempat operasi direka. Matlamat kawalan adalah untuk menguatkuasakan suatu set kekangan linear pada tingkah-laku penandaan atau keadaan Petri Net. Hasil termasuk Reka bentuk dan teknik-teknik yang digunakan untuk memenuhi sifat-sifat pengawal seperti liveness, reachability, keterbalikan, safeness dan boundedness. Keputusan yang dibentangkan dalam kertas ini akan membantu a) Pelaksanaan PN berasaskan pengawal untuk mesin dan robot dalam industri-industri lain pada masa depan, dan b) Meyakinkan penyelidik, jurutera perindustrian dan pengeluar robot PN itu pengawal yang cekap dan layak yang terpakai dalam industri.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

In this chapter, project background, problem statement, project overview, objective and the scope of the project will be presented.

1.1 Project background

The logic control system is operate by turning on off motors, switches, valves, and other devices in response to operating conditions and as a function of time. Basically the control problem is to cause/ prevent occurrence of:

i. Particular values of outputs process variables
ii. Particular values of outputs obeying timing restrictions
iii. Given sequences of discrete outputs
iv. Given orders between various discrete outputs

Robotic controls are advancing in capabilities even as other options have been emerging for control. These include open control software that can be used with multiple vendors' robots, Petri Net, Fuzzy Net, programmable logic controllers (PLCs), and other non-robotic controllers.

Robots should be easier to use for more applications, and Southwest Research Institute, SwRI established ROS-Industrial, open source industrial robotic software and working group, to broaden the application of robotics and increase robotic interoperability. ROS stands for robotic operating system. As more manufacturing facilities and distribution centres discover the benefits of robotic material handling solutions, they must decide how best to control the robot. While robot original equipment manufacturers (OEMs) offer their own tightly integrated controller, recent developments have enabled control by a Petri Net.

Adding open-control software such as Petri Net to the convergence of well-known controls principles makes it possible to create machine designs that feature seamlessly integrated robots. This results in game-changing advantages for machine builders and manufacturers and the ability to integrate robot technology into more applications, including those that are traditionally among the most cost-sensitive.

The cost-saving benefits that make this possible include: reduced wiring, network and software platforms that are shared with the overall machine automation system, and a significantly reduced machine footprint. This has led to higher performance mechatronic and robotic solutions, including product packaging with variable product flow and complex material handling lines.

Petri Net is one of several mathematical modelling languages for the description of distributed systems Petri Nets and their concepts have been extended and developed, and applied in a variety of areas. A Petri Net is a collection of directed arcs connecting places and transitions. Places may hold tokens. A design method controller for a manufacturing assembly cell modelled by a Petri net will be presented. The control goal is to enforce a set of linear constraints on the marking behaviour or state of the Petri net. The method discussed here is a powerful means of realizing these constraints because it is simple to calculate, and the Petri net structure of the solution makes the controller easy to implement. The controller computation is derived using the concept of Petri net place invariants. [10]

Last but not least, to be precisely this project is about the KUKA robot pick and place system controlled using PN simulation. This robot is a part from eight stations which available to complete a disc casing loading and unloading assembly line. It is the 7th station of the whole system. The design of the controller using PN will compromise the effective control sequence of the robot. The properties of the design that must satisfy is safeness, boundedness, reachability, liveness and reversibility. The model of pick and place operation in Boolean function occupies design of robotic arm and position of sensors and axis.

1.2 Problem Statement

Initially, many industries use Programmable Logic Controller (PLC) as controller for their machines. This is because PLC is a microcontroller that can act as a central processor unit to control the sensor and other devices using the program which being design by the user itself. PLC system very precise to be used in the packaging workstation because it can be programmed by the operation of sequence instruction. At the same time, the design of PLC requires to maintain the quality of product and reduce the time to operate. As the complexity of the application increases, it is very crucial to ensure the safeness, liveness and reversibility properties of the system, while to maintain the performance of the system.

Currently, ladder logic diagram (LLD) which is one of the PLC elements used to resemble the operation of KUKA pick and place robot. Traditionally, LLD has lack flexibility, extremely complex to understand the diagram and less adaptability to the current changes. Even for the real time purpose, if there is any problem occurs in the system, it is hard to detect the cause of the problem via LLD. So Petri Net (PN) introduced as an alternative controller for the KUKA pick and place robot. PN act as the controller for the robot and the whole operation can be controlled with the simulation. The new controller design for pick and place robot will reduce the design complexity and able to increase the adaptability of the system to the current changes. Last but not least, PN offers a faster response time in real time response compared to LLD.

1.3 Overview of the project

The project is about designing a controller for the pick and place robotic system using PN. Petri Net still a new software which also able to function as logic controller for the robots. It is more convenient and easy to handle compare to other types of controller which are already in use. In this project, the pick and place operation also modelled in Boolean function. Thus it gives a better view of the process, inputs and outputs of the KUKA robotic arm. In the next chapter, the use advantages and techniques used in Petri Net will be discussed in further.

1.4 Objectives

Objectives of this project are:

1. To analyse the Petri Net simulation for logic control system.
2. To model pick and place robotic system using Boolean function
3. To design a controller for pick and place robotic system by using Petri Net

1.5 Scope

The KUKA pick and place robot is a part of the disc casing loading assembly line. The experiment is carried in Faculty of Electrical Engineering (FKE), Universiti Teknikal Malaysia Melaka (UTeM). The model of the robot is RB-CIM-UL-03 which used to pick and place the souvenir at the pallet area position. The robot is manufactured in the year 2011 and replaced the robot which used previously. PLC and KUKA programming language used as the controller for the RB-CIM-UL-03 robot. The scope includes generate a KUKA robot program and PLC using LLD for pick a souvenir from base position and place it at pallet area position. The Petri net is a simulation software which is used to design a controller for the robot. The function and the operation of the Petri Net identified clearly and controller for pick and place system design using Petri Net. Model for the KUKA pick and place operation also generated using Boolean function. The modelling for the pick and place robot includes the inputs and outputs of the system.

CHAPTER 2

LITERATURE REVIEW

2.0 Introduction

This chapter present about some basic principle and theories in the project and review of previous journals about the Petri Net (PN), Boolean function for pick and place robot system, KUKA pick and place robot, KUKA programming language and the PN controller design for KUKA pick and place robot system. In addition, the history of PN and KUKA pick and place robot also provided in this chapter. Besides that, this chapter also explained the implementation of PN in discrete control system. Last but not least, the comparison between PN and other types of controller are also included in this chapter.

2.1 Theory of basic principle of Petri Net

In this section basic principle such as basic operation and usage of PN are discussed. The theory involved in analysis of the PN as a controller for pick and place robot.

2.1.1 Study of PN

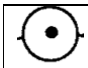
There are many techniques used for mathematical modelling of system. Which are namely basic queuing networks, queuing system with MM1 queuing, Fuzzy Net and close doing networks. Petri Net is another form of mathematical modelling to formally specify a particular system. Petri Net (PN) was invented by Carl Adam Petri in his PhD thesis in 1962 and initially had the form of Condition/Event Systems with simple arcs and binary marking. After that, there are a few modification of the basic system model done, which includes the integer markings and weighted arcs. [10]

PN used to model the functionality and behaviour of a system. It also can derive some performance result. In term of application, it can use in computer systems, computer networks and protocols, manufacturing system, production system, scheduling systems and controllers. PN also used to combine the computer system and manufacturing system by testing or checking the behaviour of certain components and parts. [1]

PN is a formal language to represent system and also a graphical language for modelling systems with concurrency. PN have an exact mathematical definition of their execution semantics, with proper mathematical theory for process analysis. Indirectly, a Place/Transition (P/T) Petri net is known as a bipartite graph. It consists of two types of nodes places and transition. Places typically drawn as circles and transition represented by bars or rectangles. Place represent condition where the condition needs to be full filled and the transition is described as an event that occur or processing activity. When condition is satisfied transition will trigger. Then, there is a changing in system from one state to another by transition. Symbols used in PN are given in Table 2.1.

Table 2.1: List of Input/output and symbol

symbol	Name	Function
□	Transition	Event/Activity
○	Place	Condition
↗	Arc	Connector

	Token	Number of resourses

Places and transitions are interconnected by directed arcs. Arcs only exist in between places and transition or vice versa. Arc always connects two nodes of different types. Arc can be weighted, which is representation of set of parallel arcs. There are tokens exist between the places and transitions. It circulates in this system via the transitions.

2.1.2 Concepts and examples of Petri Net

A marked Petri net (PN) Z = (P, T, I, O, m) is a five tuple where

1. P is a finite set of places
2. T is a finite set of transitions with $P \cup T \neq 0$ and $P \cap T=0$
3. $I: P \times T \rightarrow N$, is an input function that defines the set of directed arcs from P to T where N = (0, l, 2,. . .}
4. O: $P \times T \rightarrow N$, is an output function that defines the set of directed arcs from T to P
5. m: $P \rightarrow N$, is a marking whose i^{th} component represents the number of tokens in the i^{th} place. An initial marking is denoted by mo.

Example 1: A marked PN is shown in figure 2.1 and its formal description is given as follows:

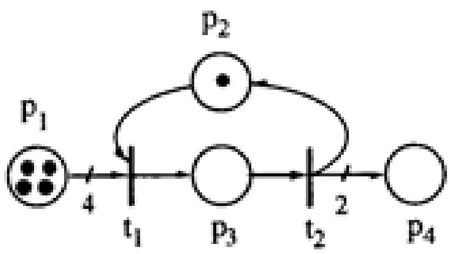


Figure 2.1: A Petri net modelling one operation stage requiring a single resource.

$P = \{p_1, p_1, p_1, p_4\}, \quad T = \{t_1, t_2\}$

$I(p_1, t_1) = 4, \quad I(p_1, t_1) = 0;$
$O(p_1, t_1) = 4, \quad O(p_1, t_1) = 0;$
$I(p_1, t_1) = 4, \quad I(p_1, t_1) = 0;$
$O(p_1, t_1) = 4, \quad O(p_1, t_1)$
$I(p_1, t_1) = 4, \quad I(p_1, t_1) = 1;$
$= 1;$

$O(p_1, t_1) = 4, \quad O(p_1, t_1) = 0;$
$I(p_1, t_1) = 4, \quad I(p_1, t_1) = 0;$
$O(p_1, t_1) = 4, \quad O(p_1, t_1) = 2;$
$m = (4 \quad 1 \quad 0 \quad 0)^T.$

Input and output functions can be represented as matrices,

$$I = \begin{pmatrix} 4 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix} \quad \text{and} \quad O = \begin{pmatrix} 4 & 0 \\ 1 & 1 \\ 0 & 0 \\ 0 & 2 \end{pmatrix}.$$

The incidence matrix:

$$C = O - I = \begin{pmatrix} -4 & 0 \\ -1 & 1 \\ 1 & -1 \\ 0 & 2 \end{pmatrix}.$$

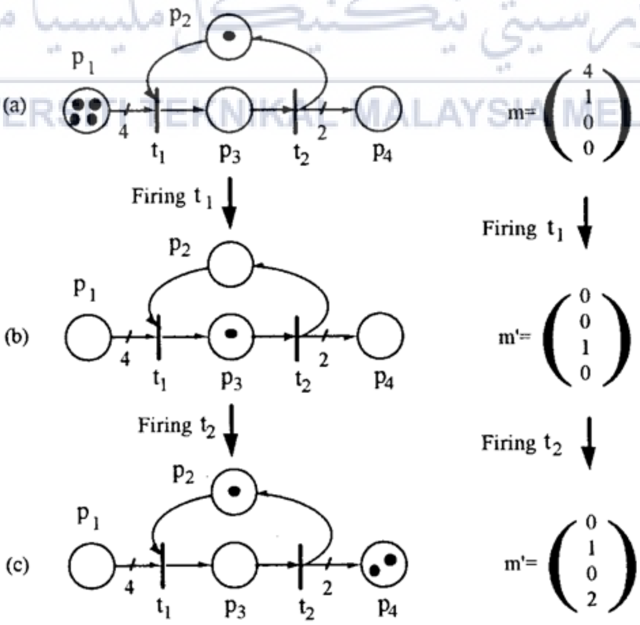


Figure 2.2: Evolution of markings of PN in example 1: (a) m= (4 1 0 0) T; (b) m0= (0 0 1 0) T; and (c) m00= (0 1 0 2) T

2.1.3 Petri Net simulation analysis

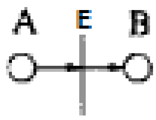


Figure 2.3: PN input and output

The example of PN in system is shown figure 2.3. A is the input in circular form for the particular transition E. It is a place where certain condition occupied. E is the transition and can be said a processing part which also can be draw in rectangular form. Transition is like gate which goes from place to place through gates. The gates is only will open when certain conditions are fulfilled. In this example transitions correspond to some activity which is A. B is the output where which is also in circular form. Place B is the new input to the subsequent transitions which was changed state from A when pass through E. Tokens get created in places like A and move around the system. The simplest rule for a transition take place is minimum one token must be present in all the input places that connected to the transition and the transition will fire. The result of the fire is one token will be removed from the input places and one token added to the output places. The number of input places is not dependent. There can be more number of input places then the output places and vice versa. Whenever the transition occurs some values deposited in the output places and tokens get created.

2.1.4 Important properties of Petri Net

2.1.4.1 Liveness

Liveness is considered as the absence of deadlocks. This property guarantee that all places can be act and it make the operation or the condition which is modeled can happen in evolution of process. The specification can call ass the properties that being produce base on condition of the model. Thus, this property is related to the safety that always in good condition and verification process will check the properties being specify during the executions. Hence, this liveness properties can be check as safety checking by doubling the number of state elements that only can be divided by finite length counter examples.

2.1.4.2 Safeness

The safeness property will guarantee the stable behavior of the system without any overflow. Furthermore, as for the safeness property of the places which reflects that operation, that there is no attempt in request more execution of ongoing operation. The important effect of safeness is it Boolean representation of plate and can make a direct conversion to Petri net.

2.1.4.3 Reversibility

Reversibility indicates the cyclic behavior of a system that performs at its function repeatedly and the recoverability of the initial state from any states of the system.

2.1.4.4 Boundedness

Given a PN Z, Z is bounded if each place in P is bounded. Safeness is 1-boundedness. These two properties are behavioural. The structural one is defined as follows: Z is structurally bounded if Z is B- bounded for some B given any finite initial marking *mo*. Places are frequently used to represent storage areas for parts, tools, pallets, and automated guided vehicles in manufacturing systems. Boundedness is used to identify the existence of overflows in the modelled system. The concept of boundedness is often interpreted as stability of a discrete manufacturing system when it is modelled as a queuing system.

2.1.4.5 Reachability

Reachability checks whether the system can reach a specific state, exhibiting particular functional behaviour. It is behavioural property.

2.1.5 Application of Petri Net in discrete control system

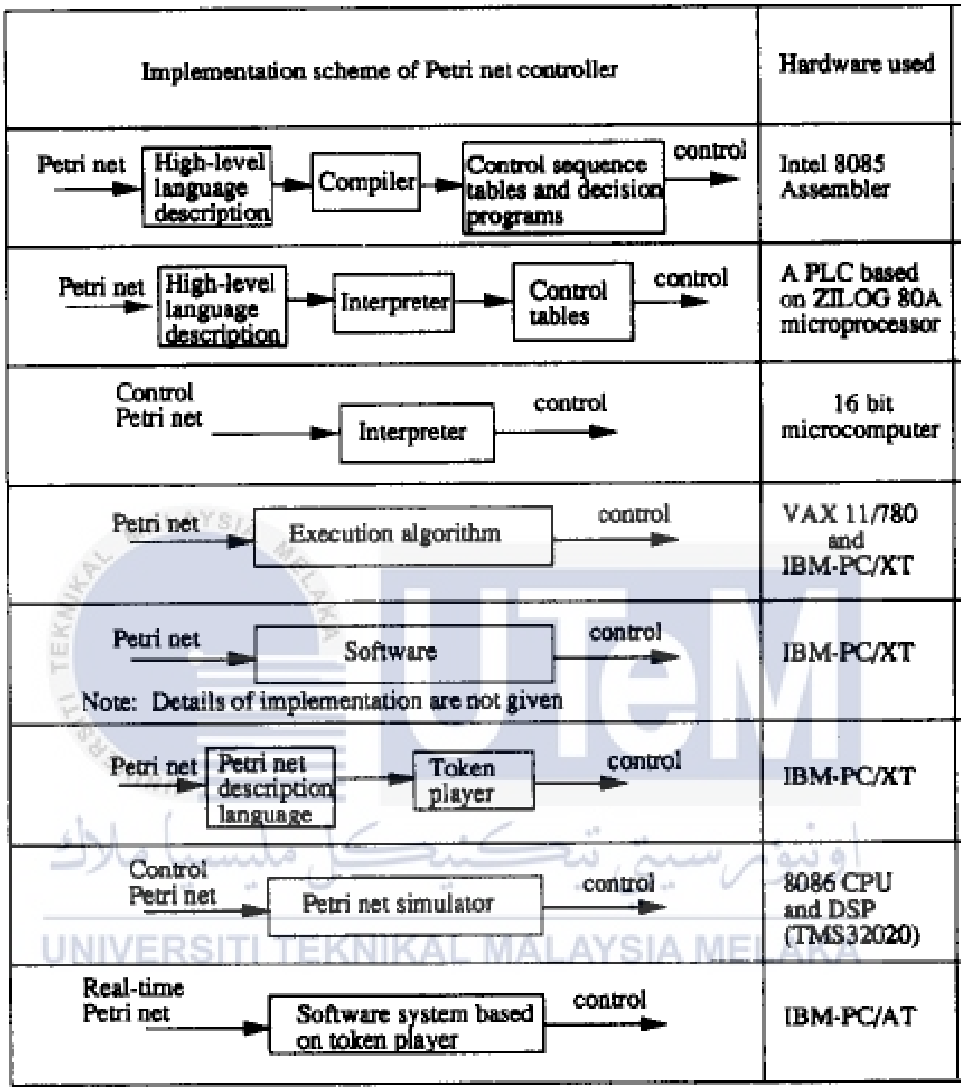


Figure 2.4: Various methods of PN based sequence control

There exist several classes of PN's with many applications for discrete control. Figure 2.4 shows the methods which are based on the PN sequence control. This proves the efficiency of PN as controller in various fields. A new design for pick and place robot will be created using PN and it will act a flexible and effective controlling tool for any kind of robots.

2.1.6 Petri Net working principle

In term of programming terminology, pre-conditions and post conditions are satisfied. The system is able to function as prescribed. The simulation which is created can used to control the pick and place robot. First, the simulation needs to be converted into programming tool. Then it will be saved in the new folder as new programming instruction. Once the system is ready the simulation can be executed. The simulation designed now act as the new controller for the KUKA pick and place robot. The design must occupy the closed loop control system as it response from the input from the sensors.

2.2 Theory of basic principle of KUKA pick and place robot

In this section basic principle such as types controllers available for pick and place robot and its usage are discussed. The theory involved in analysis of the PN as a controller for pick and place robot.

2.2.1 Study of KUKA pick and place robot

The term robot comes from the Slavic word robota, meaning hard work. According to the official definition of an industrial robot, a robot is a freely programmable, program controlled handling device. The robot thus includes the controller and the operator control device, together with the connecting cables and software. This robot used the mechanical arm to pick and place the objects. The programmable mechanical arm was first designed by George Devol in 1954. [6] He called the term Universal Automation. The robot not only performs function to pick and place item, but also widely used in industries to hazardous duty service, maintenance jobs, and medical applications.

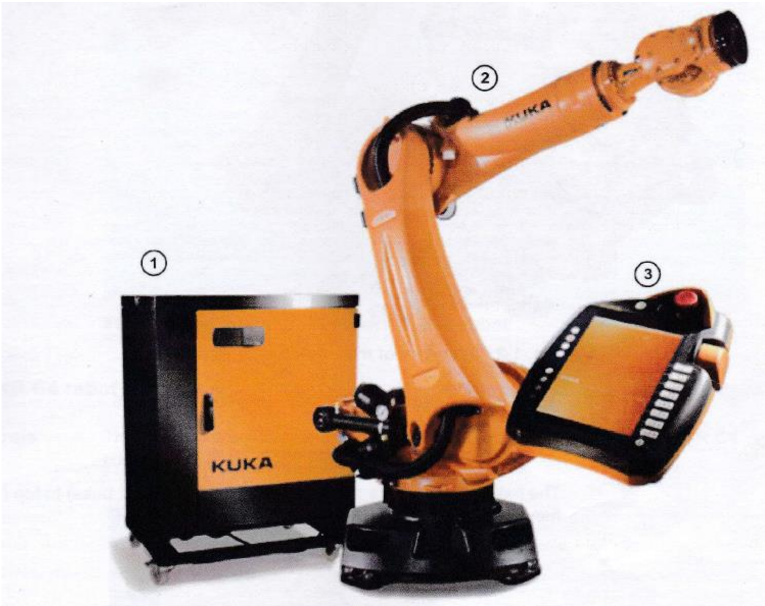


Figure 2.5: KUKA industrial robot

2.2.2 Robot specification

The model of the pick and place robot is RB-CIM-UL-03 made by KUKA robotics corporation. KUKA is a German manufacturer of industrial robots and solutions for factory automation. It has a Cartesian mechanical arm for pick and place work. This means arm has three prismatic joints, whose axes are coincident with a Cartesian coordinator. [6]This robot is placed as the 7th station of the disc casing loader assembly. The pick and place robot function to unload the disc which fixed with a casing. The other stations which are included in this assembly cell are stated in table 2.3.

Table 2.2: Stations in assembly line

Number of station	Function	Remarks
1	Souvenir Base Loader	Siemen robots
2	Engraving	Siemen robots
3	Disc casing Loader	Siemen robots

4	Press Station	Siemen robots
5	Casing cover opener	Siemen robots
6	Disc loader	Siemen robots
7	Un-loader	KUKA pick and place robot
8	Dummy	-

The sequence of the stations and its function is stated in table 2.2. The whole system is controlled by main control system unit where the power supply and other sources integrated.

2.2.3 Components of pick and place robot

2.2.3.1 Structure

The structure of a pick and place arm is normally generally mechanical and can be known as a kinematic chain. The chain is shaped of connections, actuators, and joints which can permit one or more degrees of flexibility. Pick and place robot arm use open serial chains in which every connection join the one preceding to the one after it. These pick and place robot arm are regularly take after the human arm. [6]

2.2.3.2 KUKA robot programming language

Table 2.3: The movement of KUKA robot from one point to another in three main ways[6]

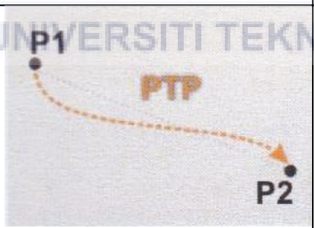
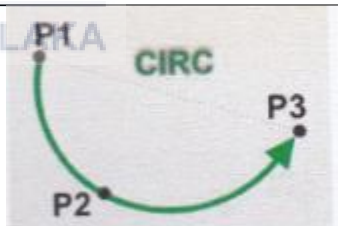
Type	PTP – Point-to-Point	LIN – Linear	CIRC – Circular
Explanation	Motion along the quickest path to an end point. This motion requires the programmer to “teach” one point.	Motion at a defined velocity and acceleration along a straight line. This motion requires the programmer to “teach” one point. The robot uses the point defined in the previous move as the start point and the point defined in the current command as the end point and interpolates a straight line in between the two points.	Motion at a defined velocity and acceleration along a circular path or a portion of a circular path. This motion requires the programmer to “teach” two points, the mid-point and the end point. Using the start point of the robot (defined as the end point in the previous motion command) the robot interpolates a circular path through the mid-point and to the end point.
Images			

Table 2.4: Inputs and Outputs

Type	Inputs	Outputs
Explanation	An input is something (digital or analogue) coming from another system and is read in and used to make decisions. Inputs cannot be changed by the robot is represent the state of something external to the robot such as whether a sensor is on or off. In the robots our inputs are defined from 33 through 40. These inputs can be set to any number but the external inputs that are numbered 0 through 7 are reflected in the programming language as 33 through 40. Therefore to refer to physical input 0 in the program the syntax is $IN [33]. Physical input 4 would be $IN [37].	Outputs can be changed by the robot but can also be monitored. The numbering is the same as the inputs. Physical output 0 is output 33 in the program. The syntax is $OUT [33] for output 33. The syntax to change the state of the output is $OUT [33] =TRUE to cause physical output 0 to turn on and $OUT [33] =FALSE to cause physical output 0 to turn off.

2.2.3.3 Execution Control

The following are different ways to control the execution of program.

1. If statements –if statements checks a condition and executes code if the condition is true and may execute code (if written) if the condition is false. The syntax is as follows.

IF *conditional* == TRUE THEN

Whatever code want to execute when the conditional is TRUE

ELSE

Whatever code you want to execute when the conditional is FALSE

ENDIF

For example, if a switch connected to physical input 0 the following code might be used. IF $IN [33] ==TRUE THEN

The code written here would execute when the switch was on.

ELSE

The code written here would execute when the switch was off.

ENDIF

The ELSE statement is optional and if not used should not be entered in. In other words if in the example above nothing should happen if the switch was off the following code could be used.

IF $IN [33] ==TRUE THEN

The code written here would execute when the switch was on.

ENDIF

For those of you who are familiar with programming, there is no “ELSEIF” option. However, you can nest the IF statements within each other.

2.2.3.4 Actuation

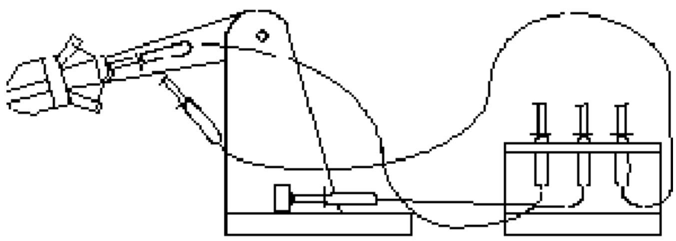
Actuators are similar to the "muscles" of pick and place robot arm, the parts which convert stored energy into movement. By a wide margin the most prevalent actuators are servo motor that turns a wheel or gear, and direct actuators that control modern pick and place robot arm in variables. It also rotate to a specific angular position, like steering. Yet, there are some late advances in option sorts of actuators, controlled by power, chemicals, or packed air. [6]

Figure 2.6: Servo motor

2.2.3.5 Vision

Computer vision is the science and technology of machines that see. The computer visions depend on image sensors which detect electromagnetic radiation in the form of either visible light or infra-red light. The sensors are designed using solid-state physics where light propagates and reflects off surfaces.

2.2.3.6 Mechanical gripper of pick and place robot

Pneumatic(compressed gases)	

Gripper is a very important in order to pick an object. It is just two fingers which can open and close to pick up and release a range of small objects. Typically, it has a chain with a metal wire run through.

2.2.3.7 Working principle of robot

The KUKA pick and place robot controlled by the main KR C4 control cabinet where the teach pendant used to control the robot manually. The KUKA pick and place robot use mechanical arm for pick and place the products in cartons. The pick and place mechanical arm combined with a visual observation system for identifying products. When the disc arrived after packed with a disc cover, pick and place robot from station seven pick the disc, then place it in the cartons provided. This is the function pick and place robot in this assembly line. Generally, this is a human controlled based system that detects the object, picks that object from source location and places at desired location.

2.3 Programmable Logic Controller (PLC)

PLC is an example of microprocessor based controller which implements programmable memory in order to do and store instructions to control the process of machine. Many industry platforms were being used such as in petro-chemical plants, smelting furnace, automobile production line and much more.

The PLC is used to apply in entirely in real-time systems. The PLC are function to control the systems by using actor and sensor. For example, a sensor will act whether the tank is full or not. If the tank is full occupied, the signal from sensor will on, otherwise it will turn off. For actor the example is used to control opening and closing a valve. This condition happens when the valve is open, the tank will be filled. Therefore, the PLC will control the input signal from sensor and control the actor of the system.

The advantage of the PLC is on the flexibility on their working. Thus, the program can be modified easily and quickly according to situation involved. As for the disadvantages of PLC, there is certain delay between the change in input and output signals.

The pick and place robot in this project will use the PLC of OMRON SYSMAC CJIM programmable controller like in the Figure 2.7 below. The PLC can be divided into two which are module type and fix type. As for this CJIM is a module type of PLC which can be connect into rack and the electrical connection can be done through input and output port. This type of PLC it is being implemented in many control system applications in the production assembly line. This is due to the advantage that can overcome and withstand the humidity, temperature, vibration and noise. Moreover, the program language is easier to understand by non-programmer to modify the changes in the program without having no problem and smoothly.

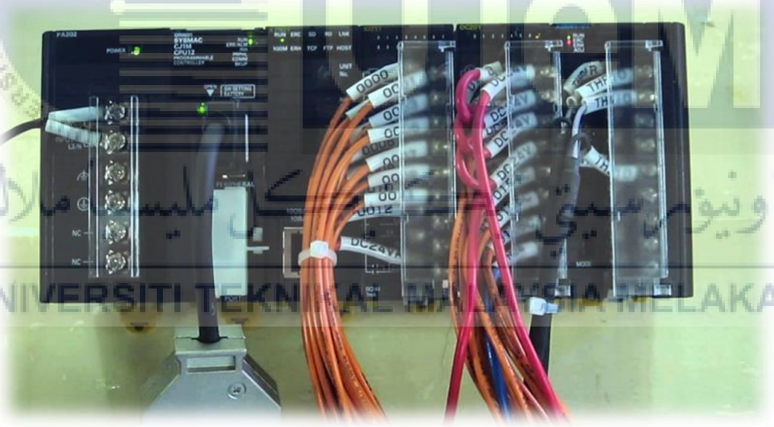


Figure 2.7: OMRON SYSMAC CJIM

2.3.1 PLC Programming Language

The program language is a list of instruction set which been program or written in a form of language that can be synthesis and read by the processor computer such as the PLC. This is done by storing the input language data in the memory and then it can be execute and run by the sequence of the instruction set that being program by the user. Nowadays, each manufacturer of the PLC has its own specification of different program language that will be used to program the PLC. For example, there many types of program language that which can use to program according the type of the PLC itself. Hence, for the OMRON type PLC, the software called CX Programmer being used to program and write the design of system.

The program language that will be used in the PLC usually to be in simple design language that can be understand by the user that average computing skill in order to manage the changes and modify the program which to control the system. To improve the software program language in PLC, the International Electro-Technical Commission (IEC) has introduced a standard programming language which is called IEC 1131-3. Base on the standard program language IEC. 1131-3, there is five type of language program which it is then classify into two category of method which in textual and graphical. Examples of the graphical language program for the PLC are Ladder Diagram (LD), Function Block Diagram (FBD), and Sequential Functional Chart (SFC). Whereas as for example in the textual language program in PLC are includes structure text (ST) and instruction list (IL).

IEC standard included and covered all the aspect about the technology which is been implemented nowadays and the electrical is an example in their field that sustain the requirement of the standard in term of safety, performance, and etc.

2.3.2 Ladder Diagram (LD)

Ladder diagram program it was a very popular method for programming. This is because, ladder programming based on electrical wiring diagrams. Thus, in this diagram, it can see how the electricity runs through the control scheme. This require only basic programming skill needed to design ladder diagram and can be learnt in a short time because this graphical presentation can be understood almost intuitively. However, when the troubleshoot situation occurred, it is very difficult to analyze the problem of program. This is because, only the programmer that design the program will know how exactly to analyze, interpret, troubleshoot and make modification for the program.

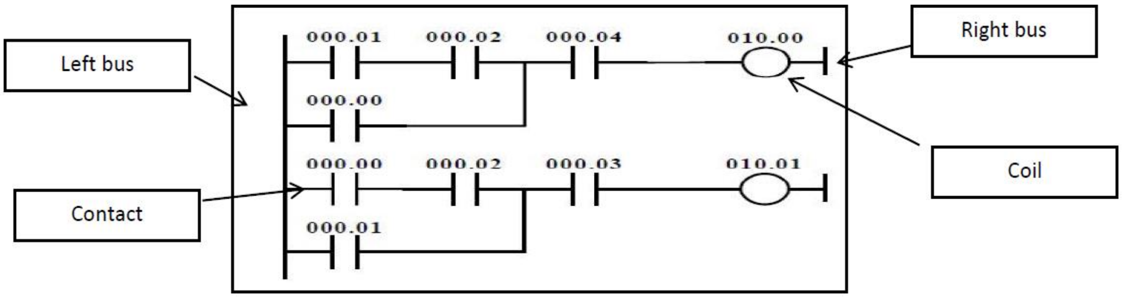


Figure 2.8: Ladder diagram program

The Figure 2.8 above shows the basic example of program using ladder diagram. The contact can be in normally open (NO) or normally close (NC) condition. This contact is to give the input for program to energize the output of the system. The input that are used in the system can be a device such as switch or send which have function to send the input signal to the PLC. As the coil being used in the program determine the output of the system. The output can be motor or solenoid where it is represent the actuator of the system. In this ladder program, there is also the other instruction used to run the program become more reliable when it is apply.

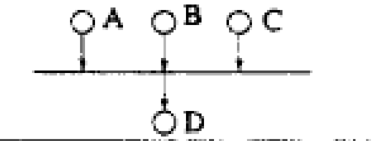
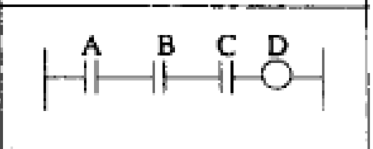
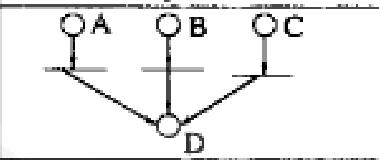
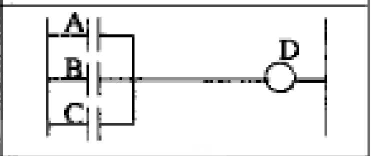
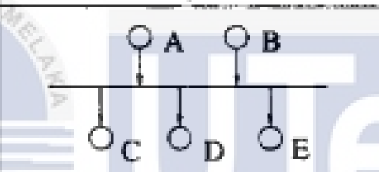
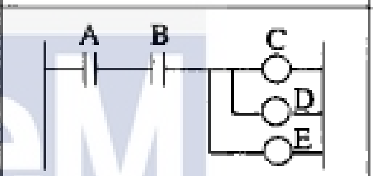
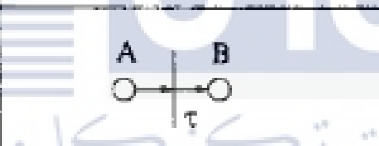
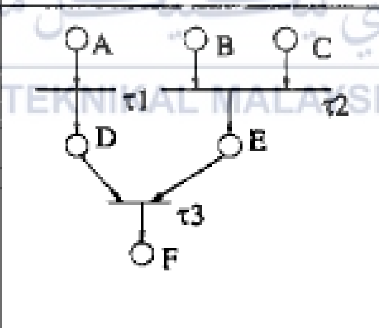
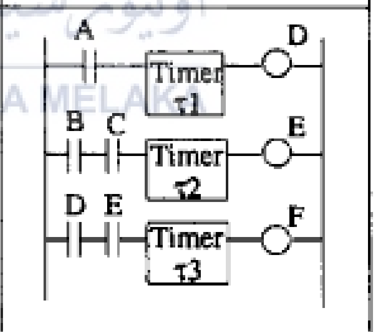
2.4 Summary

There are many types of logic controller practised recently. Namely, Petri Net, ladder logic diagram (LLD), and Fuzzy Net. These controllers widely used in automotive industry to control the robot operation. Now, there is comparison to be made between these controllers. The comparison made based on the type, flexibility, function, and complexity.

2.4.1 Comparing Petri Net and Ladder logic diagram for sequence controller design through a discrete manufacturing system.

Ladder logic diagrams (LLD) are used show the abbreviation of operations executed by the systems control software. The diagrams used in the LLD grow so complex and locating the target when a problem is arising becomes very difficult. Their usage is limited only to control a system but not to analyse and qualitative the performance characteristics. The changes need to be fulfilled with new updates. The limitations in LLD as integrated tool eliminated by contribution of PN which has numerous application in manufacturing system. The comparison is made between Petri Net and Ladder logic diagram for sequence controller design through a discrete manufacturing system. [3] The logic and other basic building blocks used in sequence control are modelled by PN and LLD is shown in table 2.5.

Table 2.5: Controller logic representation of Petri Nets and Ladder Logic Diagrams [3]

Logic constructs	Petri nets	Ladder logic diagrams
Condition or the status of a system element	Place	No explicit representation
An activity	Transition	No explicit representation
Flow of information or material.	Directed arc	No explicit representation
Objects such as machines, robots, pallets, etc.	Token(s) in place(s)	No explicit representation
Logical AND IF A = 1 and B = 1 and C = 1 THEN D = 1		
Logical OR IF A = 1 or B = 1 or C = 1 THEN D = 1		
Concurrency IF A = 1 and B = 1 THEN C = 1 and D = 1 and E = 1		
Time delay IF A = 1 THEN delay "τ time units" B = 1		
Synchronization IF A = 1 THEN delay "τ1 time units"; D = 1 IF B = 1 and C = 1 THEN delay "τ2 time units"; E = 1 IF D = 1 and E = 1 THEN delay "τ3 time units"; F = 1		

There are four rows shows the basic PN element to model the conditions, status, activity, information and material flow and resources. Logical gates which are AND and OR can be modelled easily in LLD and PN. Other aspects which are time delay, synchronization, and concurrency also illustrated in Table 2.5. The two main factors that PN and LLD compared for are complexity and response time that will be discussed further.

2.4.1.1 Design complexity

Design complexity means the complexity in the design for control logic for a given specification. Basically it is influenced by many factors such as experience of designers, and control program weight and size, and yet it is difficult to quantify. In any case, it can be portrayed by two elements, to be specific graphical complexity and adaptability for change in specification which shown in table 2.6.

Table 2.6: Comparison between LLD and PN

Factors	Definition	Petri Net	LLD
Graphical complexity	It is primarily controlled by the quantity of nodes and connections for a given graphical control logic design. Graphical complexity impacts the understanding of control logic by individuals who don't have learning of either PN's or LLD's. Henceforth, it is a critical component in designing the logic during implementation.	Complex	Very complex
Adaptability	Control sequence should be changed frequently to meet the powerfully changing necessities of the market. The control programming ought to be effortlessly versatile to changes in determinations with a specific end goal to enhance the product profitability and subsequently keep insignificant advancement time. One of two outlines is said to be more versatile on the off chance that it needs less changes contrasted with another keeping in mind the end goal to satisfy a determination change. [3]	High	Low

2.4.1.2 Response Time

Response time is termed as scan time in LLD and execution time in PN. The system decides how fast the control system responds to an event. In the case of response time, it is not only affected by physical appearance which is size of the model but also the method of implementation. The response time can be measured accurately for a given logic design. PN have nodes which are places and transition which linked by arcs, while LLD nodes opened/ closed switches, timers, counters, relays, and push buttons, and linked which is complex material and the response time is slower than PN. The nodes and links in real time Petri net are less than LLD. Other than that, PN model the framework all the more reasonably by common mapping of the farthest point switches, start, and stop buttons as places and can be effortlessly reached out to model breakdown handling procedures using methods of Augmented Timed Petri nets. Lastly, using RTPN timers, counters, and an emergency stop are can be modelled easily as automatic resetting of timers and counters is also embedded in PN.[3]

2.5 Conclusion

In summary the basic principles such as the background study of the PN and KUKA pick and place robot with some theory involve in analysis in this chapter. The simulation techniques and various method of PN also have been discussed in this chapter. At the same time, the pick and place robot specification and components of the robot also have been discussed in this chapter. Other than that, the working principle of PN simulator and pick and place KUKA robot have been analysed in this chapter. Lastly, the comparison between the PN and LLD also discussed and evaluated.

CHAPTER 3

METHODOLOGY

3.0 Introduction

This chapter discuss in more detail about the method and the process that is involved to complete this project. This chapter will also discuss more briefly about the flow chart, block diagram, simulation for both PN and PLC and also the data analysis. This study will employ the descriptive method using certain materials and instruments to complete the project research in order to create a project that can be achieve the outcomes which have been expected. This project will show how the Petri Net simulation is applied in automation system to control the operation system. The souvenir assembly line (RB-CIM-UL-03) robot that is located in CERIA laboratory at University Teknikal Malaysia Melaka (UTeM) Faculty of Electrical Engineering will be used as the model for this final year project, it consist of eight work station that is souvenir base loader, engraving, disc casing loader, press, disc casing opener, disc loader, un-loader and spare. There will be only one station that being used for this project which is a souvenir base unloader station where it is the last station for this system. This system is being monitored using the supervisory control and data acquisition (SCADA) system that have being integrated and operated in CIM environment where the process is make consequently. Petri net is very unique controller that can ensure the reliability and efficiency of the system. The topics that will cover in this chapter is:

I. Model for pick and place robot using Boolean function
II. KUKA programming language

III. Ladder diagram for pick and place operation

IV. Petri Net simulation for pick and place operation

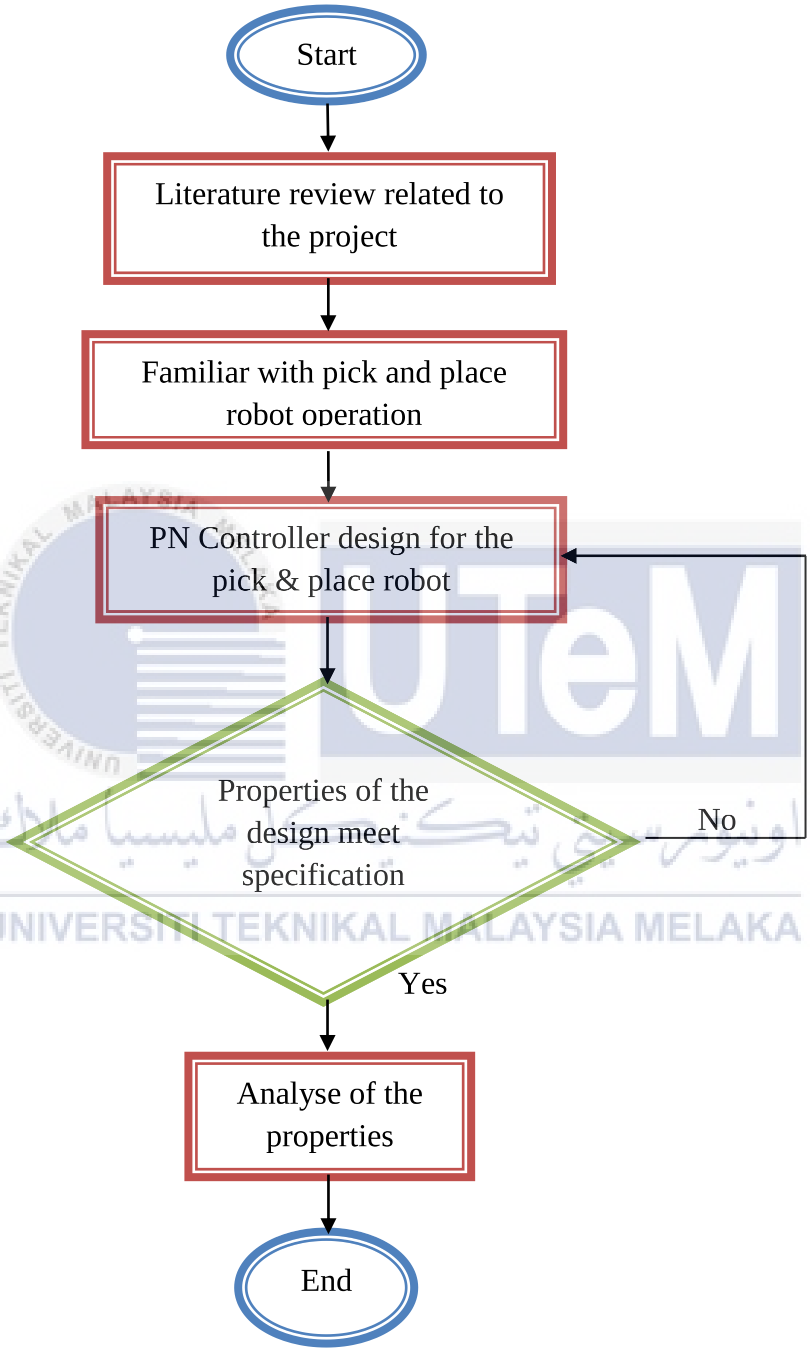


Figure 3.1: Flowchart methodology for overall process

3.1 Project methodology to achieve first objective.

This section discuss about the project methodology to achieve the first objective which is to analyse the Petri Net simulation for logic control system.

3.1.1 First objective

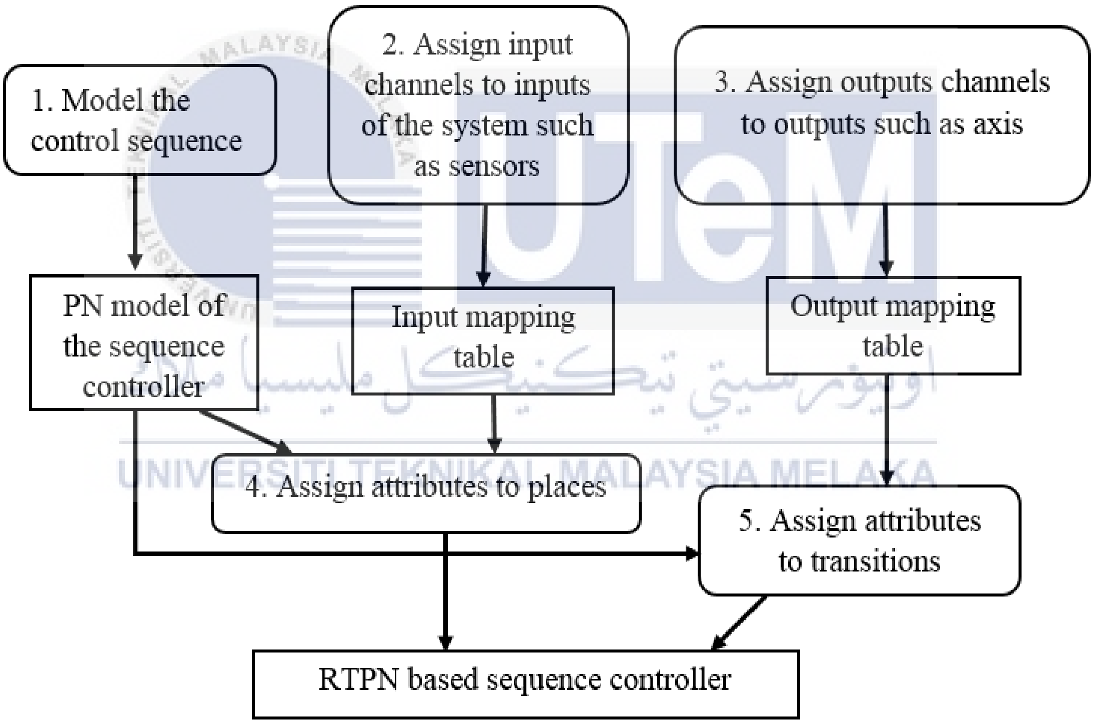


Figure 3.2: The procedure to design a real-time Petri net based discrete event controller [3]

3.1.1.1 Process

Based on Figure 3.2, there are five steps must be followed in order to design PN based controller. These procedures aid to formulate RPTN based controller for a specific sequence. The process is as follows:

1. Model the control sequence with PN to get the PN model of the sequence controller. This is similar to the idea of identifying the input which is the places in PN.
2. The input channel assigned to input of the system. The inputs of the system are limit switches, and sensors. This will help to formulate an input mapping table. If relate this process to the simulation, the condition from sensor must be satisfied in order further process to continue.
3. Assign the output channels to outputs of the system. The output to the system is solenoids and switches. The timing information also must be acquired for activities to get an output mapping table.
4. The process using the input mapping table to assign an input channel to every place in the PN based controller. The starting state of the system controls the initial marking of RTPN. In the PN model a few places do not represent to the inputs of the system as they represent to the middle conditions of system or logical places to model counters in the grouping. Consequently, no channel must be assigned to these spots, but spoken to by "- ".
5. Utilizing the output mapping table and the action(s) that are displayed by a move, allocate a number to every transition in a PN based controller. The operations and the time a delay given in the grouping to be controlled chooses firing time function of RTPN. In the PN model a transitions do represent concurrent actions. Thus, care ought to be taken to allocate the numbers for such transitions.

3.2 Project methodology to achieve second methodology

This section discuss about the project methodology to achieve the second objective which is model pick and place robot system using Boolean function.

3.2.1 Model for pick and place robot movement using Boolean function

The model that used in the PLC is using logical sequence to control the process, the behaviour of the system must be determined primarily to construct the model the system, below on Figure 3.3 shown the sequence of the pick and place robot.

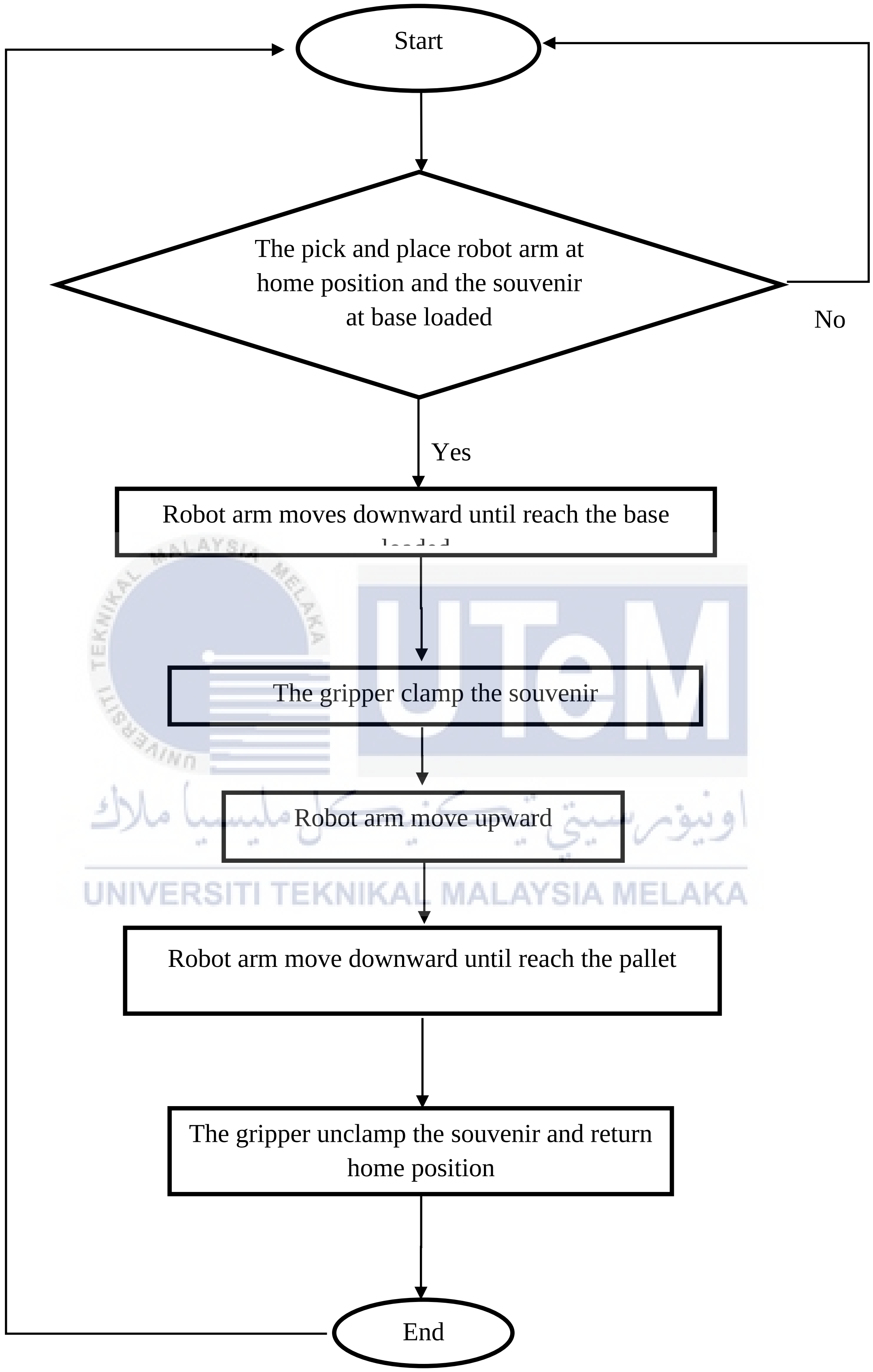


Figure 3.3: Flow chart for pick and place robot process

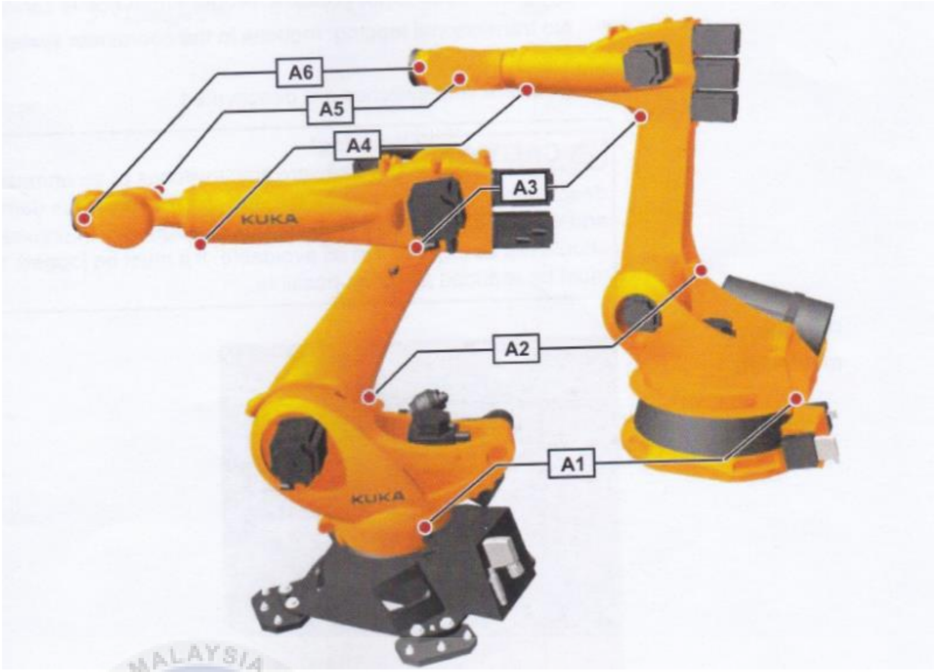


Figure 3.4: Pick and place robot model

Figure 3.5: Robot axis movement

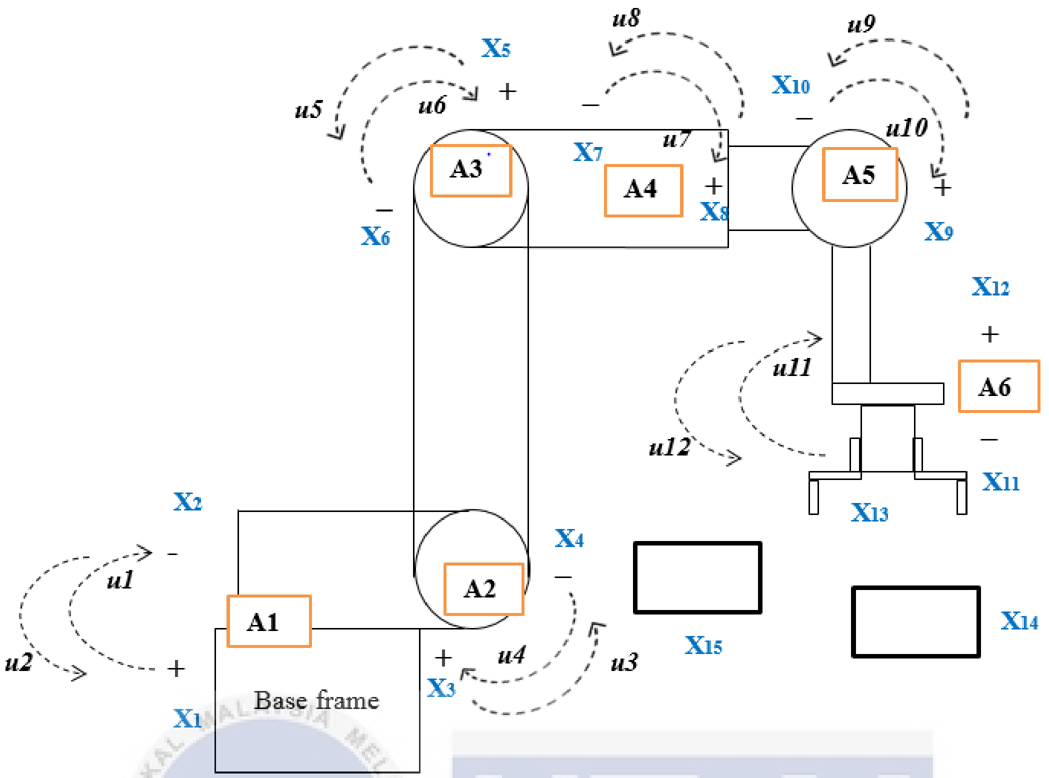


Figure 3.6: Initial position of robot arm

Table 3.1: Operation model of axis A1

Precondition		Input		Post condition	
X1	X2	U1	U2	dX1	dX2
1	0	1	0	0	0
0	0	1	0	0	1
0	1	0	1	0	0
0	0	0	1	1	0

$$dX_1 = \overline{x_1}.\overline{x_2}.\overline{u_1}.u_2 \vee x_1.\overline{\overline{x_2}.u_1.\overline{u_2}}$$

$$dX_2 = \overline{x_1}.\overline{x_2}.u_1.\overline{u_2} \vee x_2.\overline{\overline{x_1}.\overline{u_1}.u_2}$$

Table 3.2: Operation model of axis A2

Precondition		Input		Post condition	
X3	X4	U3	U4	dX3	dX4
0	1	0	1	0	0
0	0	0	1	1	0
1	0	1	0	0	0
0	0	1	0	0	1

$$dX_3 = \overline{x_3}.\overline{x_4}.\overline{u_3}.u_4 \ V \ x_3.\overline{\overline{x_4}.u_3.\overline{u_4}}$$

$$dX_4 = \overline{x_3}.\overline{x_4}.u_3.\overline{u_4} \ V \ x_4.\overline{\overline{x_3}.\overline{u_3}.u_4}$$

Table 3.3: Operation model of axis A3

Precondition		Input		Post condition	
X5	X6	U5	U6	dX5	dX6
0	1	0	1	0	0
0	0	0	1	1	0
1	0	1	0	0	0
0	0	1	0	0	1

$$dX_5 = \overline{x_5}.\overline{x_6}.\overline{u_5}.u_6 \ V \ x_5.\overline{\overline{x_6}.u_5.\overline{u_6}}$$

$$dX_6 = \overline{x_5}.\overline{x_6}.u_5.\overline{u_6} \ V \ x_6.\overline{\overline{x_5}.\overline{u_5}.u_6}$$

Table 3.4: Operation model of axis A4

Precondition		Input		Post condition	
X7	X8	U7	U8	dX7	dX8
1	0	1	0	0	0
0	0	1	0	0	0
0	1	0	1	0	0
0	0	0	1	1	0

$$dX_7 = \overline{x_7}.\overline{x_8}.\overline{u_7}.u_8 \text{ V } x_7.\overline{\overline{x_8}.u_7.\overline{u_8}}$$

$$dX_8 = \overline{x_7}.\overline{x_8}.u_7.\overline{u_8} \text{ V } x_8.\overline{\overline{x_7}.\overline{u_7}.u_8}$$

Table 3.5: Operation model of axis A5

Precondition		Input		Post condition	
X9	X10	U9	U10	dX9	dX10
0	1	0	1	0	0
0	0	0	1	1	0
1	0	1	0	0	0
0	0	1	0	0	1

$$dX_9 = \overline{x_9}.\overline{x_{10}}.\overline{u_9}.u_{10} \text{ V } x_9.\overline{\overline{x_{10}}.u_9.\overline{u_{10}}}$$

$$dX_{10} = \overline{x_9}.\overline{x_{10}}.u_9.\overline{u_{10}} \text{ V } x_{10}.\overline{\overline{x_9}.\overline{u_9}.u_{10}}$$

Table 3.6: Operation model of axis A6

Precondition		Input		Post condition	
X11	X12	U11	U12	dX11	dX12
1	0	1	0	0	0
0	0	1	0	0	0
0	1	0	1	0	0
0	0	0	1	1	0

$$dX_{11} = \overline{x_{11}}.\overline{x_{12}}.\overline{u_{11}}.u_{12} \text{ V } x_{11}.\overline{\overline{x_{12}}.u_{11}.\overline{u_{12}}}$$

$$dX_{12} = \overline{x_{11}}.\overline{x_{12}}.u_{11}.\overline{u_{12}} \text{ V } x_{12}.\overline{\overline{x_{11}}.\overline{u_{11}}.u_{12}}$$

Table 3.7: Operation model of gripper

Precondition															Input	Post condition
X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	X13	X14	X15	U13	dX13
1	0	1	0	1	0	0	1	1	0	0	1	0	*	1	1	1
0	1	1	0	1	0	0	1	1	0	0	1	0	1	*	1	1
*	*	*	*	*	*	*	*	*	*	*	*	1	*	*	0	0

$$dX_{13}= x_1.\overline{x_2}.x_3.\overline{x_4}.x_5.\overline{x_6}.\overline{x_7}.x_8.x_9.\overline{x_{10}}.\overline{x_{11}}.x_{12}.\overline{x_{13}}.x_{15}.u_{13}$$

$$V\ \overline{x_1}.x_2.x_3.\overline{x_4}.x_5.\overline{x_6}.\overline{x_7}.x_8.x_9.\overline{x_{10}}.\overline{x_{11}}.x_{12}.\overline{x_{13}}.x_{14}.u_{13}$$

Table 3.8: Operation model base loaded position

Precondition														Input	Post condition
X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	X13	X14	U13	dX14
1	0	1	0	1	0	0	1	1	0	0	1	0	1	1	0
1	0	1	0	1	0	0	1	1	0	0	1	1	0	0	1

$$dX_{14}= x_1.\overline{x_2}.x_3.\overline{x_4}.x_5.\overline{x_6}.\overline{x_7}.x_8.x_9.\overline{x_{10}}.\overline{x_{11}}.x_{12}.\overline{x_{13}}.x_{14}.u_{13}$$

$$V\ x_1.\overline{x_2}.x_3.\overline{x_4}.x_5.\overline{x_6}.\overline{x_7}.x_8.x_9.\overline{x_{10}}.\overline{x_{11}}.x_{12}.x_{13}.x_{14}.u_{13}$$

Table 3.9: Operation model pallet area position

Precondition														Input	Post condition
X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	X13	X15	U13	dX15
0	1	1	0	1	0	0	1	1	0	0	1	1	0	0	1
0	1	1	0	1	0	0	1	1	0	0	1	0	1	1	0

$$dX_{15} = \overline{x_1}.x_2.x_3.\overline{x_4}.x_5.\overline{x_6}.\overline{x_7}.x_8.x_9.\overline{x_{10}}.\overline{x_{11}}.x_{12}.x_{13}.\overline{x_{15}}.\overline{u_{13}}$$

$$V\ \overline{x_1}.x_2.x_3.\overline{x_4}.x_5.\overline{x_6}.\overline{x_7}.x_8.x_9.\overline{x_{10}}.\overline{x_{11}}.x_{12}.\overline{x_{13}}.x_{15}.u_{13}$$

Table 3.10: Definition of state pick and place robot

X1= Rotation A1 + position	U1= Sensor 2
X2= Rotation A1 - position	U2= Sensor 1
X3= Vertical A2 - position	U3= Sensor 4
X4= Vertical A2 + position	U4= Sensor 3
X5= Vertical A3 - position	U5= Sensor 5
X6= Vertical A3 + position	U6 = Sensor 6
X7= Rotation A4 + position	U7= Sensor 8
X8= Rotation A4 - position	U8= Sensor 7
X9= Vertical A5 + position	U9= Sensor 9
X10= Vertical A5 - position	U10= Sensor 10
X11= Rotation A6 + position	U11= Sensor 11
X12= Rotation A6 - position	U12= Sensor 12
X13= Gripper attach to the robot arm	U13= Sensor 13
X14= Base loaded position	
X15= Pallet area position	

Table above shows the pre and post condition for the pick and place robot that referring to the Figure 3.5, the changes that happens interpreted in the Table 3.1 until Table 3.9. Six axis robot and a gripper to hold the souvenir used in completing this process, for the axis A1, it will turn towards front and rear. At the same time the axis A2, A3 and A5 move upward and downward in vertical position. The axis A4 and A6 rotate according to + or – direction. The gripper clamp and unclamp the souvenir to achieve the requirement pick from base loaded position and place at pallet.

The operational model for A1 is represented in table 3.1.Table 3.2, Table 3.3, Table 3.4, Table 3.5 and Table 3.6 represent the logical model for A2, A3, A4, A5 and A6 respectively. Then, the gripper logical model represented in Table 3.7. At the same time, the logical model for the souvenir movement is based on Table 3.8 for the base loaded area while Table 3.9 for pallet area. The Boolean expression on Table 3.1 until Table 3.9 are based on the logic model of the pick and place robot, the model is based on the movement of the robot.

3.2.2 Block diagram process

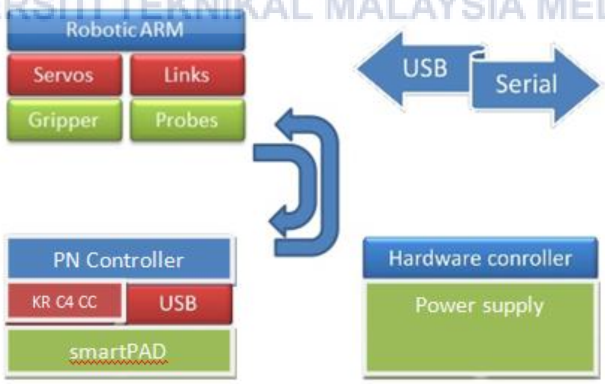


Figure 3.7: The block diagram above shows the working of the system

The PN controller design is the input to the system while the robotic arm movement is the output to the system. The inputs from the PN can upload to the smartPad by USB.

KR C4 control cabinet act as the neurones to the system which analyse the system before the robot operation starts. In term of real application, the input to the KUKA pick and place robot is to the teach pendant which is known as KUKA smartPAD. The robotic arm is the output and manipulator. Robot arm have servos, links, probes and gripper in order perform pick and place task. At the same time the hardware controller contains the power supply. The input and output interfaces in order to perform a specific task.

3.2.3 Hardware used in project

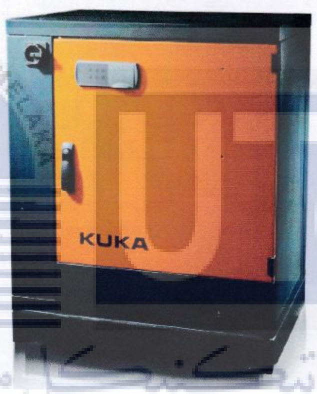


Figure 3.8: KR C4 robot controller for pick and place robot

It controls all the units in the robot. The controller instruct all the input unit, where it stores the data after receiving from user. It manages the flow of data and instructions from the storage to the ALU and result from the ALU to the storage. Typically, it is known as nervous system of the process.

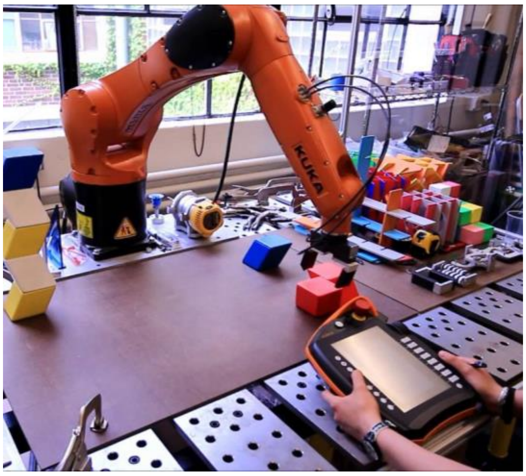
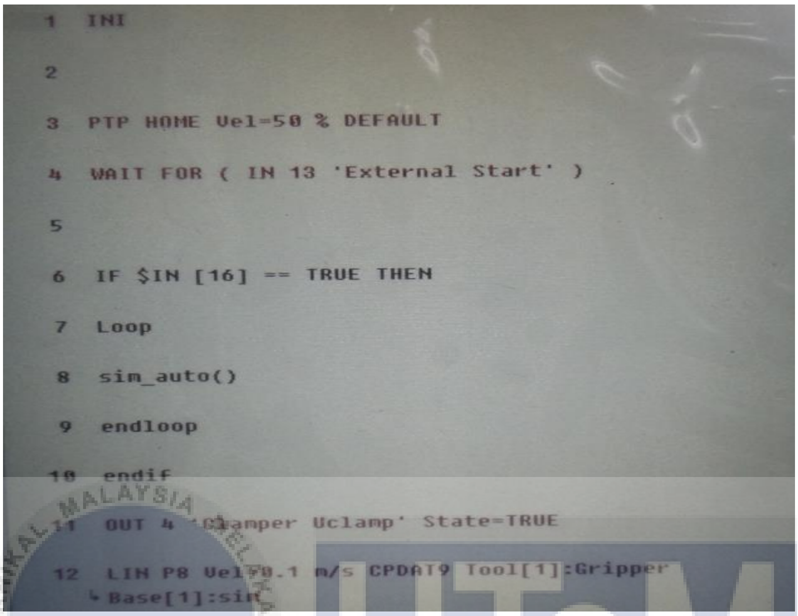


Figure 3.9: Smart PAD used to control the robot movement

The KUKA robot is operated by means of the KUKA smartPAD teach pendant. Features of the KUKA smartPAD are:

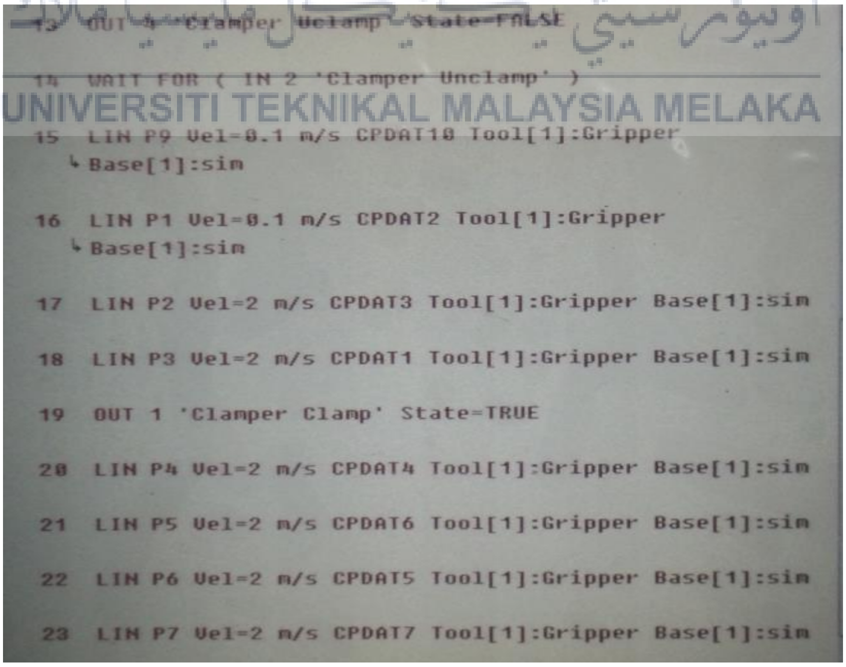
- Touch screen (touch-sensitive user interface) for operation by hand or using integrated stylus.
- Large display in portrait format
- KUKA menu key
- Eight jog keys
- Keys for operate control of the technology packages
- Program execution keys(stop/backwards/forwards)
- Keys for displaying the keypad
- Key switch for changing the operating mode
- Emergency stop button
- Space mouse
- Unplugable

3.2.4 KUKA robot program to control pick and place operation



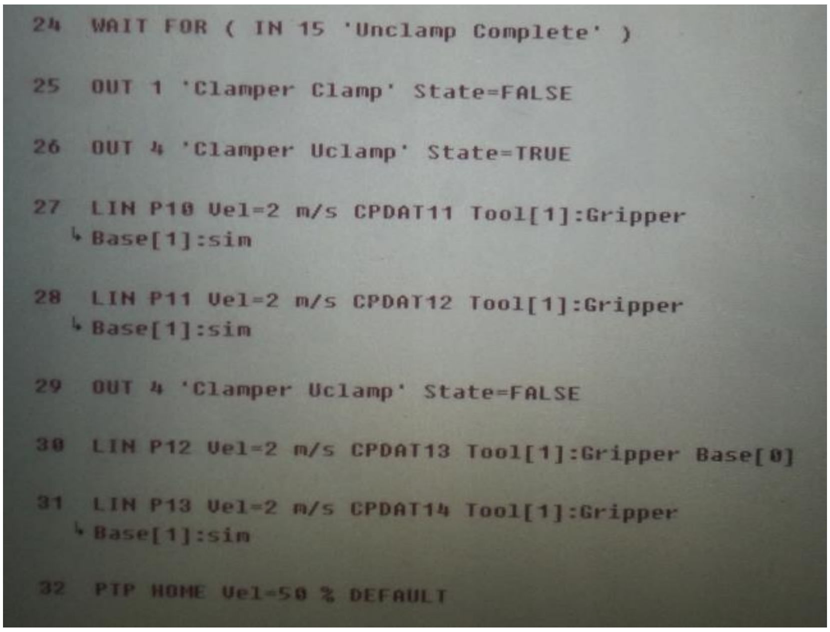
```
INI

PTP HOME Vel=50 % DEFAULT
WAIT FOR ( IN 13 'External Start' )

IF $IN [16] == TRUE THEN
Loop
sim_auto()
endloop
endif
OUT 4 'Clamper Uclamp' State=TRUE
LIN P8 Vel=0.1 m/s CPDAT9 Tool[1]:Gripper
  Base[1]:sim
```

Figure 3.10: SIM KUKA program (a)



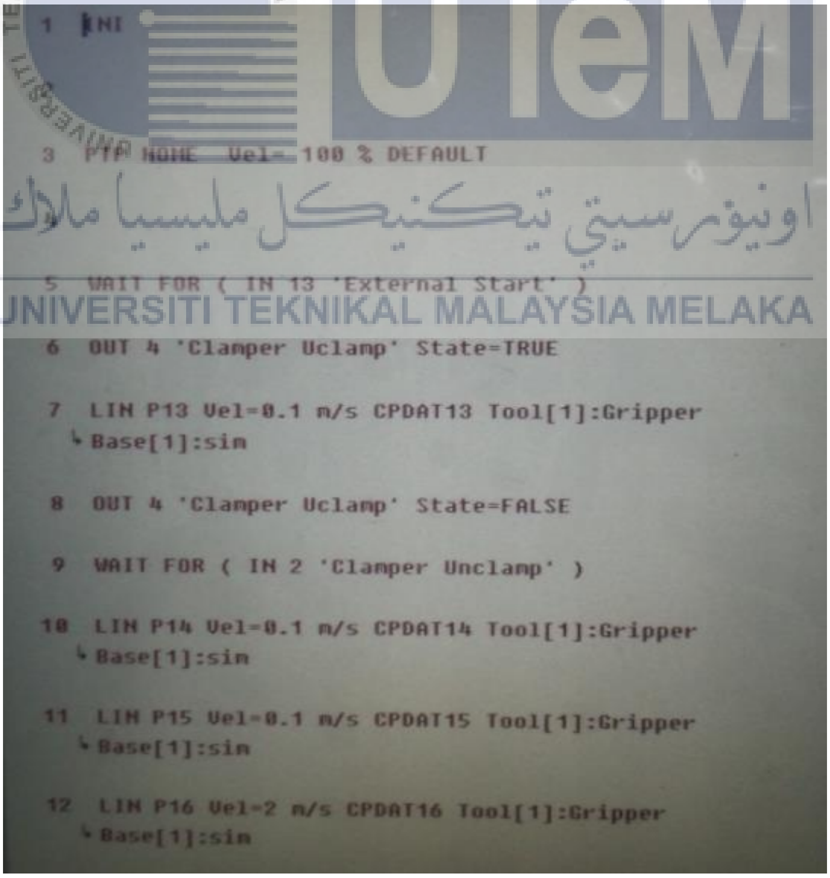
```
OUT 4 'Clamper Uclamp' State=FALSE
WAIT FOR ( IN 2 'Clamper Unclamp' )
LIN P9 Vel=0.1 m/s CPDAT10 Tool[1]:Gripper
  Base[1]:sim
LIN P1 Vel=0.1 m/s CPDAT2 Tool[1]:Gripper
  Base[1]:sim
LIN P2 Vel=2 m/s CPDAT3 Tool[1]:Gripper Base[1]:sim
LIN P3 Vel=2 m/s CPDAT1 Tool[1]:Gripper Base[1]:sim
OUT 1 'Clamper Clamp' State=TRUE
LIN P4 Vel=2 m/s CPDAT4 Tool[1]:Gripper Base[1]:sim
LIN P5 Vel=2 m/s CPDAT6 Tool[1]:Gripper Base[1]:sim
LIN P6 Vel=2 m/s CPDAT5 Tool[1]:Gripper Base[1]:sim
LIN P7 Vel=2 m/s CPDAT7 Tool[1]:Gripper Base[1]:sim
```

Figure 3.11: SIM KUKA program (b)



```
24  WAIT FOR ( IN 15 'Unclamp Complete' )
25  OUT 1 'Clamper Clamp' State=FALSE
26  OUT 4 'Clamper Uclamp' State=TRUE
27  LIN P10 Vel=2 m/s CPDAT11 Tool[1]:Gripper
    Base[1]:sim
28  LIN P11 Vel=2 m/s CPDAT12 Tool[1]:Gripper
    Base[1]:sim
29  OUT 4 'Clamper Uclamp' State=FALSE
30  LIN P12 Vel=2 m/s CPDAT13 Tool[1]:Gripper Base[0]
31  LIN P13 Vel=2 m/s CPDAT14 Tool[1]:Gripper
    Base[1]:sim
32  PTP HOME Vel=50 % DEFAULT
```

Figure 3.12: SIM KUKA program (c)



```
1   INI
3   PTP HOME Vel= 100 % DEFAULT
5   WAIT FOR ( IN 13 'External Start' )
6   OUT 4 'Clamper Uclamp' State=TRUE
7   LIN P13 Vel=0.1 m/s CPDAT13 Tool[1]:Gripper
    Base[1]:sim
8   OUT 4 'Clamper Uclamp' State=FALSE
9   WAIT FOR ( IN 2 'Clamper Unclamp' )
10  LIN P14 Vel=0.1 m/s CPDAT14 Tool[1]:Gripper
    Base[1]:sim
11  LIN P15 Vel=0.1 m/s CPDAT15 Tool[1]:Gripper
    Base[1]:sim
12  LIN P16 Vel=2 m/s CPDAT16 Tool[1]:Gripper
    Base[1]:sim
```

Figure 3.13: SIM_AUTO KUKA program (a)

```
OUT 1 'Clamper Clamp' State=TRUE

LIN P18 Vel=2 m/s CPDAT18 Tool[1]:Gripper
↳ Base[1]:sim

LIN P19 Vel=2 m/s CPDAT19 Tool[1]:Gripper
↳ Base[1]:sim

LIN P20 Vel=2 m/s CPDAT20 Tool[1]:Gripper
↳ Base[1]:sim

LIN P21 Vel=2 m/s CPDAT21 Tool[1]:Gripper
↳ Base[1]:sim

WAIT FOR ( IN 15 'Unclamp Complete' )

OUT 1 'Clamper Clamp' State=FALSE
```

Figure 3.14: SIM_AUTO KUKA program (b)

```
OUT 4 'Clamper Uclamp' State=TRUE

LIN P22 Vel=2 m/s CPDAT22 Tool[1]:Gripper
↳ Base[1]:sim

LIN P23 Vel=2 m/s CPDAT23 Tool[1]:Gripper
↳ Base[1]:sim

OUT 4 'Clamper Uclamp' State=FALSE

LIN P24 Vel=2 m/s CPDAT24 Tool[1]:Gripper Base[0]

LIN P25 Vel=2 m/s CPDAT25 Tool[1]:Gripper
↳ Base[1]:sim


PTP HOME Vel=50 % DEFAULT
```


Figure 3.15: SIM_AUTO KUKA program (c)

The program saved in the file SIM. The program execute the pick and place operation. The main program is SIM and the sub program is SIM_AUTO. SIM_AUTO execute a loop operation. The “INI” means the initialization of the program and it contains the calls of standard parameters that required for correct execution of the program. The motion command “PTP Home” is often used at the start and end of a program, as this is a known and clearly defined positions. The external start refers as the input from the PLC to start the program and sensor input from the conveyer base. The program will start once the base loaded and the sensor detects. Ṣ is indicate the KUKA variable. It shows that the program can be executed. The “END” defines the end of a program. The output from the execution is clamp or unclamp. The state true or false means it determine whether to execute or not base on the state given.

The operation of the program begins with an INI and home position of the robot arm. The robot cannot start operate if it is at home position. The velocity of the motion set to 50% from the maximum velocity. The maximum velocity will be 5m/s. Then, wait for external start. The input from the conveyer sensor is the external start. If true it enters the loop phase. The loop will execute the program in SIM_AUTO.

The program in SIM_AUTO is shown in figure 3.13, 3.14 and 3.14. After initiated and the robot at home position the robot move at 100% velocity which is 5m/s. it receives input from the external which is the PLC to start. The PLC act as the starting and ending point. Then the gripper is verified whether in unclamp state. If true then than the robot arm will move to the point P13, P14, P15, P16 and to point P17 before clamp. The output is send via two channel which is out 4 and out 1. Out 4 is to gripper unclamp while out 1 is to gripper to clamp. The order to execute the clamp and unclamp operation is as follows:

“Out 4 ==true and Out 1==false” to unclamp

“Out 4 ==false and Out 1==true” to clamp

The robot arm at P17 is where the base loaded. The gripper clamp the souvenir at base and start to move to point P18, P19, P20 and P21 before unclamp the souvenir at pallet area position. The robot arm move then move to point P22, P23, P24, and P25 before reach home position. The gripper must be in unclamp state at home position. It must agrees with both the input from actuator 1 and 4 to conduct clamping and unclamping. The velocity from point P13 to P15 is 0.1m/s while from P16 to P25 the velocity increased to 2m/s. The velocity can be adjusted according to the programmer. The type motion used is line to line (LIN). This motion requires the programmer to “teach” one point. The robot uses the point defined in the previous move as the start point and the point defined in the current command as the end point and interpolates a straight line in between the two points. The same operation will be conducted as it a loop program. The loop will end if the program SIM selected.

The routes used in SIM is different from SIM_AUTO. The point executed is from P1 to P13. The clamping and unclamping operation is done at point P9 and P7 respectively. The program for SIM is shown in figure 3.10, 3.11 and 3.12. The robot arm return to home position after unclamp the souvenir at pallet. The velocity varied from 0.1m/s to 2m/s. The tool centre position (TCP) of the tool 1 is guided from the starting point to the end point with constant velocity and a defined orientation. Maximum velocity of the robot is 5m/s but it is not preferred considering the safety purpose and any collision of the robot will cause severe damage. The motion must consider the safety of both the robot, operator and equipment around the robot.

3.2.5 PLC Ladder Diagram to control pick and place operation

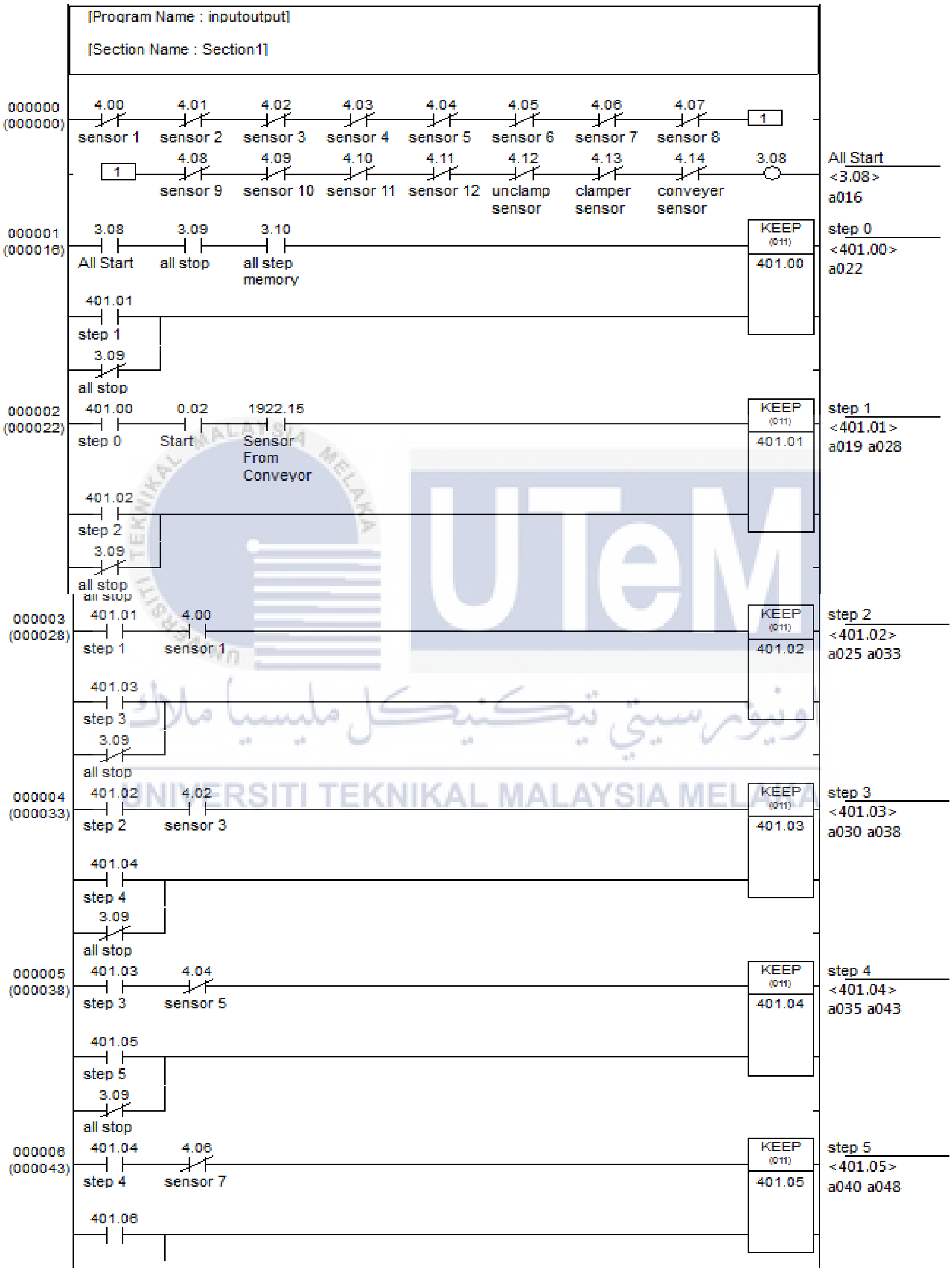


Figure 3.16: Ladder Diagram to control KUKA robot pick and place operation (a)

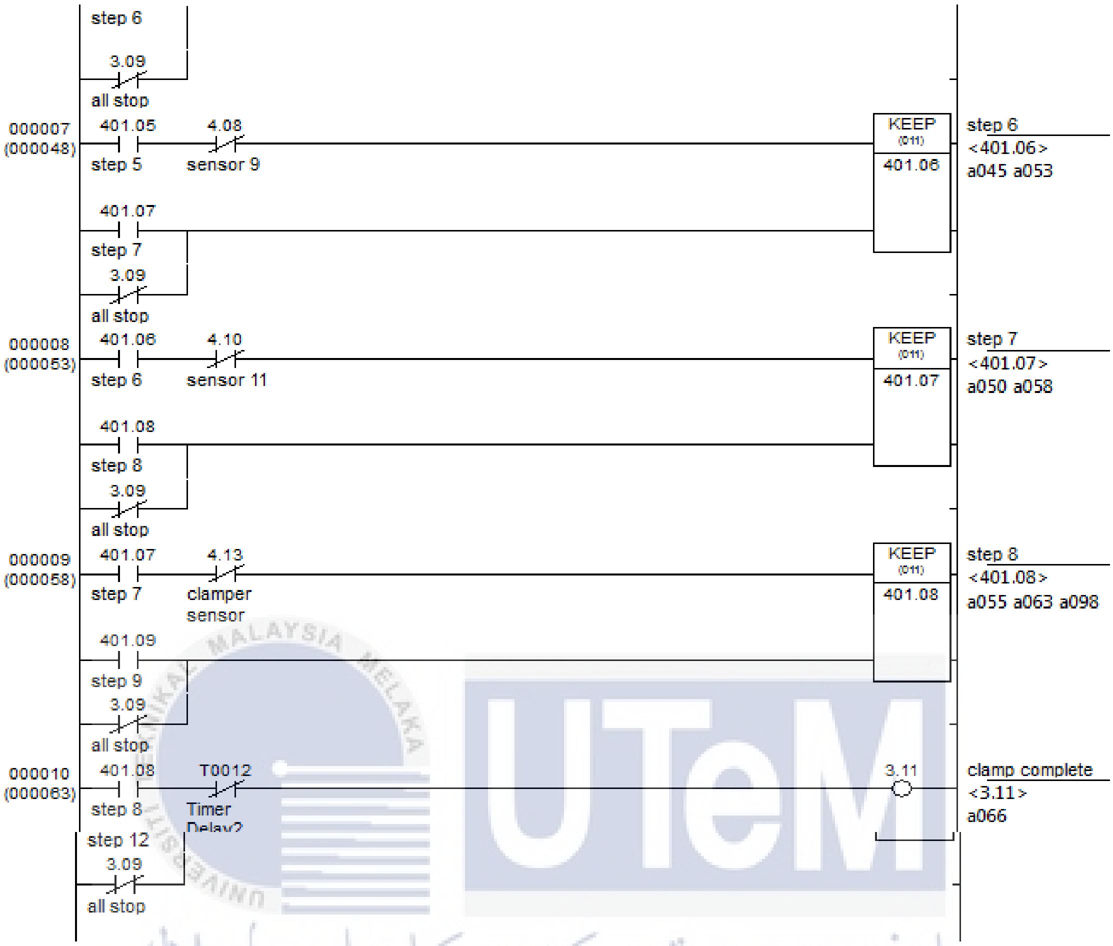


Figure 3.17: Ladder Diagram to control KUKA robot pick and place operation (b)

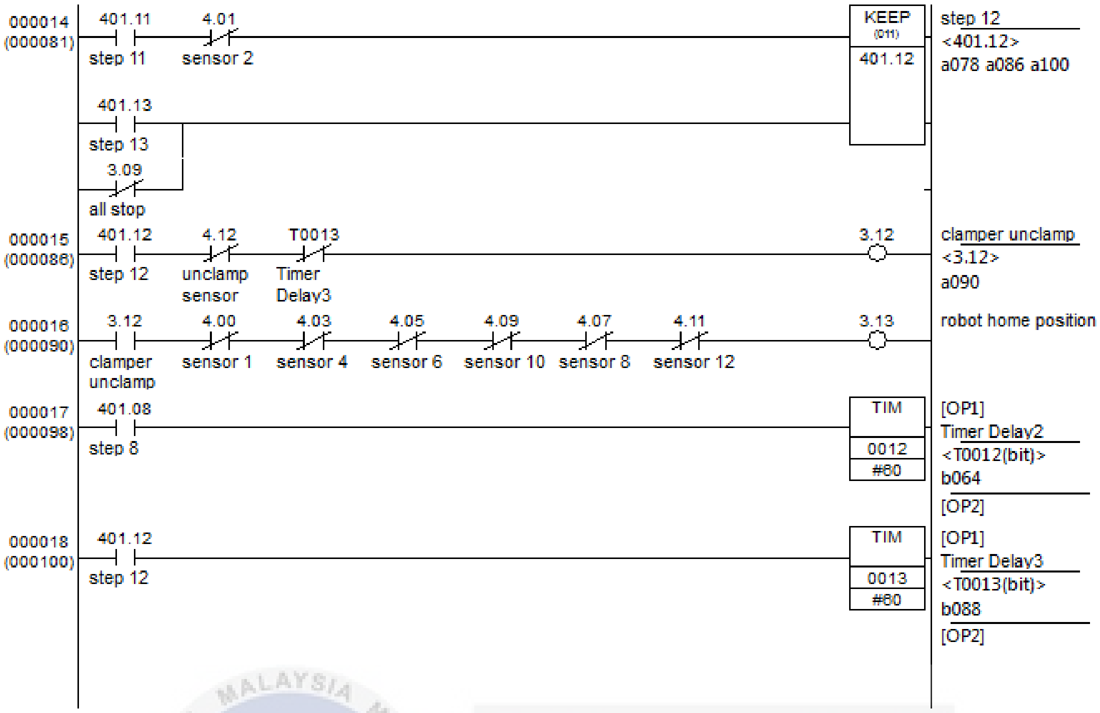


Figure 3.18: Ladder Diagram to control KUKA robot pick and place operation(c)

Figure 3.16, 3.17 and figure 3.18 shows the flow of sensors in order to control the movement of robot arm for pick and place system. The sensors used and the movement control by the sensor stated in Table 3.11.

Table 3.11: Sensor and control direction

Sensor	Axis and direction
Sensor 1	A1 + direction
Sensor 2	A1 - direction
Sensor 3	A2 + direction
Sensor 4	A2 - direction
Sensor 5	A3 + direction
Sensor 6	A3 - direction
Sensor 7	A4 + direction
Sensor 8	A4 - direction
Sensor 9	A5 + direction

Sensor 10	A5 - direction
Sensor 11	A6 + direction
Sensor 12	A6 - direction
Sensor 13	Unclamp
Sensor 14	Clamp
Sensor 15	Conveyer

The process starts with the all starts 3.08 where it only activated if sensor 1 to 14 are active. All start, all stop and all step memory initiated the keep 401.00 step 0. Step 1 initiated by step 0 and step 1 will initiate step 2. Step 1 also depend on the start button and sensor from conveyer. The process end at step 12 and clamping and unclamping occurred after step 8 and step 12 respectively. Each step initiated by the output of previous steps and end by the output of next step. All stop button help to stop the whole process in case of emergency. The keep function choose to conduct the steps because if a holding bit used for bit to set, the bit status will be retained even during power interruption. Keep can used to maintain status even after restarting the PLC following any power interruption and can used to maintain a flow. Once the gripper unclamp process complete the robot arm return to the home position. The delay between each clamp and unclamp process set to 60 seconds to give time for the process to occur. The LD in figure 3.16, 3.17 and 3.18 able to stand alone to control the pick and place process with the aid from sensors and motors.

3.3 Project methodology to achieve third objective.

This section discuss about the project methodology to achieve the third objective which is to design a controller for pick and place robotic system by using Petri Net.

3.3.1 Third objective process

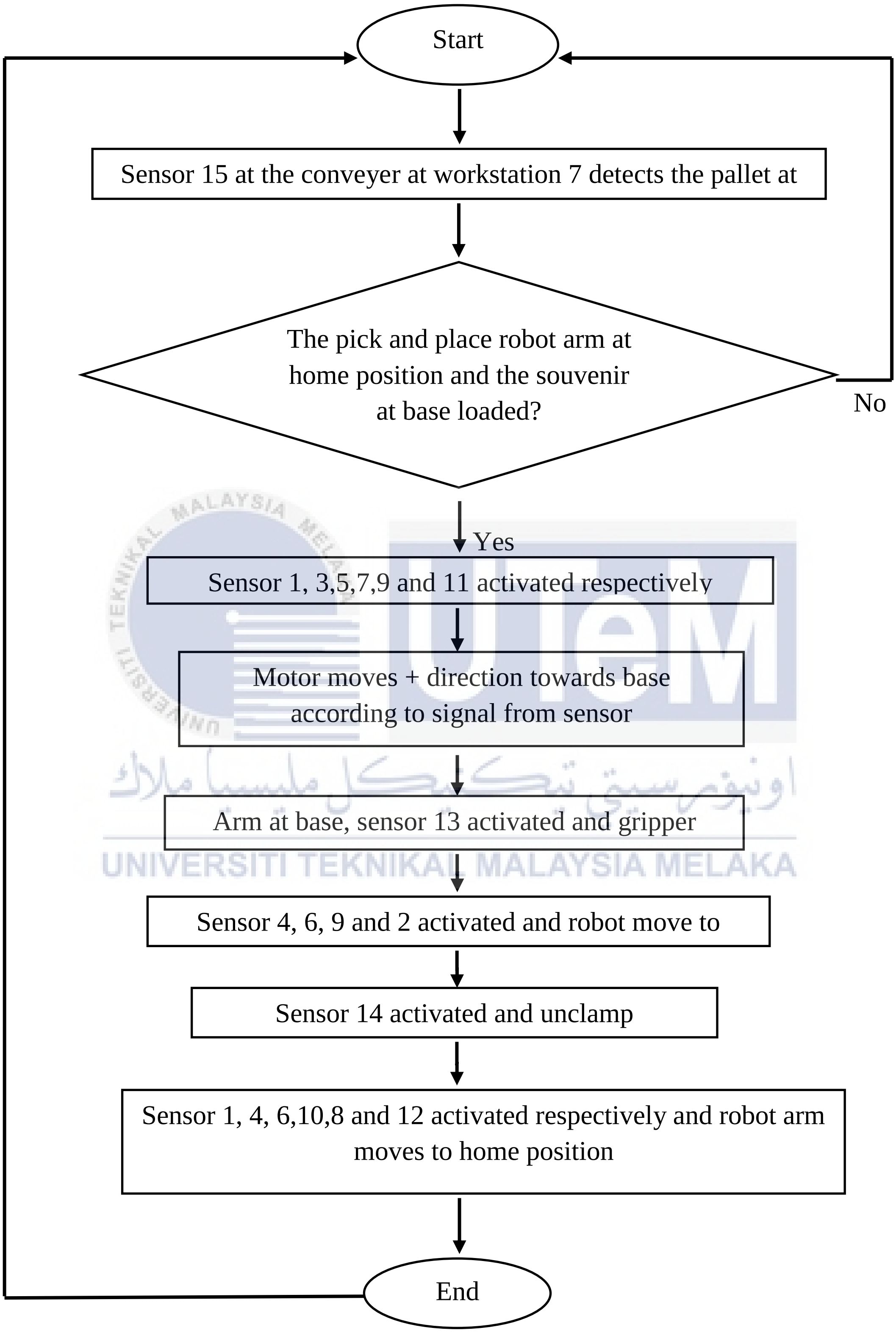


Figure 3.19: Flowchart for pick and place robot based on sensor

3.4 Equipment used in this project

Table 3.12: Equipment used in this project

EQUIPMENT	NO OF EQUIPMENT
KUKA pick and place robot	1
KUKA smartPAD	1
KR C4 Control Cabinet	1
Laptop	1

3.5 Project milestones and Gantt chart

The Table 3.13 shows the milestones that need to be achieved throughout the final year project (FYP).

Table 3.13: Project Milestones

No	Milestone	Date
1	Project analysis	16-9-2015
2	Identify Input/ Output	27-10-2015
3	Complete project proposal and presentation	7-11-2015
4	Controller Design and model of KUKA robot	22-3-2015
5	Complete software	10-4-2015
6	Analysing data	21-4-2015
7	Complete PSM report	15-5-2015

The Table 3.13 show the milestones that need to be achieved throughout the final year project (FYP). The milestone that will achieved throughout this project are project analysis, identifying input/output, complete project proposal and presentation, controller design and model of KUKA robot, Complete hardware with software, Analysing data and Complete PSM report.

Table 3.14: Gantt chart

NUMBER	ACTIVITY	2015/2016 (SEMESTER 7) FYP I																		2015/2016 (SEMESTER 8) FYP II																
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
FYP I		FYP Briefing									Mid Term Break						Study Week	Final Semester Examination	Semester Break								Mid Term Break									Final Semester Examination
1	Title Registration																																			
2	Literature Review and Library Research																																			
3	Submit Pre-proposal																																			
4	Log Book Preparation																																			
5	Proposal Preparation																																			
6	Analyze Software																																			
7	Design model																																			
8	Submit Report Proposal																																			
9	Presentation FYP 1																																			
FYP II																																				
10	Internet and Library research																																			
11	Design Simulation																																			
12	Identify Properties																																			
13	Testing and troubleshooting																																			
14	Prepare Final Report and Presentation																																			
15	Submit Final Report																																			
16	Final Presentation FYP II																																			

Table 3.14 shows the Gantt chart of the both FYP 1 and FYP 2. The Gantt chart shows the schedule of project flow to complete the whole project within the time given.

3.6 Conclusion

In summary, this project is start with the project analysis. The projects related to the PN in manufacturing and automation industries focused at first place. After that, introduced with the KUKA pick and place robot which is new to the disc casing loading assembly line. The operation of the robot is analysed to design a model using Boolean function. The controllers used in this robot also play a big part such as KUKA programming language and PLC. Then, the right PN installer obtained from internet source and tried to figure out the process and operation of PN. Finally, familiarize with the PN simulation. The flowchart and Gantt chart of research activities clarifies about the steps to be taken in order to accomplish the project without any delay. Last but not least, the methods to design controller for pick and place robot identified.

CHAPTER 4

RESULT AND DISCUSSION

4.0 Introduction

This chapter comprises of two important part of the project which are simulation to control the KUKA pick and place system and analysis of the design to satisfy the basic properties. The simulation model of the pick and place system design using PN simulation software. This chapter also covers about the result of objective.

4.1 Simulation Result

This section is about the explanation of the simulation result.

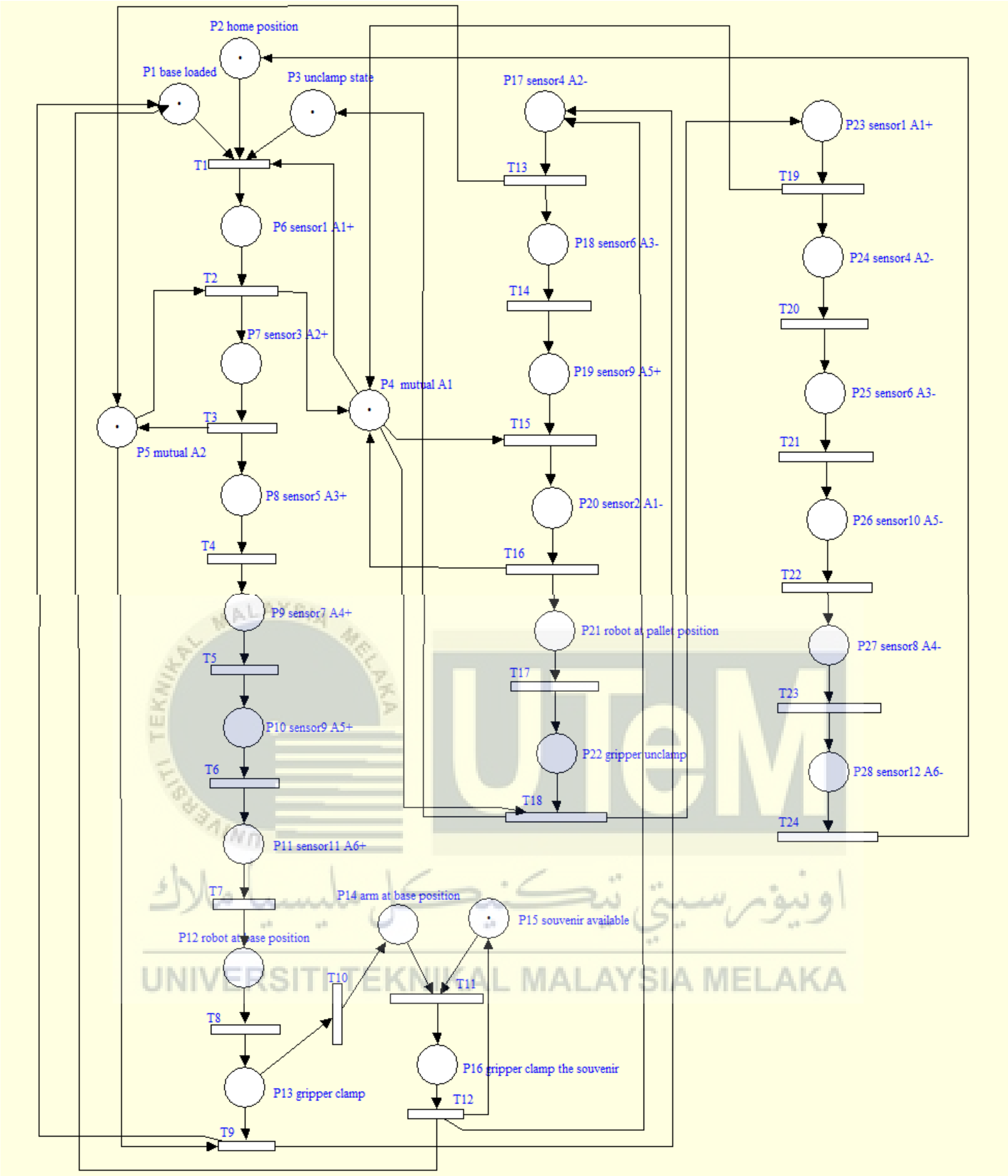


Figure 4.1: PN controller design for pick and place robotic system

Table 4.1: Meaning of places and transitions for PN controller design

Places	Transition
P1= Base loaded with souvenir	T1= Processing P1,P2,P3 and P4
P2= Robot home position	T2=Processing P6 and P5
P3= Gripper unclamp state	T3= Processing P7
P4= Mutual axis A1	T4= Processing P8
P5= Mutual axis A2	T5= Processing P9
P6= Sensor1, axis A1, + direction	T6= Processing P10
P7= Sensor3, axis A2, + direction	T7= Processing P11
P8= Sensor5, axis A3, + direction	T8= Processing P12
P9= Sensor7, axis A4, + direction	T9= Processing P13 and P5
P10= Sensor9, axis A5, + direction	T10= Processing P13
P11= Sensor11, axis A6, + direction	T11= Processing P14 and P15
P12= Robot at base position	T12= Processing P16
P13= Gripper clamp souvenir	T`13= Processing P17
P14= Robot arm at base position	T14= Processing P18
P15= Souvenir available	T15= Processing P19 and P4
P16= Gripper clamp souvenir	T16= Processing P20
P17= Sensor4, axis A2, - direction	T17= Processing P21
P18= Sensor6, axis A3, - direction	T18= Processing P22 and P4
P19= Sensor9, axis A5, + direction	T19= Processing P23
P20= Sensor2, axis A1, - direction	T20= Processing P24
P21= Robot at pallet area position	T21= Processing P25
P22= Gripper unclamp state	T22= Processing P26
P23= Sensor1, axis A1, + direction	T23= Processing P27
P24= Sensor4, axis A2, - direction	T24= Processing P28
P25= Sensor6, axis A3, - direction	
P26= Sensor10, axis A5, - direction	
P27= Sensor8, axis A4, - direction	
P18= Sensor12, axis A6, - direction	

4.1.1 Pick and place robotic model using PN

The system used to control the tasks such as pick the souvenir from conveyer and place it at pallet area position. With following on the activities of the six axis robot arm in the system, the PN model design using the following steps.

The robot perform repeatedly the following operation: picking, moving, placing and moving. To use in the common workspace each robot axis has to request first before get access to and leaves the common workspace. The resources include the six axis robot, souvenir, gripper conveyer and pallet. Since the robot has six axis the conflict in controlling each axis without interfacing has to be considered. Thus, it can be avoided using the mutual exclusion structure. Thus when one sensor is doing a job other sensor of the same axis prohibited from interrupting. The relations among these operations and events for each axis are sequential. The sensors movement are mutually exclusive thus the robot collision can be avoided.

Due to the precedence relation between picking, moving, placing and moving the arcs add from Ti1 to Pi2 and to Ti2. Since enabling ti1 requires the availability of the base loaded, unclamp state and robot home position. Continue this process till all the relations are represented. Finally, the tokens add to all resources places to signify the initial system state. The use of token from pi1 and the arc from ti9 back to pi1 models actually the case that components to be inserted will be available over the concerned period. The final model is shown in figure 4.3. The initial marking is:

$$(1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)T$$

The net has a total of 28 places and 24 transitions. Figure 4.2 shows the PN controller design for pick and place six axis robotic system. The design made in Figure 4.1 used two important nodes which is places and transitions. Table 4.1 shows the meaning of each places and transition used in the PN controller design. Places are used to represent the condition or status of a component and transitions represent the events or operation. They are pictured by empty rectangles or solid bars such as T1to T24. Two common events are “start” and “end.” Instead of bidirectional links in some physical nets, a PN utilizes directed arcs to connect from places (called input places with respect to a transition) to transitions or from transitions to places (called output places). In other words, the information transfer from a place to a transition or from a transition to a place is one-way. Two- way transfer between a place and transition is achieved by designing an arc from a transition to a place and another arc from the transition back to the place.

Places, transitions, and directed arcs make a PN a directed graph, called the Petri net structure. The dynamics is introduced by allowing a place to hold either none or a positive number of tokens pictured by small solid dots as shown in P1, P2, P3, P4, P5 and P15. These dots could represent the number of resources or indicate whether a condition is true or not in a place. When all the input places hold enough number of tokens, an event modeled by a transition can happen, called transition firing. This firing changes the token distribution in the places, signifying the change of system states.

The number of place used are 28 while the number of transitions are 24. The techniques used to control the pick and place robotic system are mutually exclusive, sequential, error recovery and cyclic. There are 12 sensors used to control six axis robot movement with each axis controller by 2 sensors. The robot axis direction varied according to + and – direction. The design include starting at home position, pick the souvenir at base loaded position , place the souvenir at pallet area position and return to home position. These stages are designated with flow and different places used to represent different sensor in order to avoid conflict among sensors in execution.

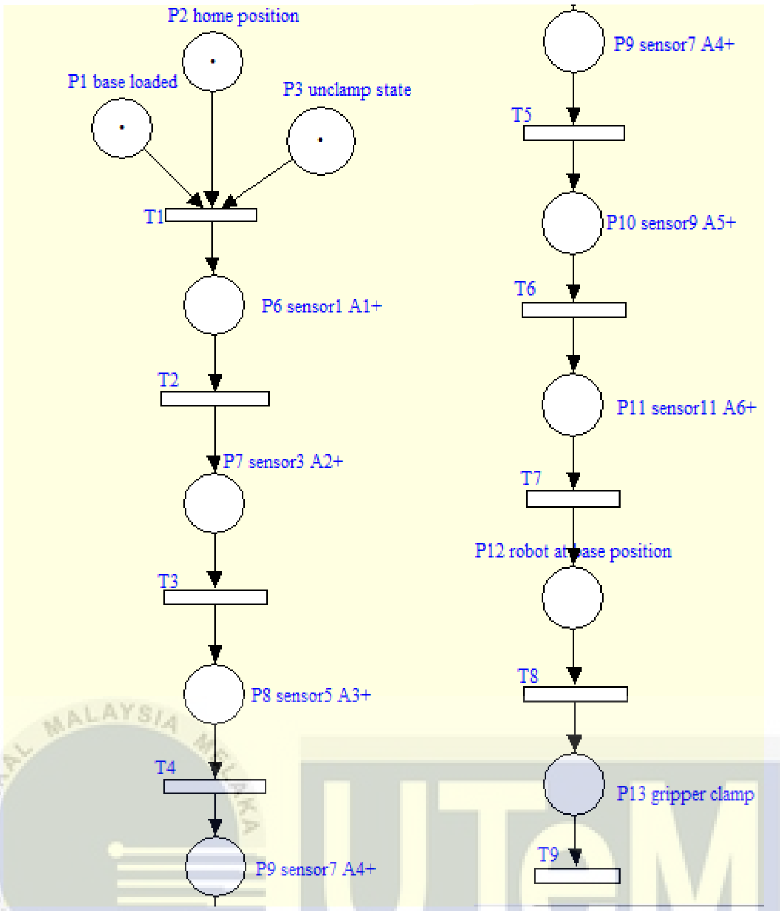


Figure 4.2: Souvenir clamping process

Souvenir clamping process shown in figure 4.2 initiated by three tokens which is robot at home position at P2, gripper in unclamp state at P3 and the base loaded at P1. Otherwise the process cannot be initiated or must wait until these three conditions satisfied to start a firing through T1 and gain output token at P6. The output will be activation of sensor A1+ but three tokens required to start process. P6 identified as sensor 1 which enable axis A1 to move to + direction. Then it pass through T2 to P7. P7 is sensor 3 which control the movement of A2 in the direction of +. After that, the token flow through P8, P9, P10, and P11 which triggered by sensor 5, 7, 9 and 11 respectively before reach the base area position at place P12. The gripper change from unclamp state to clamp state to clamp the souvenir. Once the gripper clamped the souvenir, the base will be empty and loaded again to pick other souvenirs. This a cyclic process where the process run continuously.

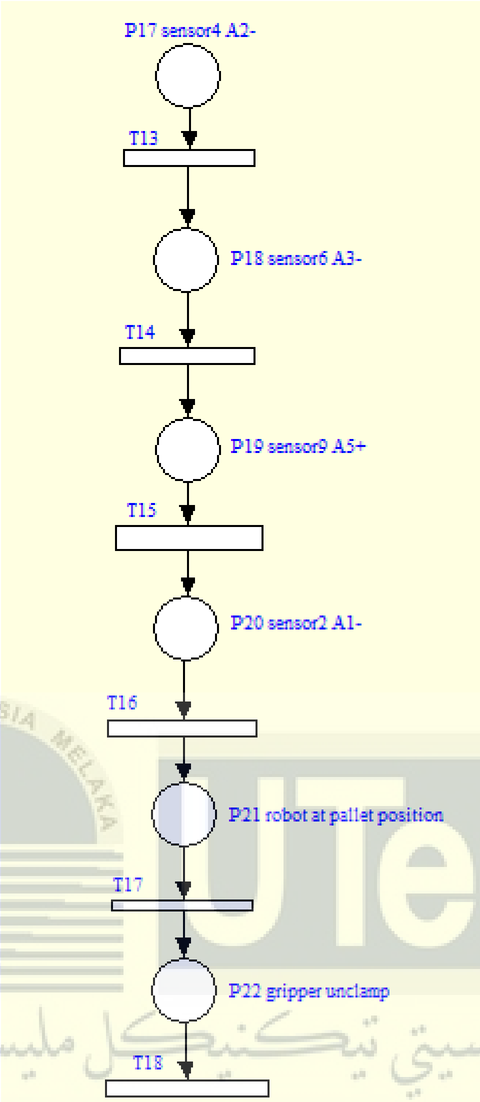


Figure 4.3: Souvenir unclamping process

Souvenir unclamping process shown in figure 4.3 is a sequential flow from figure 4.2. The gripper which hold the souvenir carries the load to pallet area position before unloading. The token pass through P17, P18, P19 and P20 triggered by sensor 4, 6, 9 and 2 respectively. The robot achieve the pallet area position at P21 and stop. Gripper unclamp the souvenir at P22 and fired by T18 to P3 which is unclamp state before make it available for another cyclic process.

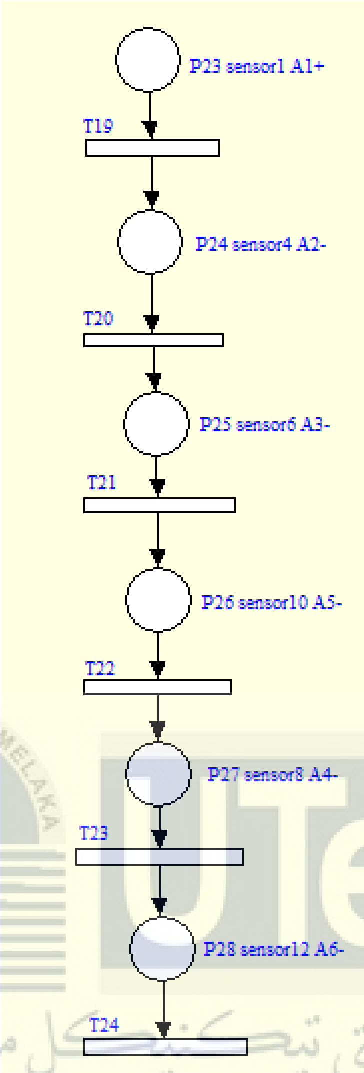


Figure 4.4: Return to home position

Figure 4.4 shows the process of robotic arm to return to home position. Sensor 1 and sensor 4 activated at P23 and P24 where A1 moves in + direction and A2 moves in – direction respectively, then sensor 6, 10, 8 and 12 activated at P25,26, P27 and 28 to move the axis A3, A5, A4 and A6 respectively in – direction. T24 fire the input from P28 to P2, robot arm reach at home position to execute another pick and place process.

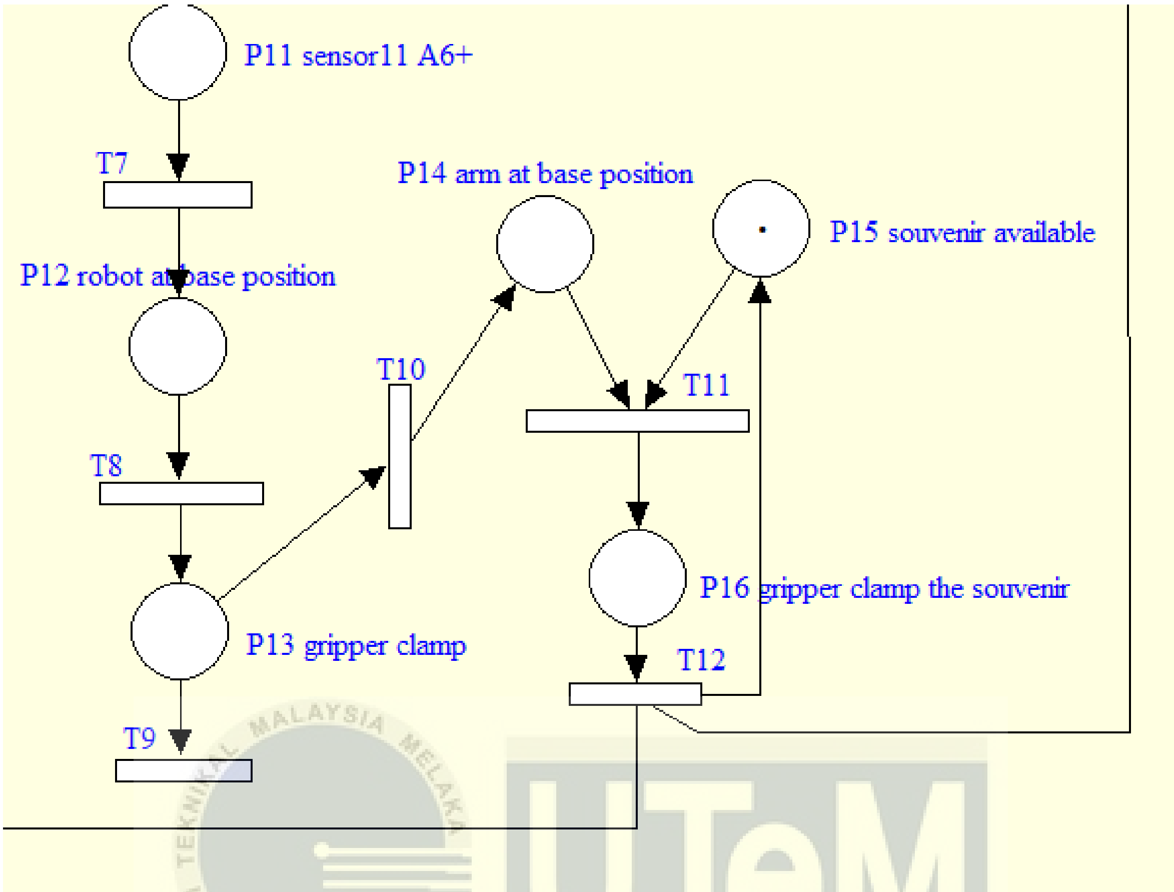


Figure 4.5: Error recovery process

If there is an error in clamping the souvenir from base loaded position the robot have to stop and restart from starting point to ensure a smooth flow but error recovery process shown in figure 4.5 help to recover the error without restarting from starting point. If there is an error occurred at P13 in clamping process, T10 initiate and fire through arcs to P14 where a similar process executed. Robot return to the base position to clamp the souvenir and P15 with a token shows the souvenir availability and execute the clamping process at P16. The number of error recovery process recorded at P5 to check the efficiency of the design. If there is an error the error recovery system will recover from the error and record the number of times the error occurred. The output from recovery process fired to P17 to execute the unclamping process.

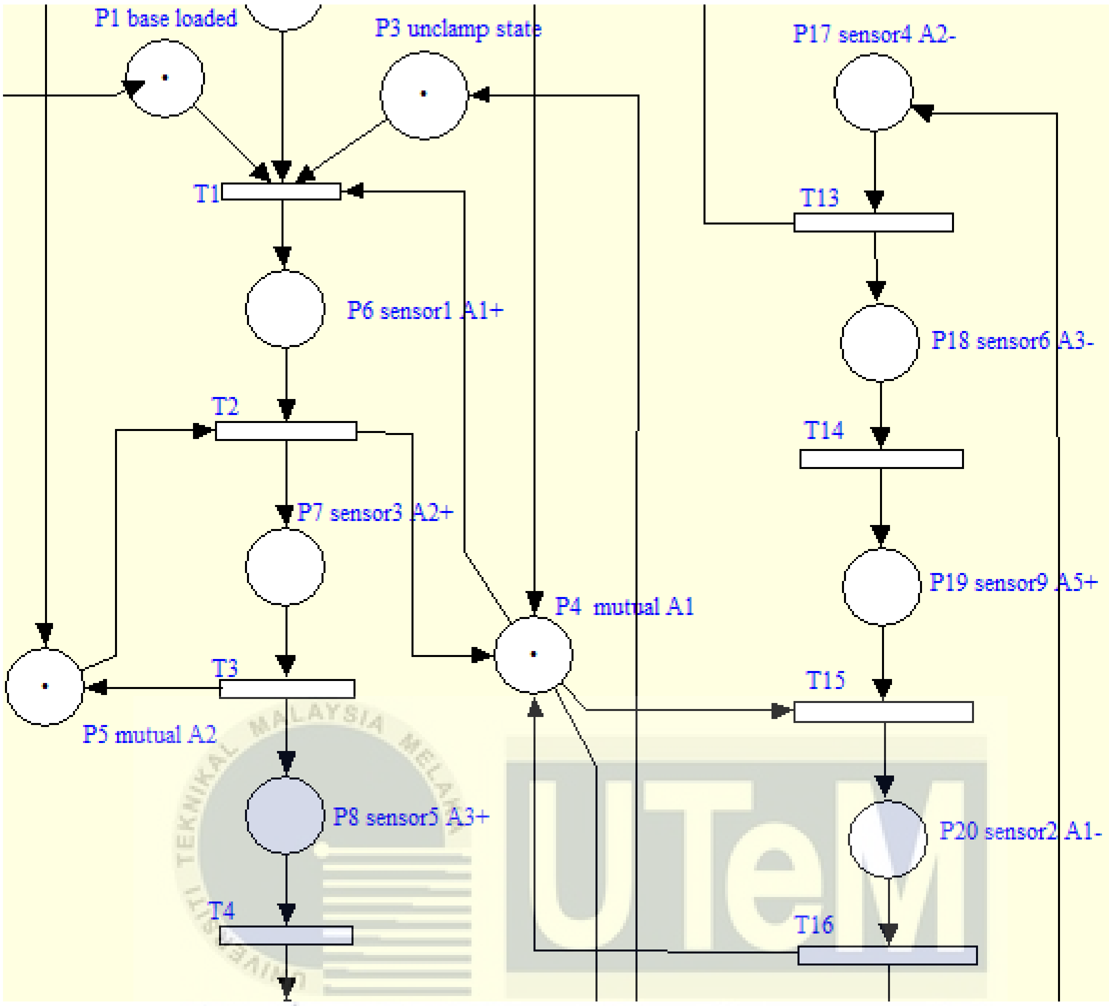


Figure 4.6: Mutually exclusive

The technique used to control the process flow in figure 4.6 is a mutually exclusive process. Two processes are mutually exclusive if they cannot be performed at the same time due to constraints on the usage of shared resources. A structure to realize this is through a common place marked with one token plus multiple output and input arcs to activate these processes. P4 allows sensor 1 or 2 to activate to move axis A1 either in + or – direction at a time. Mutually exclusive technique allow the robot arm execute the pick and place process without conflict between sensors.

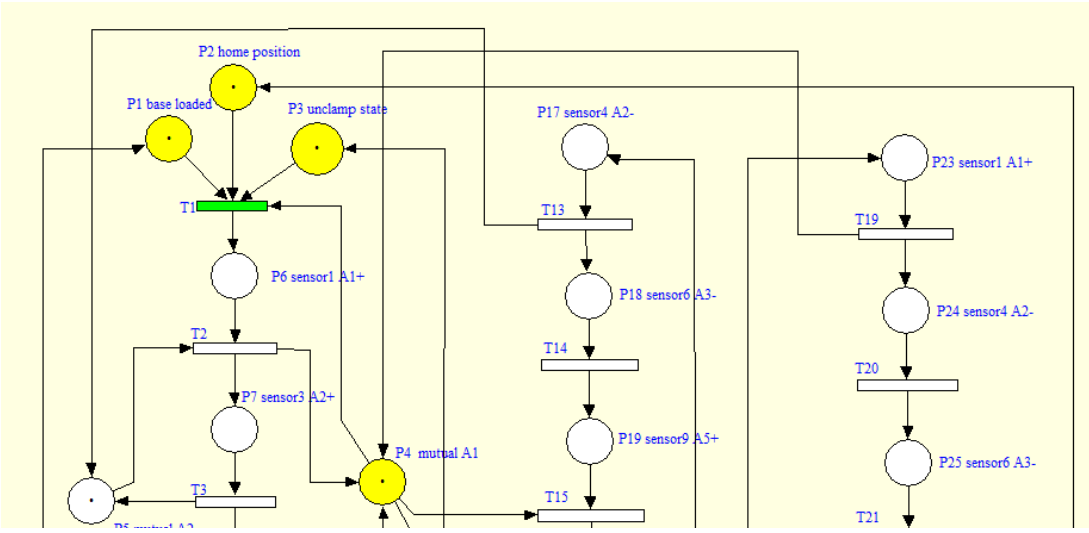


Figure 4.7: Simulating process

Simulation starts as in figure 4.7 with the presence of three tokens which are base loaded at P1, arm at home position at P2 and gripper unclamp state at P3. The robot pick and place process cannot starts without the presence of these three tokens. This ensure the safeness and boundedness of the system. The tokens ready for firing through transition T1 and the output be the input at place P6. P1, P2 and P3 become the input while P6 is the output for the event T1. Transition T1 only can fired if place P4 available with token. The token at P4 bounded with sensor 1 and 2. The movement of A1 either in + or – direction decided by the transition. The collision and conflict between sensors and motor rotation avoided through mutual exclusive method as P4.

4.1.2 The eight tuple

Real time Petri net can be obtained by associating timing, I/O sensory information to the untimed PN's and defined as follows:

An RTPN is an eight tuple and defined as: RTPN = (P, T, I, O, m**,** D, X, Y) where:

1. P is a finite set of places
2. T is a finite set of transitions with P U $T \neq$ 0 and P ∩ T=0

3. *I: P* x *T* →N, is an input function that defines the set of directed arcs from P to T where N = (0, l, 2,. . .}
4. O: P x T→ N, is an output function that defines the set of directed arcs from T to P
5. m: P→ N**,** is a marking whose i^{th} component represents the number of tokens in the i^{th} place. An initial marking is denoted by mo.
6. D: T→ R+, is a firing time function where *R+* is the set of nonnegative real numbers.
7. X: P→ {-, 0. 1, 2 . . . . *K }* and X(pi) ≠ X(pi),i≠ *j,* is an input signal function, where *K* is the maximum number of input signal channels, and "-" is the dummy attribute indicating no assigned channel to the place.
8. Y: T→ L is an output signal function, where *L* is a set of integers.

4.1.3 Design of conversion between LD and PN

The controller design for pick and place robotic system using PN is shown in figure 4.1 and figure 3.12 shows the design in LD. The modification of each design and the input and output for each design shown in table 4.1 for PN and table 3.1 for LD. The physical appearance of PN is preserved from the first sequence to the last sequence. This is not true for the LD case as the changing in property is not preserved. The design reveals that PN design and LD design do not differ much when the control sequence is relatively simple, as seen in the figure 3.13. In fact, the PN model appear more complex than LD at first sight, as shown for the sequence in figure 3.13. But, when this sequence is modified to result in a complex one, PN are more easily modified and hence maintainable than LLD. In LLD, nodes appear multiple times which may lead to difficulty in understanding the logic and cause errors when developing the logic. LLD needs more basic elements to model timers and counters compared to PN. The following are experienced during implementation of sequence of controllers using LD and PN:

1. The control logic can be qualitatively analysed to check the properties such as absence of dead locks using PN. Cannot do qualitative analysis until it is simulated or implemented.

2. PN allows dynamically track the system with the help of states of places and transition. The states can be obtained through reachability graph as shown in figure 4.14.
3. The initial state of the system can be directly represented by its initial marking in PN.

4.2 Model checking with Petri Nets simulator

In model simulator analysis, Petri nets properties verify that modeling token into complex calculation to simplify result. The result from simulator can be used to ensure the liveness, safeness, boundedness and others properties included. Figure below shown the example of the program from Petri nets simulator generate from bipartite graph to analyze the pick and place robot controller design.

```
Integrated Net Analyzer [v2.2p6-Mar 23 2001-win32] session report:
Current net options are:
    token  type: black        (for Place/Transition nets)
    time option: no times
    firing rule: normal with capacities
    priorities : not to be used
    strategy   : single transitions
    line length: 255

Net read from robot_4.pnt
Information on elementary structural properties:
Current name options are:
     transition names to be written
     place names to be written
Static conflicts:
transition 1.T1 is in conflict with:
      15.T15, 18.T18,
transition 2.T2 is in conflict with:
       9.T9,
transition 9.T9 is in conflict with:
       2.T2, 10.T10,
transition 10.T10 is in conflict with:
       9.T9,
transition 15.T15 is in conflict with:
       1.T1, 18.T18,
transition 18.T18 is in conflict with:
       1.T1, 15.T15,

The net is not statically conflict-free.
The net is pure.
The net is ordinary.
The net is homogenous.
The net is not conservative.
The net is not subconservative.
The net is not a state machine.
The net is not free choice.
The net is not extended free choice.
The net is not extended simple.
The net is marked.
The net is not marked with exactly one token.
The net is not a marked graph.
The net has a non-blocking multiplicity.
The net has no nonempty clean trap.
The net has no transitions without pre-place.
The net has no transitions without post-place.
The net has no places without pre-transition.
The net has no places without post-transition.
Maximal in/out-degree:  4
```

Figure 4.8: Properties analysis of pick and place controller design

The figure 4.8 analyse the system properties of PN controller design. The properties of the PN controller design shown in analyser. The information regarding the transition conflict, connectivity status, place capacities, input matrix and output matrix can be viewed through the analyser. The report from the analyser shows the type of token used, firing rule, strategy, priorities, information on elementary structural properties and the net properties. Elementary structural properties written based on the conflicts between transitions. Figure 4.8 shows that T1 is in conflict with T15 and T18 which means these three transitions shares a common place which is place P4. T1, T15 and T18 fired depend on the input from P4. P4 is mutually exclusive with three transitions. Other than that, net has details about the post places and pre places. All the information regarding the design can be obtain from the properties analyser.

4.2.1 Matrix Place/Transition

Matrix (P/T) | Properties | Reachability graph | P - invariants | T - invariants

	1. T1	2. T2	3. T3	4. T4	5. T5	6. T6	7. T7	8. T8	9. T9	10. T10	11. T11	12. T12	13. T13	14. T14	15. T15	16. T16
1. P1 base loaded	-1								1			1				
2. P2 home position	-1															
3. P3 unclamp state	-1															
4. P4 mutual A1	-1	1													-1	1
5. P5 mutual A2		-1	1						-1				1			
6. P6 sensor1 A1+	1	-1														
7. P7 sensor3 A2+		1	-1													
8. P8 sensor5 A3+			1	-1												
9. P9 sensor7 A4+				1	-1											
10. P10 sensor9 A5+					1	-1										
11. P11 sensor11 A6+						1	-1									
12. P12 robot at base position							1	-1								
13. P13 gripper clamp								1	-1	-1						
14. P14 arm at base position										1	-1					
15. P15 souvenir available											-1	1				
16. P16 gripper clamp the sou											1	-1				
17. P17 sensor4 A2-									1			1	-1			
18. P18 sensor6 A3-													1	-1		
19. P19 sensor9 A5+														1	-1	
20. P20 sensor2 A1-															1	-1
21. P21 robot at pallet position																1
22. P22 gripper unclamp																
23. P23 sensor1 A1+																
24. P24 sensor4 A2-																
25. P25 sensor6 A3-																
26. P26 sensor10 A5-																
27. P27 sensor8 A4-																
28. P28 sensor12 A6-																

Figure 4.9: Place and transition in matrix

Figure 4.9 shows the representation of place and transitions in matrices. There are a total of 28 places and 24 transition. Input and output of each transition and place shown in figure above. -1 means one token is fired at current transition and 1 shows the place receive a new token. For example at transition T1 four tokens fired which are from place P1, P2, P3 and P4 and the output token stored in P6 as 1. The output from T1 to P6 become the input to P6 when firing through T2.

4.2.2 Reachability

Matrix (P/T) | Properties | Reachability graph | P - invariants | T - invariants

```
State nr.      1
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   1   1   1   1   1   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0
==t1=> s2
State nr.      2
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   0   0   0   0   1   1   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0
==t2=> s3
State nr.      3
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   0   0   0   1   0   0   1   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0
==t3=> s4
State nr.      4
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   0   0   0   1   1   0   0   1   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0
==t4=> s5
State nr.      5
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   0   0   0   1   1   0   0   0   1   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0
==t5=> s6
State nr.      6
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   0   0   0   1   1   0   0   0   0   1   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0
==t6=> s7
State nr.      7
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   0   0   0   1   1   0   0   0   0   0   1   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0
==t7=> s8
State nr.      8
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   0   0   0   1   1   0   0   0   0   0   0   1   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0
==t8=> s9
State nr.      9
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   0   0   0   1   1   0   0   0   0   0   0   0   1   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0
==t9=> s10
==t10=> s22
State nr.     10
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   1   0   0   1   0   0   0   0   0   0   0   0   0   0   1   0   1   0   0   0   0   0   0   0   0   0   0   0
==t13=> s11
State nr.     11
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   1   0   0   1   1   0   0   0   0   0   0   0   0   0   1   0   0   1   0   0   0   0   0   0   0   0   0   0
==t14=> s12

toks:   1   0   0   1   1   0   0   0   0   0   0   0   0   0   1   0   0   0   1   0   0   0   0   0   0   0   0   0
==t15=> s13
State nr.     13
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   1   0   0   0   1   0   0   0   0   0   0   0   0   0   1   0   0   0   0   1   0   0   0   0   0   0   0   0
==t16=> s14
State nr.     14
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   1   0   0   1   1   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   1   0   0   0   0   0   0   0
==t17=> s15
State nr.     15
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   1   0   0   1   1   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   1   0   0   0   0   0   0
==t18=> s16
State nr.     16
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   1   0   1   0   1   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   1   0   0   0   0   0
==t19=> s17
State nr.     17
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   1   0   1   1   1   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   1   0   0   0   0
==t20=> s18
State nr.     18
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   1   0   1   1   1   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   1   0   0   0
==t21=> s19
State nr.     19
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   1   0   1   1   1   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   1   0   0
==t22=> s20
State nr.     20
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   1   0   1   1   1   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   1   0
==t23=> s21
State nr.     21
P.nr:   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
toks:   1   0   1   1   1   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   1
==t24=> s1
```

Figure 4.10: Reachability of pick and place controller design

4.2.2.1 Reachability Analysis Method

The figure 4.10 shows the reachability graph for pick and place system. Starting from the initial system condition or state, it is desired to derive all the possible states the system can reach, as well as their relationship. The resulting representation is called the reachability tree or graph. Then all the behaviour properties discussed above can be discovered if the number of states is finite. The method toward enumeration of all states or markings in Petri nets for analysis of their properties is called a reachability or coverability analysis method.

The change in state of places after each firing shown clearly. There are 24 transition firing starts at T1 and end at T24 before return to initial state. The maximum number of token given to each place is one. The firing sequence and cyclic.

There are two strategies to generate and label all the markings of a marked Petri net: depth-first and breadth-first. In the depth-first strategy, starting from the initial marking, identify all the enabled transitions, firing one of them (randomly) resulting in a "new" marking. If it is an old or dead marking, stop exploring it, come back to the marking generating it, and continue with those unfired transitions. Otherwise, at this new marking, again identify all the enabled transitions, firing one of them resulting in a "new" marking. Continue the above process until all the enabled transitions are fired and all the markings are generated if the number of markings is finite.

The second strategy, breadth-first, identifies all the enabled transitions, firing all of them resulting in "new" markings. For each marking, if it is an old or dead marking, go to the next marking. Otherwise, identify all the enabled transitions and fire them to generate all the "new" markings, and then go to next marking until the markings at the same level are exhausted. Then start the next level of markings.

The resulting representation consists of nodes or markings and the arcs. An arc links from one node to another and is labelled with its corresponding fired transition. The representation is termed as a reachability or coverability tree. Elimination of duplicate markings from the tree leads to a reachability or coverability graph.

4.2.3 P-Invariant

```
Matrix (P/T) | Properties | Reachability graph | P - invariants | T - invariants
place invariants basis of net 0.robot_4          :
=================
Nr.          1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

   1 |       1   0   0   0   0   1   1   1   1   1   1   1   1   1  -1   0   0   0   0   0   0   0   0   0   0   0   0   0
   2 |       1   0   0   0   0   1   1   1   1   1   1   1   1   1   0   1   0   0   0   0   0   0   0   0   0   0   0   0 +
   3 |       1   0  -1   0   0   0   0   0   0   0   0   0   0   0   0   0  -1  -1  -1  -1  -1  -1   0   0   0   0   0   0
   4 |       0   0   0   1   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   1   0   0   0   0   0 +
   5 |       0   1  -1  -1   0  -1   0   0   0   0   0   0   0   0   0   0   0   0   0  -1   0   0   0   1   1   1   1   1

@
At least the following places are covered by semipositive invariants:
       1,  4,  6,  7,  8,  9, 10, 11, 12, 13, 14, 16, 20, 23,

Non-reachability test of the state:
  p :  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
m(p):  1  1  1  1  1  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0

No conclusions possible.


@
```

Figure 4.11: Place invariant

Place invariant is state where place unaltered or the state does not change under a transformation. The non-reachability test indicates the initial state of marking before firing occur. The places P2, P3, P5, P15, P17, P18, P19, P21, P22, P24, P25, P26, P27 and P28 intercept with two transitions only.

4.2.4 T-invariant

```
Matrix (P/T) | Properties | Reachability graph | P - invariants | T - invariants
transition invariants basis of net 0.robot_4          :
======================
Nr.          1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

   1 |       1   1   1   1   1   1   1   1   1   0   0   0   1   1   1   1   1   1   1   1   1   1   1   1 +

@
At least the following transitions are covered by semipositive invariants:
       1,  2,  3,  4,  5,  6,  7,  8,  9, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24,


@
```

Figure 4.12: Transition invariant

T-invariant is state where transition unaltered or the state does not change under a transformation. All the transitions covered by semi positive invariant except P10, P11, and P12. These transitions cause a change in the output if there is an error detected. P10, P11, and P12 only called to recover the system to its right state. The other transitions have at least one input from places to be fired and obtain a semi positive invariant state.

4.2.5 Safeness and Boundedness

The safeness of the design using PN ensured with a test of flow of the system. Safeness of the system threaten only when overflow occurred or any attempt in request more execution of ongoing operation occurred. When a place models an operation, its safeness guarantees that the controller have no attempt to initiate an ongoing process. The safeness of the system is ensured by assigning capacity of each place with a certain amount. The places in the controller design assign with a capacity of one for each place. If there is two token in a place then the overflow detected at that place. Other than that the safeness ensured by simulation techniques which carried out namely mutually exclusive. This technique allows only one place or sensor can be activated at a time to control the same axis. The concept of boundedness is often interpreted as stability of a discrete manufacturing system when it is modelled as a queuing system. Safeness is 1-boundedness. These two properties are behavioural.

4.2.6 Liveness

Liveness is considered as the absence of deadlocks. This property guarantee that all places can be act and it make the operation or the condition which is modeled can happen in evolution of process. This property is relate to the safety that always in good condition and verification will check the properties being specify during the executions. The controller design for pick and place robotic system in figure 4.2 is have a liveness property.

The sequence of firing transition T1 reaches a marking that enables T2. A PN is live if every transition in it is live. A transition T is dead if there is no sequence of transition firing to enable another T starting from m. A transition is dead when it contains deadlock at which no transition is enable and called ass dead marking. The deadlock only happen when there is inappropriate resource allocation or exhaustive use of some or all resources. The deadlock can be viewed if the system is jammed or two or more processes are in circular chain, each of which waits for resources held by the process next in the chain. The PN design in Figure 4.2 is live and it has no deadlocks.

4.2.7 Reversibility

The design for robotic pick and place system in figure 4.2 is reversible. The reversibility is a cyclic behaviour of a system that performs at its function repeatedly and recoverable. The initial state of place P1, P2 and 3 recoverable after the execution of the system. Place P1 recover to its initial state after transition T9 as the gripper clamp the souvenir from the base and make the base available to fill with new souvenir. Place P2 return to its home position after transition 24 as the arm place the souvenir at pallet area position. Place P3 obtain its initial state to token one after transition T18 as the gripper unclamp the souvenir at place P22. This shows the reversibility of the design as it recovered to the initial state and perform the same operation repeatedly. Cyclic sequence technique allows the operation to be carried out repeatedly.

4.2.8 Reachability Graph

$$mo = (1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$$

$$\text{T1}\downarrow$$

$$m1 = (0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$$

T2 ↓

$$m2 = (0,0,0,1,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0)$$

T3 ↓

$$m3 = (0,0,0,1,1,0,0,1,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0)$$

T4 ↓

$$m4 = (0,0,0,0,1,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0)$$

T5 ↓

$$m5 = (0,0,0,1,1,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0)$$

T6 ↓

$$m6 = (0,0,0,1,1,0,0,0,0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0)$$

T7 ↓

$$m7 = (0,0,0,1,1,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0)$$

T8 ↓

$$m8 = (0,0,0,1,1,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0)$$

…………

…………

T24 ↓

$$mo = (1,1,1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0)$$

Figure 4.13: Reachability graph

Figure 4.13 shows the reachability graph for the pick and place robotic controller design. Firing of transitions until transition T8 shown in graph above. Each state after the firing marked with an m' marking. Tokens of each place changes after each firing and return to initial marking after transition T24. The marking mi is reachable from marking mo if there exist a sequence of transition firings which transforms mo to mi. Reachability checks whether a system can reach a specific state, exhibiting particular behaviour. The design satisfy the reachable property because it is able to execute pick, place and return to home position operation which is the initial state.

CHAPTER 5

CONCLUSION AND RECOMENDATION

5.0 Conclusion

The Petri Net is an excellent mathematical modelling software which invented by Carl Adam Petri. It will be an effective controller for robotic pick and place system. The basic elements are places and transitions in form circle and rectangle enable new beginners understand the function and operation of PN easier and faster. High level methods and techniques used by other controllers such as PLC would cause extremely difficult situation when refers to the design complexity and adaptability. PN used as controller in many fields but it is still new as the controller for robots. The reason it stood as the solo controller compared other controllers in controlling the robot operation are the simpler design and its adaptability. Other than that, PN contains input and output system which depends on the condition and event. The condition which means input and event is the output. The operation of robot must satisfy the input which is a condition to get the output or an event to occur. The methods used to design an efficient controller are cyclic, sequence, concurrent and mutually exclusive. These techniques allow the inputs to meet the outputs without interruption between sensors which controlling same axis. The comparison made between LLD and PN clearly shows the advantage of PN over one of the famous robotic controller in current situation. Apart from that, the basic operation of KUKA pick and place robot identified to construct model based on Boolean function. The KUKA pick and place robot controlled by PLC program and KUKA programming language where the teach pendant can be used to control the movement of each axis manually. The controller

KRC4 acts as the neurons to the system. KUKA pick and place robot operates by picking any object that it detects by gripping and place it at the location or place specified. The movement of pick and place robot can be more linear and can reduce the fault in accurateness by PN controller. The basic work of pick and place robot and simulation using PN identified in PSM 1. The simulation using PN includes identifying the method of designing controller in various situations. Pick and place robot operated manually.

The three objective of this project is completed in PSM2. The model for robotic pick and place system using Boolean function created. The inputs and outputs assigned according to the axis. An effective controller design for robotic pick and place system accomplished using Petri net. The design will satisfy less design complexity and high adaptability to the changes to the current situation. The design also satisfies the properties such as liveness (freedom from deadlock), boundedness (no capacity overflow), reachability (have flow), reversibility (cyclic behaviour) and safeness (no overflow) to ensure that the process can initiate the initial marking and return to the initial marking after the execution. Their analysis method based on reachability, invariant and simulation. Reachability analysis technique can help to check all the properties of bounded PN. The axis controlled by different sensors and the conflict among the sensors to control certain axis avoided by using PN controller techniques.

5.1 Recommendation

Further implementation of PN based controllers for other machines and robots in industries must be done to encourage the use PN in real time purpose. Currently most of the manufactures using PLC as a controller for their machines. Researchers, industrial engineers and robot manufacturers have to convince that PN is an effective and worthy controller which is applicable in their industries. For the RTPN, it can be extended by adding more attributes to places and transitions in order to control complex hierarchical manufacturing systems that use advanced communication protocols and several computer for control. The present study also can be extended to control all the eight stations simultaneously.

REFERENCE

[1] John 0. Aloody, Panos J. Antsalclis, and hfichael D. Lemnion," Automated design of a Petri net feedback controller for a robotic assembly cell" Journal of Electrical Engineering(1995)

[2] Prof.Krishna Moorthy Sivalingam. [nptelhrd]. (2012, july 25). Mod-01 Lec-39 Petri Nets-1 [Video file]. Retrieved from http:// www.youtube.com/watch?v=aNnTEeLeMCw [Accessed: 23 December 2015]

[3] Venkatesh K, MengChu Zhou, Caudill, R.J., "Comparing Ladder Logic Diagrams and Petri Nets for Sequence Controller Design Through a Discrete Manufacturing System" Journal of industrial electronics (1994)

[4] Jiacun Wang," Petri Nets for Dynamic Event-Driven System Modeling"(2005)

[5] Feldmann, K., Colombo, A.W., Schnur, C., Stockel, T.,"Specification, Design, and Implementation of Logic Controllers Based on Colored Petri Net Models and the Standard IEC 1131 Part II: Design and Implementation" Control Systems Technology (1999)

[6] Suchit Moon. (2014, Sep 28). Pick and place robot [image]. Retrieved from http:// www.slideshare.net/suchitmoon/pickandplacerobot [Accessed: 20 February 2016]

[7] Adriano Jose´ Cunha deAguiar , Emı´lia Villani , Fabrı´cio Junqueira ,” Coloured Petri nets and graphical simulation for the validation of a robotic cell in aircraft industry” Robotics and Computer-Integrated Manufacturing(2011)

[8] MengChu Zhou, Mu Der Jeng,”Modeling, Analysis, Simulation, Scheduling, and Control of Semiconductor Manufacturing Systems: A Petri Net Approach” journal of transactions on semiconductor manufacturing (1998)

[9] Training Documentation, KUKA Roboter GmbH (version O2KSS8).(2012). Robot operation PRO:KUKA system software 8.x

[10] Hafner, I., Breitenecker, F., Korner, A.” Learning Petri Net Dynamics through a Matlab Web Interface” journal of modelling and simulation (2013)

[11] MengChu Zhou, Frank Dicesari.”Adaptive Design of Petri Net Controllers for Error Recovery in Automated Manufacturing Systems” journal of transactions on systems, man and cybernetics (September/ October 1989)

[12] Bruno Lacerda, Pedro U.Lima, Javi Gorostiza, Miguel A. Salichs” Petri Net Based Supervisory Control of a Social Robot with LTL Specification” Proceedings of Robotica 11 (2011)