FAKULTI KEJURUTERAAN ELEKTRIK

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

LAPORAN PROJEK

SARJANA MUDA

A FORMAL METHOD FRAMEWORK FOR AUTOMATED VERIFICATION OF A DEAERATOR SYSTEM

Nur Amirah Binti Othman

Bachelor of Electrical Engineering (Control,Instrumentation and Automation)

June 2014

“ I hereby declare that I have read through this report entitle “A Formal Method Framework for Automated Verification of A Deaerator System” and found that it has comply the partial fulfillment for awarding the degree of Bachelor of Electrical Engineering (Control, Instrumentation and Automation)”

Signature : ……………………………………………

Supervisor’s Name : DR SAIFULZA BIN ALWI @ SUHAIMI

Date : 9 JUNE 2014

A FORMAL METHOD FRAMEWORK FOR AUTOMATED VERIFICATION OF A DEAERATOR SYSTEM

NUR AMIRAH BINTI OTHMAN

A report submitted in partial fulfillment of the requirement for the degree of Bachelor of Electrical Engineering (Control, Instrumentation and Automation)

Faculty of Electrical Engineering

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2014

I declare that this report entitle "*A Formal Method Framework for Automated Verification Of A Deaerator System*" is the result of my own research except as cited in the references. The report has not been accepted submitted in candidature of any other degree.

Signature : ……………………………………………………………………………

Name : NUR AMIRAH BINTI OTHMAN

Date : 9 JUNE 2014

Specially dedicated:

To my beloved father Othman bin Mohamad Noor,

To my beloved mother Nor Haizan binti Embong,

My beloved sister and brothers,

My supervisor and all my lecturers,

All my friends.

For their encouragement, support and motivation through my journey of education.

ACKNOWLEDGEMENT

Bissmillahirrahmanirrahim,

Alhamdulillah. Many grateful and thanks to Allah SWT for His continuous blessing and giving me the consent to complete this Final Year Project. This Final Year Project was organized by Faculty of Electrical Engineering (FKE), Universiti Teknikal Malaysia Melaka (UTeM) for student in final year to complete the undergraduate program in Bachelor of Electrical Engineering Major in Control, Instrumentation and Automation with Honors.

First of all, my sincere gratitude goes to my supervisor, Dr Saifulza bin Alwi @ Suhaimi to give me the opportunity to be under his supervision for the Final Year Project. Many thanks for the encouragement, suggestions and guidance in achieving the goal and maintain the progress in track. The discussions and the meetings give me a lot of new knowledge to explore regarding to my project. Next, I also want to thanks to all the lecturers of FKE, UTeM for their support and motivation to complete the project.

Many thanks to my friends, Munirah Binti Mohd Siraj, Mohd Mohaimin bin Miswon and Nurrafidah binti Mohammad Rashid who help me and give me the suggestion and motivation to make my best for this project. Not to forget to all my classmates in 4 BEKC and other friends,thank you very much for the support and concern to my final year project directly or indirectly. This four years experience will be remembered as an important memory before enter the new chapter of line as an engineer soon.

Last but not least, deepest thanks and appreciation to my parents for their moral supports, love, sacrifice throughout my life. I am thankful for their sacrifice, patience, and understanding during completing this project. Their sacrifice had inspired me from day I learned to write and read until I am now.

ABSTRACT

Deaerator is important equipment in feed water system of a power plant. The role of deaerator system is to remove dissolved gases which are oxygen and carbon dioxide that comes from the water leaving of condenser and to give the adequate level of water to the deaerator storage tank. In deaerator system, the flow of steam and water has their own principles which are the flow of steam before supply to the deaerator storage tank and the flow of water from condenser flow before supply to the boiler. The principle of deaerator system must in the correct order to make sure it is in the safe condition to the plant system. Thus, formal verification of correctness of a property is used as an approach to ensure all the specification created meets the actual behavior for the system. All the specifications must always *hold* during the verification process to ensure that the model designed will not violate. The verification procedure is also need to eliminate the errors that decrease the safety of the automation system. Hence, in this project, it will show on how the computational method such as temporal logic model checking can be used to verify the correctness of the design of an automation system. The project involves the deaerator model and the design of the ladder diagram (LD) using Programmable Logic Controller (PLC). The project used Computational Tree Logic (CTL) as the temporal logic to determine the specification. By using several logical specifications to the deaerator system, the designed model of deaerator model and control logic should verify so that it will not violate the required specification. If none of the behaviors of the system violates the given specification, the model of the system will correct. Otherwise, the model checker will automatically execute the counterexample of the model system to show why the specification is false. The verification of the system will be performed by using Symbolic Model Verify (SMV) model checker software.

ABSTRAK

Deaerator ini memberikan peranan penting dalam sistem air suapan loji kuasa. Peranan sistem deaerator adalah untuk membuang gas-gas terlarut iaitu oksigen dan karbon dioksida daripada air pemeluwap dan menghantar air yang mencukupi ke tangki simpanan deaerator . Dalam sistem deaerator , aliran wap dan air mempunyai prinsip-prinsip mereka sendiri yang merupakan aliran wap sebelum bekalan kepada tangki simpanan deaerator dan aliran air dari aliran kondenser sebelum disalurkan kepada tangki dandang. Prinsip sistem deaerator mesti dalam susunan yang betul untuk memastikan ia berada dalam keadaan yang selamat untuk sistem kilang. Oleh itu , pengesahan rasmi kebenaran ciri-ciri sistem yang digunakan sebagai pendekatan untuk memastikan semua spesifikasi yang dicipta memenuhi kelakuan sebenar untuk sistem itu. Semua spesifikasi mesti sentiasa berada pada tempat yang betul bagi memastikan model yang direka tidak akan melanggar . Prosedur pengesahan juga perlu untuk menghapuskan kesilapan-kesilapan yang mengurangkan keselamatan sistem automasi. Oleh itu, dalam projek ini , ia akan menunjukkan bagaimana kaedah pengiraan seperti duniawi model logik semakan boleh digunakan untuk mengesahkan kebenaran reka bentuk sistem automasi. Projek ini terdiri daripada model Deaerator dan reka bentuk gambarajah tangga (LD) Format menggunakan Programmable Logic Controller (PLC) . Projek ini menggunakan pengiraan Tree Logic (CTL) logik duniawi untuk terjemahan spesifikasi. Dengan menggunakan beberapa spesifikasi yang logik untuk model Deaerator itu, tingkah laku model yang disahkan dan hasil pengesahan digambarkan untuk operasi yang selamat. Jika tiada tingkah laku sistem melanggar spesifikasi yang diberi, model sistem akan membetulkan. Jika tidak, pemeriksa model secara automatik akan melaksanakan penyangkal sistem model untuk menunjukkan mengapa spesifikasi itu adalah palsu.Pengesahan terhadap sistem itu akan dilakukan dengan menggunakan Model simbolik Sahkan (SMV) perisian model pemeriksa.

TABLE OF CONTENTS

LIST OF TABLE

LIST OF FIGURE

LIST OF SYMBOLS

| , ∨	-	OR gate
&	-	AND gate
~	-	NOT gate
->	-	THEN

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

This chapter briefly discussed about the motivation and the problem of this project. Besides, the objective and the project scope should be achieved at the end of the project.

1.1 Motivation

In power plant industry, steam power is the largest sector of the electrical generating industry and without it, the world will be in a big trouble. Electricity we use nowadays comes from the generation of steam from the power plant industry for example in thermal power plant, coal power plant and others.

In many years, the use of formal method in software development would help the industry meets the criteria or goals for their system as the software system in much important to engineering field. In the other disciplined of formal method approaches, such as object in structured design, it offer a methods to simplify the view of the system especially for the designer and the customer used. Formal method is a method that described the view or the behavior of the system immediately. Mathematical approached is very important to describe the behavior of a system and it will help the engineer to create the right system and help to create the system to right.

Thus, by all, the main motivation by doing this project is to enable applying the tools and techniques to check the property of reachability and resettability. It is important technique to make a checking in order to make sure the system not violate to the specification.

1.2 Problem Statement

Deaerator of a power plant is one of the important components for feed-water system. It is a feed-water heater which is widely used for remove entrained air that contained oxygen and carbon dioxide, CO_2. The example of deaerator that usually used in the power plant is Boilermate deaerator, Spraymaster deaerator and others. The deaerator system can control the steam pressure and water level of deaerator storage tank. In the deaerator system, it requires the heat which is comes from the steam at desired operating value. The water and steam will agitated together and remove the dissolved gases. The steam has high pressure steam that need to be reduced to maintain the desired pressure of deaerator operating pressure. Accurate pressure control is very important to get the saturation temperature of deaerator. The failure to maintain the steam pressure and temperature will cause too little inlet steam to remove the entrained air.

Besides remove the entrained air, the deaerator also must to keep the deaerator storage tank roughly half levels full of water to make sure enough supply feed water to the boiler. Too low water level makes the pump cavitations and the boiler shut down. Hence, adequate level control is very important in deaerator system. Hence, by all the behavior and principle flow of the deaerator system, the application of formal verification via model checking can be checked by using the software system. It is very important to check the correctness of the mathematical model of the model designed because in the plant system, the smallest change in the input will result to the different change in the output of the system. The verification is also required to eliminate design errors that decrease the safety of the system and to check the system from enters the undesired states. Thus, this proposed project is to introduce a formal method framework for automated verification of the deaerator system based on the prescribed logical assumption.

1.3 Objective of the Project

1. To determine the behavior of a deaerator system for the purpose of transformation to logical behavior.
2. To model the mathematical expression of deaerator system based on the prescribed logical assumption.
3. To verify the logical behavior of deaerator control system through Symbolic Model Verifier model checker software based on the Boolean model.

1.4 Project Scope

In power plant, there are several different types of design for deaerator which are steam flow pressurized deaerator and Boilermate deaerator. For this project, the Boilermate deaerator is used as to describe the behavior of deaerator system. deaerator system control two functions which are steam pressure and water level of deaerator storage tank before transfer to the boiler feed water pump. This project is only use logical of deaerator system based on continuous variable. The logical behavior of deaerator system is described by using the flowchart for the purpose of transformation to logical behavior. The transformation to logical behavior is described by using ladder diagram (LD) format of Programmable Logic Controller (PLC) in order to check whether the controller safe or not. Next, the designed controller, mathematical model and other temporal properties are transformed to Boolean equations before performing the verifications. The Computational Tree Logic (CTL) is used in this project as a type of temporal logic for specification and verification purposes. For the FALSE verification result, the detail process of counterexample will not be explained in this project. The verification of the behavior for deaerator system is carried out through SMV model checker software based on Boolean model.

CHAPTER 2

LITERATURE REVIEW

This chapter consists of three parts which are theory, research work and summary of literature review that are related to this project. The theory and information obtained from the published paper is very useful and as a guide to finish this project.

2.1 Theory

2.1.1 Basic Process of Deaerator System

In power plant industry, the steam and water system is in closed loop which means the water leaving the condenser will be fed back to the feed pumps and flow to the boiler [1]. The process will repeat continuously in the power plant system. Figure 2.1 shows the flowchart of main process in the power plant system.

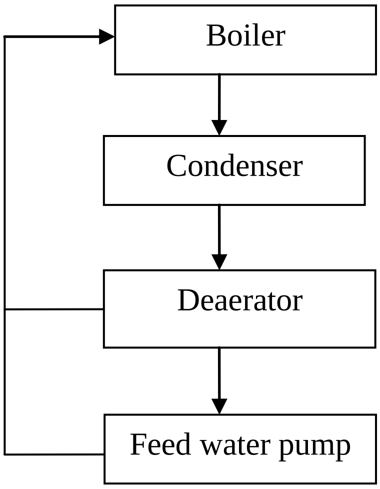


Figure 2.1 : The main process in power plant system

Based on Figure 2.1, the deaerator is one of important system in power plant. The water leaving from condenser cannot flow directly to the feed water pump without pass through the deaerator first. It is because the water from the condenser is cold and contained entrained air that must to be removed before it is transferred to the feed water pump and the boiler. The entrained of air contain dissolved gases which are Oxygen and Carbon dioxide. These two dissolved gas will make corrosion in the boiler, condensate line, steam lines and heat transfer equipments. The basic process of removing the dissolve gases in deaerator has two stages of operations. The first stage is in the Spraymaster, the water leaves the condenser and enter the top of deaerator through the self adjusting spray nozzles into a steam-filled primary heating and vent concentration section. At this stage, the temperature increase at 2 or 3°F of the steam temperature and most of dissolved gases are released at this point.[2] The second stage is in the Boilermate, the steam from the exhaust steam sources flow to the top of deaerator tank. It then flows at the bottom of the deaerator storage tank and flow upward through exchange packing. At the exchange packing, the hot water flow downward and meets the hot steam, deaeration occur and the heating water falls down to the deaerator storage tank and flow to the boiler feed pump.

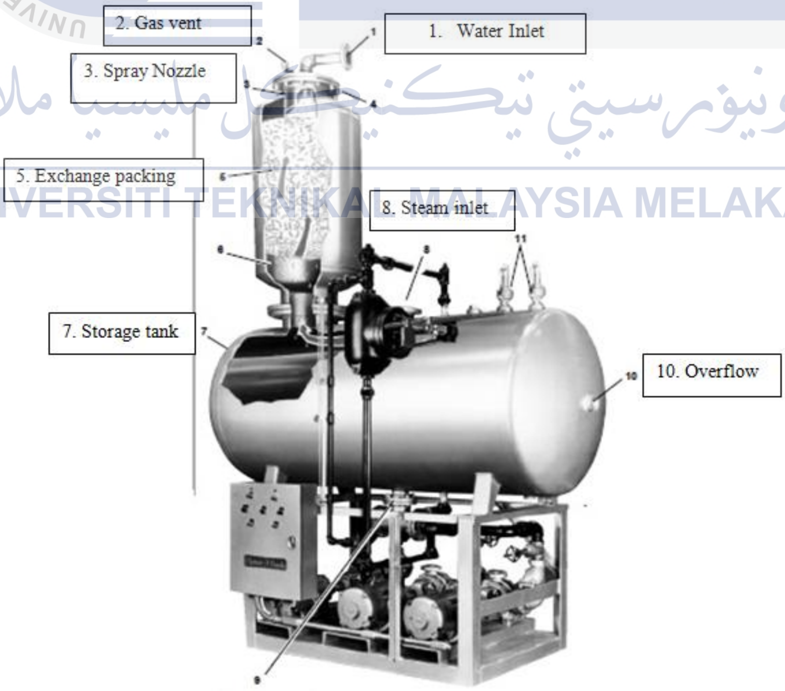


Figure 2.2 The Boilermate deaerator

2.1.2 Formal Method

The formal is frequently supported by tools, it uses mathematically rigorous semantics so that the analysis tool can give high quality of the design and software. The formal method made the interest to the researchers and engineer. It is because it can detect error early and hence reduce the cost of it use. Beside the name of formal method, it also called as mathematically method in determine the specification of the system designed which then can described the correctness with the actual system. The formal method or mathematical modeling is importance because it not possible the software system have free of error, sometimes the data that used in the software possible to become failed, hence the concept of formal method will be check and led to the correct result [3]. The formal method can be proven by using model checking and the theorem proving. The model checking technique can be check through the Symbolic Model Verify (SMV) model checker software.

2.1.3 Model Checking

Model checking is an automated verification technique which focused on determines the temporal logic and sees whether specification is same with the system designed or not. The tools of model checking can be check through SMV model checker software. Besides, by model checking, it also can detect the errors use in software system with the actual system and can give counterexample execute of model system on why the system is violated.

Three main creating the model checking technique are modeling, specification and verification. Modeling means convert the design into a suitable formal form which is compatible the model checking software that usually use formalism. Next, specification means ensure the properties of the design system are stated by using logic expression. The last one is verifying means determination the model obtained in first state is true with the next state or not.

Model checking has several of advantages. The first one is this verification technique has no proofs means it does need to construct the correctness proof. When enter model description or diagram into the software, the checking will automatically run. Second is it has counterexample means if the specification is not correct, the model checking software will show why the specification is not same with the system. Next is model checking can have many temporal logic properties for concurrent systems. A concurrent system means several behaviors that occur simultaneously and connected to each other [4].

2.1.4 Temporal Logic

The first step in formal verification is the representation of formal specification of the design consisting of a description of the desired behavior. Temporal Logic is a logic expression the ordering events in time without introducing time explicitly [5]. It is also one of the formal verification techniques that can be used to verify the correctness of a finite state concurrent system. There are two types of temporal logic in order to express the properties for verification which are linear temporal logic (LTL) and Computational Tree Logic (CTL). In LTL, time s treated as if each moment has a unique possible future. The Boolean connectives as well as temporal connectives usually uses X (“next”), G (“always”), F (“eventually”), and U (“until”).

CTL is also called as branching temporal logic. It is refer to the fact that at each moment there is several different possible futures. This type of temporal logic is described the tree of states rather than sequences. The tree is possibly representing all possible computations. CTL explicitly introduces path quantifiers which are “all path” and “Exist a path”. “All path” represent “A” notation which mean the temporal logic is true in all paths starting in the current states while “Exist a path” represent “E” notation which mean the temporal formula is true in some path starting in the current state [6]. These two notation are combined with other alternative notation are used for temporal operators. The first one is “AF *p*” describes that for all the path starting from a state, eventually in the future, condition *p* must be hold. Second, “EF *p*” describes that there exists some path that eventually in the future satisfies *p*. Third, “AG *p*” describes that condition *p* is always or globally, true in all states of all the possible paths. Fourth, “EG *p*” describes that there is some path along which condition *p* is continuously true.

2.1.5 SMV Model Checker

SMV is symbolic model checking tools which is one of the software that can be verifying the temporal logic properties of finite systems [7]. SMV verifies every possible behavior of the system's behavior of the system satisfies the specification. A specification for SMV is a collection of properties. The properties can be created or can be specified in a notation using temporal logic.

2.2 Research Work

Through out the research that had done in order to understand this project, there are several terms meets from the previous paper. The first term is the basic about the formal method. Formal method is an effective techniques for automation software verifications which is increasingly been recognized. Software verification of formal method is an integral part of the software development which is to ensure the software system is satisfies with all the requirements model system. Researchers have been developed many different approach, but technique of mathematical modeling of this formal method is focused since the formal method can finding the errors of the model system [8]. The application of formal method by the other research is presents the formal method of Partition-and-Recur (PAR) method. It is used to design and prove the algorithmic programs. It also is an effective formal method on solving the Combinatory problems by using PAR method. This formal method not only simplifies the process of algorithms and correctness the testing but also improves the atomization, standardization and corrects the algorithm by changing many creative labors to mechanized labors. It used the C++ language [9]. Next is, formal method in requirement engineering and its application. The formal method divide into two forms which are in model checking and the theorem proving. The tool of model checking is by using NuSMV model checker and SPIN model checker. Model checking is a verification of finite state system. Finite state mean when all its variable range over discrete domain hence, number of possible state is finite. Next is by theorem proving. It is another approach to proving a specification is correct. The tool of theorem that usually used is Z "Zed" [10].

The second term covered in this research work is the automated verification. The automated verification is by verifying the properties of model system. It has several steps by doing verification in this paper. The first one is state the initial conditions that must be followed by input of model system. Next are the verifiable of the properties needed to be described as formula of MCDL in VMTS and finally the proofs of the properties of the model system detailed [11].

The third term is model checking technique. The paper presents the application of model checking technique in two system design of two nuclear power plant. Model checking technique is one of the methods for verifying whether the model of system is fulfills the specification by determine all the possible behaviors that have in the system. By the given a model and the specification, the model checker which is NuSMV can determine whether the software system used is behave with the model system or not. If the system is violates, the model checker do the counterexample which it will execute the model system and determine why it's false or not behave as the model system. The model used in this paper use the flow chart to describe the emergency cooling system. It used the state variable with the different possible states. "high", "low" and "medium". The model of the system use the principle of when no water flow into reactor container, the water will remain same or decrease and if water is flowing in, the water level will remain same [12]. The example of model checking on the other research.The paper discussed the verifying the design of satellite software control system which is called attitude and orbit control system (AOCS) by using model checking technique. The system behavior is to maintain the attitude of the satellite and for performing fault at detection, isolation and recovery decision of satellite. The verifying is use by symbolic model checker NuSMV 2. The diagram is transform to temporal logic properties by using BDD-based LTL model checking to prove the model system [13].

Besides, the other terms on the research work is Boolean expression. The use of Boolean Expression Manipulator from the research by using BDDs**.**The paper presents the LSI design system using Binary Decision Diagram (BDDs) which is use the Boolean expression. It uses Arithmetic Boolean Expression Manipulator (BEM-II) which is also on BDD technique. It calculates the Boolean expression that contains mathematic expression which is subtraction, addition, multiplication and comparison and then transform to the other various format [14].

The last terms from the research work is mathematical model of deaerator model. From the previous work, the paper introduced a mathematical model of deaerator based on the physical law of mass and energy conservation. It also described the basic operation of deaerator model. It was used the differential equation in order to analyze the parameter variation over the time. The dynamic simulation model that used in the paper shows the dynamic behavior of the model. The method make the behavior of the deaerator is easier to study and different malfunction can be avoided [15]

2.3 Summary of Literature Review

Based on the knowledge gained from the literature, a formal method is widely used by the engineer and the research since a decade ago. Formal method is use in software and hardware system which is to specify the system and to ensure that the specification is meeting with the actual system or not. Formal method is an effective automated verification in a software system and use the mathematical model approached.

The formal method can be done by two concepts which are through the model checking technique and the theorem proving technique. Model checking is used to see the specification that created using temporal properties is same with the model or not. There have one of the research work that use model checking technique which is the emergency cooling system of nuclear power plant. That project uses principle that need to satisfy for a purpose of safety function.

CHAPTER 3

METHODOLOGY

This chapter is about the explanation on how the project is done. The details consist of the logical behavior of the deaerator system, mathematical model, and control logic for deaerator system.. This project will use SMV model checker software in order to verify the deaerator control system. Figure 3.1 show the procedure of this project.

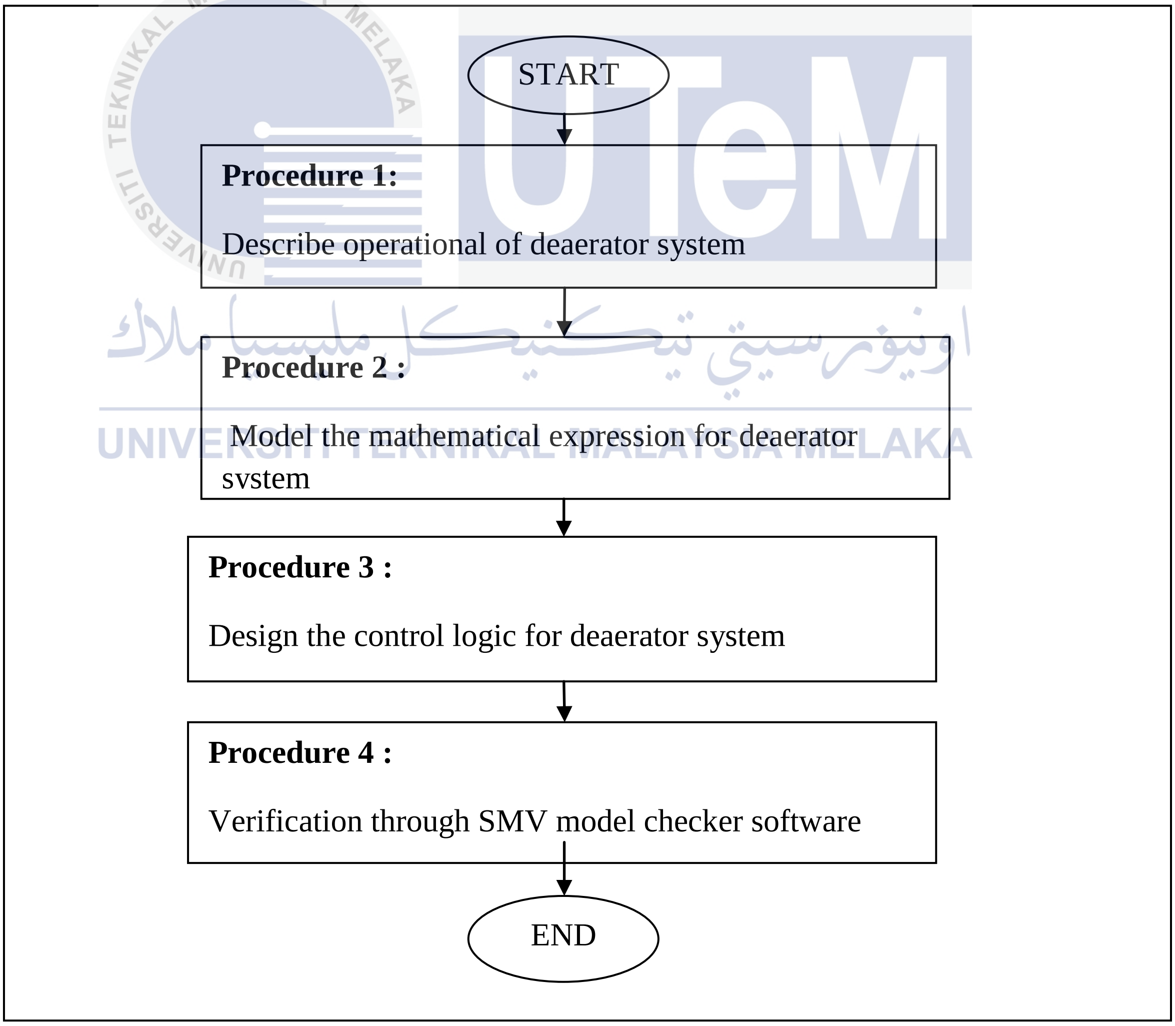

START
Procedure 1:
Describe operational of deaerator system
Procedure 2 :
Model the mathematical expression for deaerator svstem
Procedure 3 :
Design the control logic for deaerator system
Procedure 4 :
Verification through SMV model checker software
END

Figure 3.1 : Flowchart of project procedure

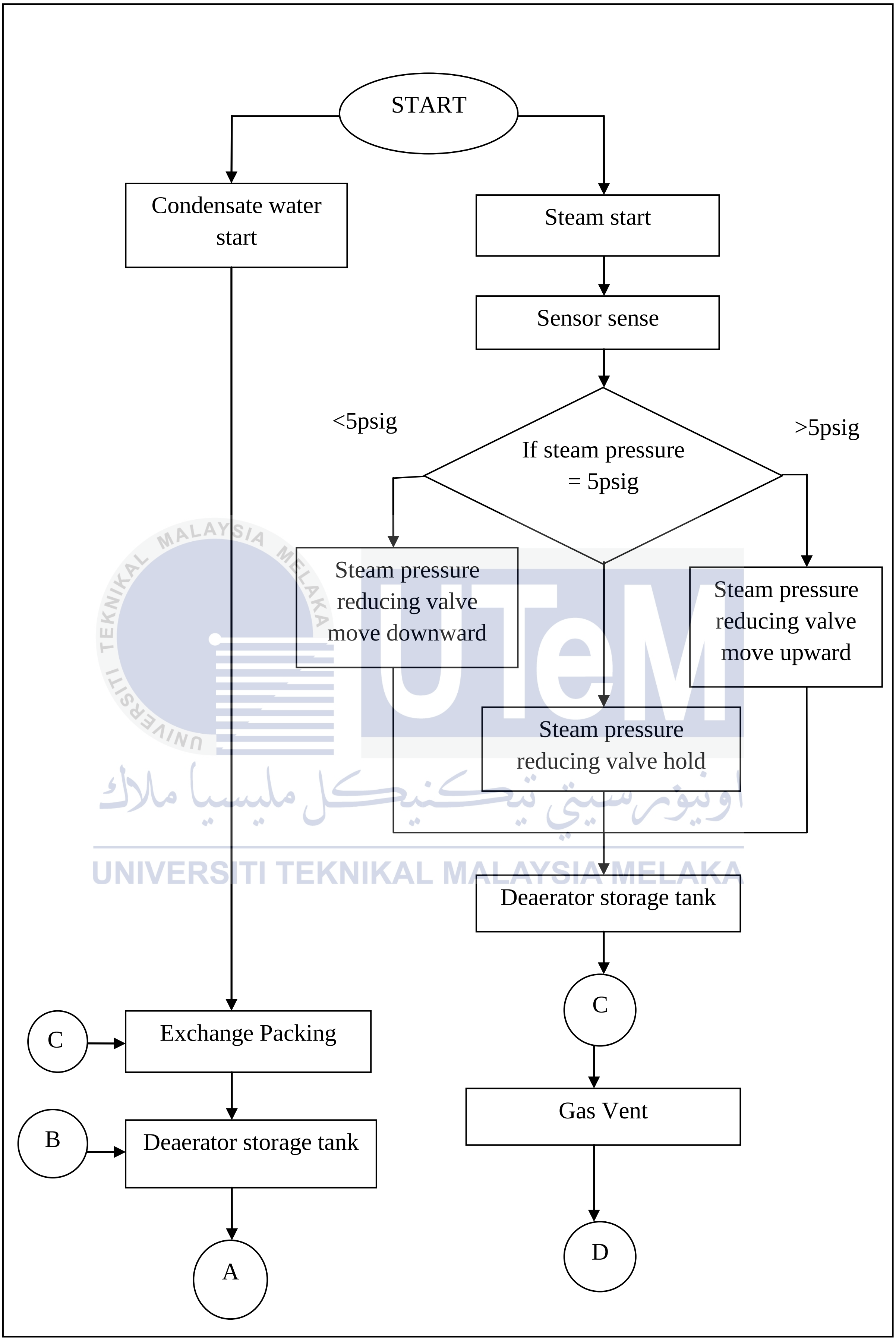
3.1 Procedure 1: Describe Operational Of Deaerator System

In this task, the behavior of deaerator system is described by using the flowchart for the purpose of transformation to logical behavior. The flow of behavior for deaerator system is described from the operation of Boilermate deaerator design [16].

From the Figure 3.2, there are two control functions required by the deaerator which are water level of deaerator storage tank and steam pressure. The condensate water enters the exchange packing and it then flows down to the deaerator storage tank. At the same time, steam is enters the system at the top of the deaerator storage tank. It then enters at the bottom of the deaerator and flows upward to the exchange packing. Steam and water sweeps and completing the deaeration process. After being deaerated, the fully heated water falls to the deaerator storage tank for the flow to the boiler feed pumps.

Both steam and water have their own control system. For the water level, when the condensate water enters the deaerator storage tank, the level control will sense the water. There are three conditions of water level which are less than 80 percent, equal to 80 percent and more than 80 percent. If the water level of deaerator storage tank is less than 80 percent, the make-up water valve will open and the make-up water will enter to the deaerator storage tank. If the water level of deaerator storage tank is more than 80 percent, the overflow control valve will open and vent the water to the overflow drainer. Lastly, if the water level of deaerator storage tank is equal to 80 percent, the boiler feed water pump will start and pass through the boiler.

For the steam pressure, steam enters the system at the top of the deaerator storage tank through the steam pressure reducing valve. As the steam pressure differs from the set point, the steam pressure reducing valve will modulate to give the constant pressure. Firstly, if the steam pressure is less than 5 psig, the steam pressure reducing valve will move downward. Second, if the steam pressure is more than 5 psig, the steam pressure reducing valve will move upward. Third, the steam pressure reducing valve will be maintained or on hold if the steam pressure is equal to 5 psig and then it will be passed through the deaerator storage tank and upward to the exchange packing. Figure 3.2 shows the operation of deaerator system.


START
Condensate water start
Steam start
Sensor sense
<5psig
If steam pressure = 5psig
>5psig
Steam pressure reducing valve move downward
Steam pressure reducing valve move upward
Steam pressure reducing valve hold
Deaerator storage tank
C
Exchange Packing
B
Deaerator storage tank
A
C
Gas Vent
D

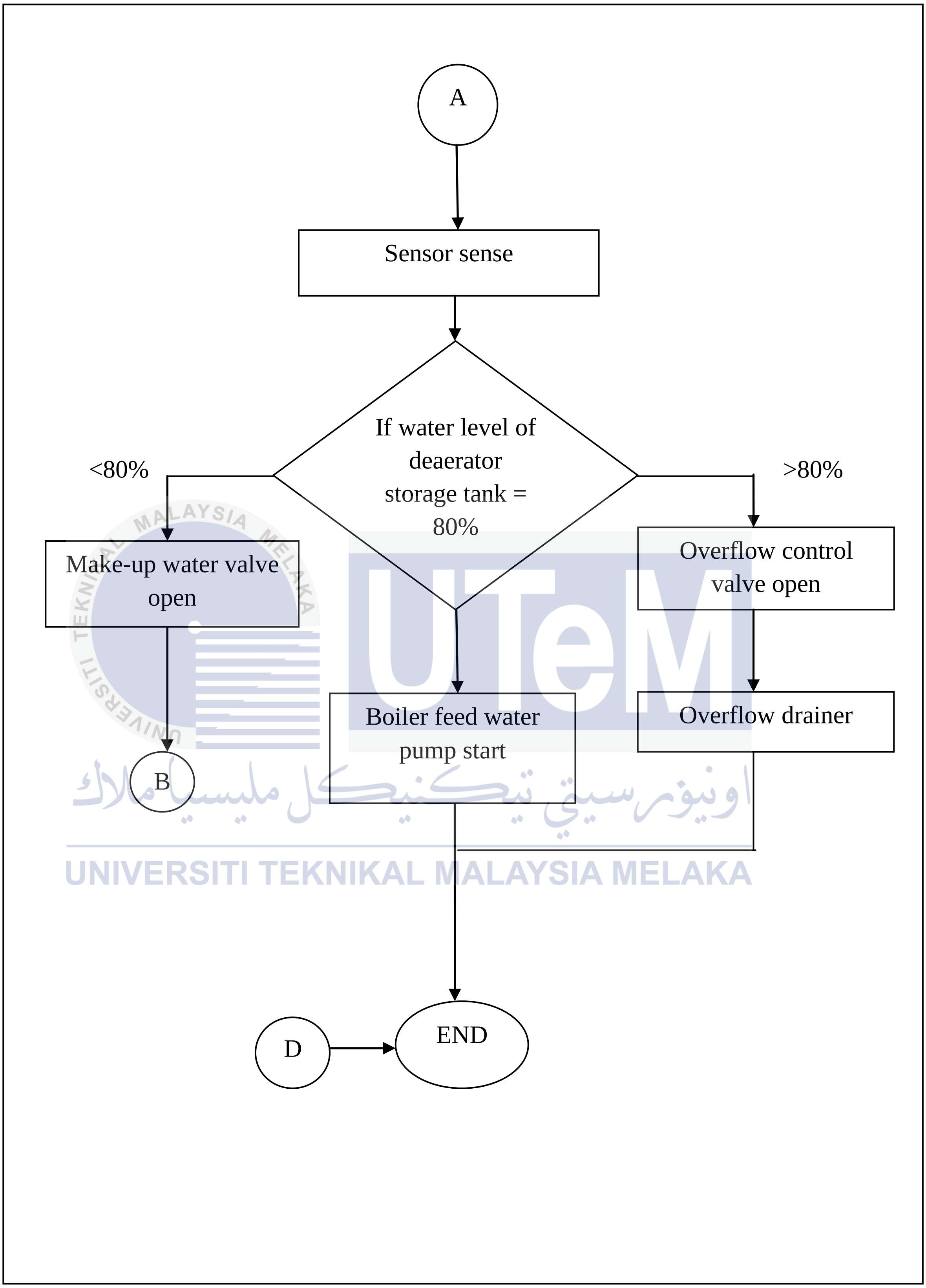

A
Sensor sense
If water level of deaerator storage tank = 80%
<80%
>80%
Make-up water valve open
Overflow control valve open
Boiler feed water pump start
Overflow drainer
B
D
END

Figure 3.2: Operation of Deaerator system

Table 3.1 shows the several of specification that indicates the condition to occur during the operation of deaerator system.

Table 3.1 : The specification of deaerator used

Data	**Specification**
Type of deaerator	Boilermate deaerator
Heating steam pressure	5 psig
Dearator storage tank capacity	230 gallon

3.2 Procedure 2 : Model the Mathematical Model for Deaerator System

In this project, all the behavior involved need to be transformed to the Boolean expression before proceed to the formal verification using SMV model checker. The mathematical expression is created based on the Grobner Bases. Grobner Bases is a series of polynomial equation that can be solved by adding, subtracting and multiplying each other to eliminate variables and generate solution of the system of algebraic equations [17].

A mathematical model is a model that uses mathematical language to describe the behavior of a system [18]. It is widely used in engineering discipline such as in electrical engineering. Mathematical modeling using logical condition using Boolean expression is easy to specify the real problem. It also is a mixing of mathematical and the logical expression.

In this section, it has two steps in order to determine the expression of the mathematical model. [19] **First step** is by creating the definition of states as shown in Table 3.2. Definition of states is the list of variables of all actuator input, *u* and sensor output, *x* used in the model system. **Second step** is by creating the pre-condition and post-condition. It can be summary that pre-condition, x_n is a statement that must indicates before the sensor output system called post-condition. Post-condition,dx_n is a statement that will indicate after the actuator input called. In pre-condition and post-condition have two basic values in the logic which are 1 and 0, 1 mean by "true" and 0 mean by "false". From the Table 3.3 until Table 3.12, in order to get the logic expression in each input variable, the minimization of combinational logic expression will be simplified via Galois field representation before executing the computations as in **third step.**

Figure 3.3 illustrate the system layout of deaerator system where the actuator input and sensor ouput are located in deaerator system.

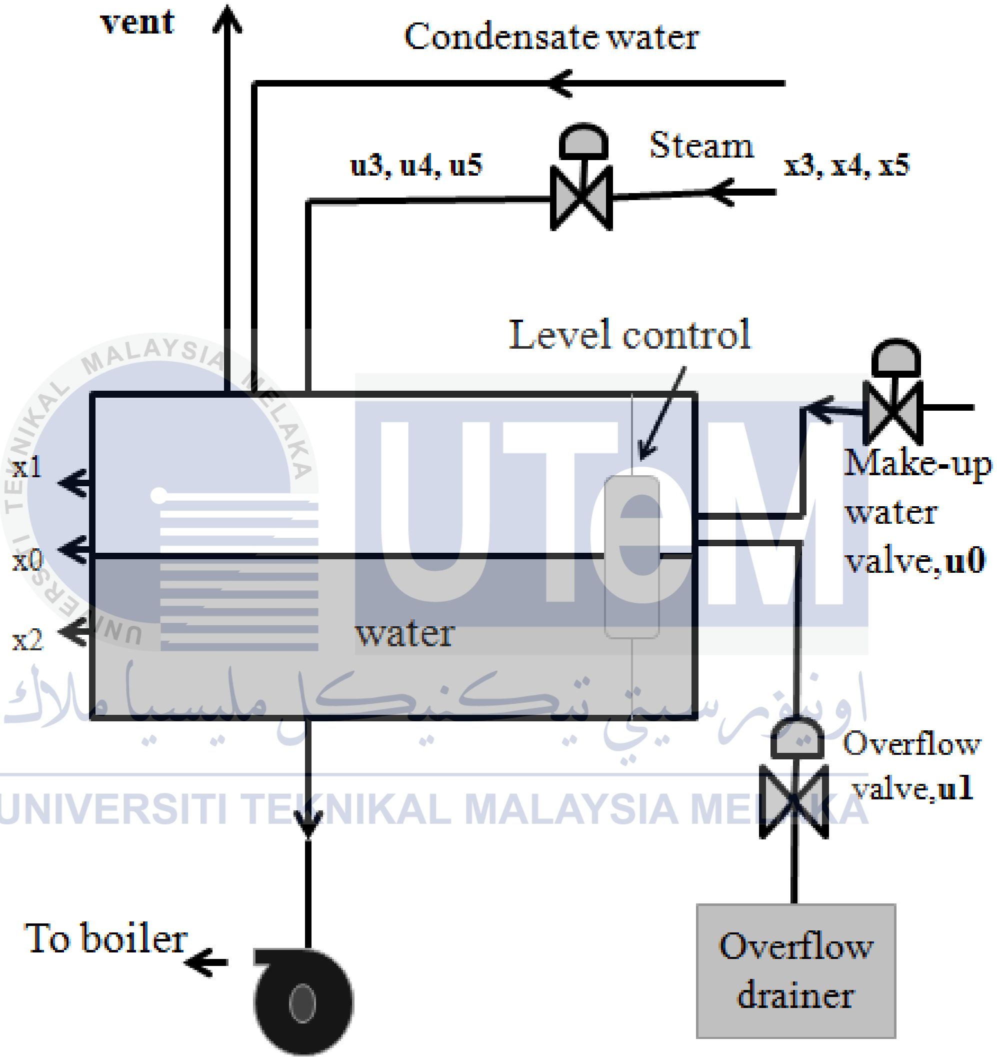


Figure 3.3: System layout of Deaerator

Mathematical models or operational models of the Deaerator system are described in Table 3.2 – Table 3.12.

3.2.1 Step 1 : Definition of States of Deaerator System

Table 3.2 : Definition of states of Deaerator system

States	Definition of states
x_0	Normal water level
x_1	High water level
x_2	Low water level
x_3	Normal steam pressure
x_4	High steam pressure
x_5	Low steam pressure
u_0	Boiler feed water pump
u_1	Overflow water valve
u_2	Make-up water valve
u_3	Steam pressure reducing valve hold
u_4	Steam pressure reducing valve upward
u_5	Steam pressure reducing valve downward

3.2.2 Step 2: Pre-Condition and Post-Condition

Table 3.3, Table 3.4, Table 3.5 and Table 3.6 below show the operational of transition states between water level and steam pressure while Table 3.7 show the combination of all operational model for deaerator system.

Table 3.3: Operational model of High_Normal Water Level and Normal Steam Pressure

Pre-condition				Input				Post-condition			
x0	x1	x2	x3	u0	u1	u2	u3	dx0	dx1	dx2	dx3
0	1	0	1	0	1	0	1	0	0	0	1
0	0	0	1	0	1	0	1	1	0	0	1

Table 3.4: Operational model of Normal_Low Water Level and Normal Steam Pressure

Pre-condition				Input				Post-condition			
x0	**x1**	**x2**	**x3**	**u0**	**u1**	**u2**	**u3**	**dx0**	**dx1**	**dx2**	**dx3**
1	0	0	1	1	0	0	1	0	0	0	1
0	0	0	1	1	0	0	1	0	0	1	1

Table 3.5 : Operational model of Low_Normal Water Level and Normal Steam Pressure

Pre-condition				Input				Post-condition			
x0	**x1**	**x2**	**x3**	**u0**	**u1**	**u2**	**u3**	**dx0**	**dx1**	**dx2**	**dx3**
0	0	1	1	0	0	1	1	0	0	0	1
0	0	0	1	0	0	1	1	1	0	0	1

Table 3.6 : Operational model of Normal_High Water Level and Normal Steam Pressure

Pre-condition				Input				Post-condition			
x0	**x1**	**x2**	**x3**	**u0**	**u1**	**u2**	**u3**	**dx0**	**dx1**	**dx2**	**dx3**
1	0	0	1	1	0	0	1	0	0	0	1
0	0	0	1	0	0	1	1	0	1	0	1

Table 3.7 : Operational model for combination of water level and normal steam pressure

Pre-condition				Input				Post-condition			
x0	**x1**	**x2**	**x3**	**u0**	**u1**	**u2**	**u3**	**dx0**	**dx1**	**dx2**	**dx3**
0	1	0	1	0	1	0	1	0	0	0	1
0	0	0	1	0	1	0	1	1	0	0	1
1	0	0	1	1	0	0	1	0	0	0	1
0	0	0	1	1	0	0	1	0	0	1	1
0	0	1	1	0	0	1	1	0	0	0	1
0	0	0	1	0	0	1	1	1	0	0	1
1	0	0	1	1	0	0	1	0	0	0	1
0	0	0	1	0	0	1	1	0	1	0	1

Table 3.8, Table 3.9, Table 3.10 and Table 3.11 below show the cases of the operational model for Steam Pressure Reducing Valve. While Table 3.12 show the overall operational model after combine each of the cases.

Table 3.8: Operational model for High_Normal Steam Pressure

Pre-condition			Input			Post-condition		
x3	**x4**	**x5**	**u3**	**u4**	**u5**	**dx3**	**dx4**	**dx5**
0	1	0	0	1	0	0	0	0
0	0	0	0	1	0	1	0	0

Table 3.9: Operational model for Normal_Low Steam Pressure

Pre-condition			Input			Post-condition		
x3	**x4**	**x5**	**u3**	**u4**	**u5**	**dx3**	**dx4**	**dx5**
1	0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0	1

Table 3.10 : Operational model for Normal_High Steam Pressure

Pre-condition			Input			Post-condition		
x3	**x4**	**x5**	**u3**	**u4**	**u5**	**dx3**	**dx4**	**dx5**
1	0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	1	0

Table 3.11: Operational model for Low_Normal Steam Pressure

Pre-condition			Input			Post-condition		
x3	**x4**	**x5**	**u3**	**u4**	**u5**	**dx3**	**dx4**	**dx5**
0	0	1	0	0	1	0	0	0
0	0	0	0	0	1	1	0	0

Table 3.12 : Overall Operational model for Steam Pressure Reducing Valve

Pre-condition			Input			Post-condition		
x3	**x4**	**x5**	**u3**	**u4**	**u5**	**dx3**	**dx4**	**dx5**
0	1	0	0	1	0	0	0	0
0	0	0	0	1	0	1	0	0
1	0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0	1
0	0	1	0	0	1	0	0	0
0	0	0	0	0	1	1	0	0
1	0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	1	0

3.2.3 Step 3 : Mathematical Expression in Boolean Form

Table 3.7 and Table 3.12 are summarization of the operational model for Deaerator system. Based on the model obtained, the Boolean expressions can be determined as in (3.1) and (3.2).

i) The Boolean expression for Table 3.7 is shown below:

$$\begin{aligned} dx0 = {} & \overline{x0}\ \overline{x1}\ \overline{x2}\,x3\,\overline{u0}\,u1\,\overline{u2}\,u3 \lor \overline{x0}\,\overline{x1}\overline{x2}\,x3\overline{u0}\,\overline{u1}\,u2\,u3 \\ & \lor \overline{x1}\overline{x2}\,x3\overline{u1}u3\overline{x0}\overline{u0}\,u2 \end{aligned}$$

$$\begin{aligned} dx1 = {} & \overline{x0}\,\overline{x1}x3\overline{u0}\overline{u1}u2u3\overline{x2} \lor x0\overline{x1}\overline{x2}\,\overline{x3}u0\overline{u1}\,\overline{u2}\,u3 \\ & \lor \overline{x0}\,\overline{x1}\,\overline{x2}x3\overline{u0}\,\overline{u1}u2u3 \end{aligned}$$

$$dx2 = \overline{x0}\,\overline{x1}x3\overline{u0}\,\overline{u1}u2u3\overline{u2} \lor \overline{x0}\,\overline{x1}\overline{x2}x3\,u0\overline{u1}\overline{u2}u3$$

$$\begin{aligned} dx3 = {} & \overline{x0}x1\overline{x2}x3\overline{u0}u1\overline{u2}u3 \lor \overline{x0}\overline{x1}\overline{x2}x3\overline{u0}u1\overline{u2}u3 \lor x0\overline{x1}\overline{x2}x3u0\overline{u1}\,\overline{u2}u3 \lor \\ & \overline{x0}\,\overline{x1}\,\overline{x2}x3u0\overline{u1}\,\overline{u2}u3 \lor \overline{x0}\overline{x1}\,x2\,x3\overline{u0}\,\overline{u1}\,u2u3 \lor \\ & \overline{x0}\overline{x1}\overline{x2}x3\overline{u0}\,\overline{u1}u2u3 \lor x0\overline{x1}\overline{x2}x3u0\overline{u1}\,\overline{u2}u3 \lor \\ & \overline{x0}\,\overline{x1}\,\overline{x2}x3\overline{u0}\overline{u1}u2\,u3 \qquad (3.1) \end{aligned}$$

ii) The Boolean expression for Table 3.12 is shown below:

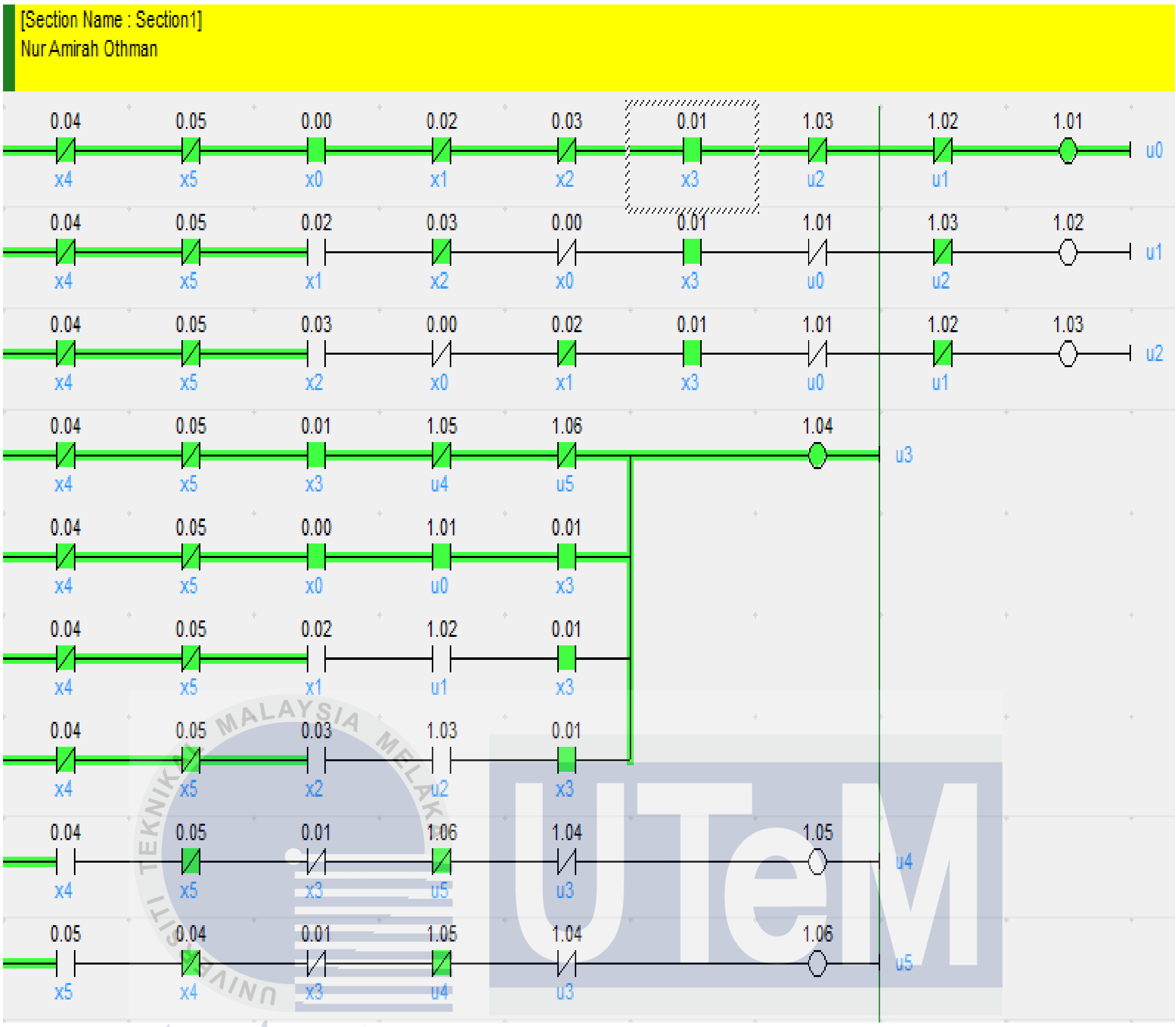
$$\begin{aligned} dx3 &= \overline{x3}\overline{x4}\overline{x5}\overline{u3}u4\overline{u5} \lor \overline{x3}\overline{x4}\overline{x5}\overline{u3}\,\overline{u4}\,u5 \\ dx4 &= \overline{x3}\overline{x4}\overline{x5}u3\overline{u4}\overline{u5} \lor \overline{x3}\overline{x5}\overline{u3}u4\overline{u5}\overline{x4} \lor \\ & \quad \overline{x4}\overline{x5}u3\overline{u4}\overline{u5}\overline{x3} \lor \overline{x3}\,\overline{x4}\,\overline{u3}\,\overline{u4}u5\overline{x5} \lor x3\overline{x5}\overline{x4}u3\overline{u4}\overline{u5} \\ dx5 &= \overline{x3}\,\overline{x4}\overline{x5}u3\overline{u4}\overline{u5} \lor \overline{x3}\overline{x5}\overline{u3}u4\overline{u5}\overline{x4} \lor \\ & \quad x3\overline{x4}\overline{x5}u3\overline{u4}\overline{u5} \lor \overline{x4}\overline{x5}u3\overline{u4}\overline{u5}\overline{x3} \qquad (3.2) \end{aligned}$$

3.3 Procedure 3 : Design the Control Logic for Deaerator System

The transformation of the logical behavior for deaerator will be design by using ladder diagram (LD) format. The logic controller that was created is defined as Safety of Controller. Figure 3.4 – Figure 3.9 show the operational control logic for Deaerator control system. The control logic is designed based consideration of the safety condition. It was tested before transform to the Boolean expression for the verification process. Table below show the variables used for the control logic :

Table 3.13 : Variable of the Deaerator Control Logic

Variable	Description
x_0	Normal water level sensor
x_1	High water level sensor
x_2	Low water level sensor
x_3	Normal steam pressure sensor
x_4	High steam pressure sensor
x_5	Low steam pressure sensor
u_0	Boiler feed water pump
u_1	Overflow water valve
u_2	Make-up water valve
u_3	Steam pressure reducing valve hold
u_4	Steam pressure reducing valve move upward
u_5	Steam pressure reducing valve move downward

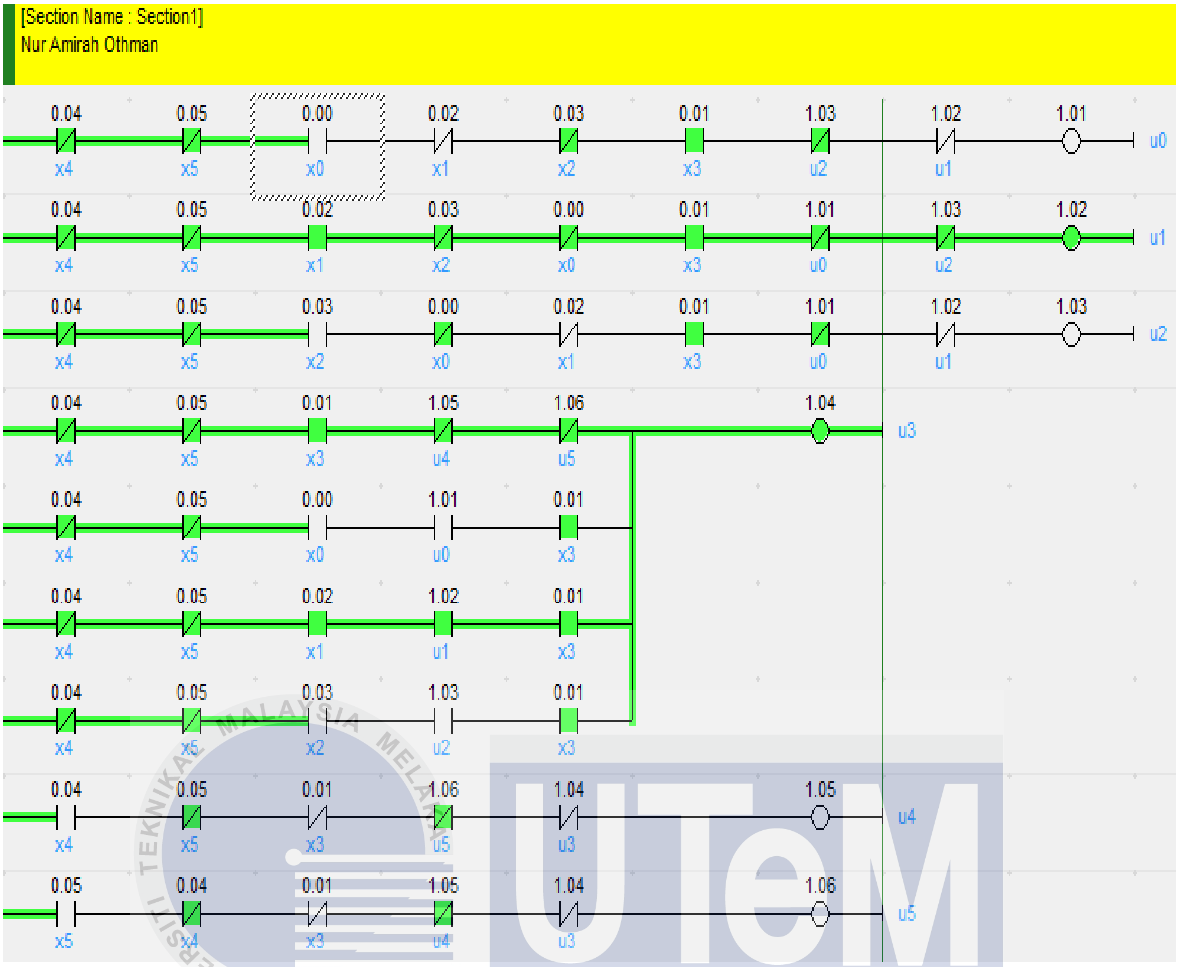


[Section Name : Section1]
Nur Amirah Othman

0.04 x4 0.05 x5 0.00 x0 0.02 x1 0.03 x2 0.01 x3 1.03 u2 1.02 u1 1.01 u0
0.04 x4 0.05 x5 0.02 x1 0.03 x2 0.00 x0 0.01 x3 1.01 u0 1.03 u2 1.02 u1
0.04 x4 0.05 x5 0.03 x2 0.00 x0 0.02 x1 0.01 x3 1.01 u0 1.02 u1 1.03 u2
0.04 x4 0.05 x5 0.01 x3 1.05 u4 1.06 u5 1.04 u3
0.04 x4 0.05 x5 0.00 x0 1.01 u0 0.01 x3
0.04 x4 0.05 x5 0.02 x1 1.02 u1 0.01 x3
0.04 x4 0.05 x5 0.03 x2 1.03 u2 0.01 x3
0.04 x4 0.05 x5 0.01 x3 1.06 u5 1.04 u3 1.05 u4
0.05 x5 0.04 x4 0.01 x3 1.05 u4 1.04 u3 1.06 u5

Figure 3.4 : Input of normal water level, x0 and normal steam pressure, x3

Based on Figure 3.4, when the water level in the Deaerator storage tank and the steam pressure in exchange packing is normal, the normal water level sensor,u0 and normal steam pressure, x3 will detect Hence, the boiler feed water pump,u0 will on and the steam pressure reducing valve, u3 will hold. Thus, the water is pumped to the boiler. The steam pressure will always normal as the water level is normal.



[Section Name : Section1]
Nur Amirah Othman

0.04 x4 0.05 x5 0.00 x0 0.02 x1 0.03 x2 0.01 x3 1.03 u2 1.02 u1 1.01 u0
0.04 x4 0.05 x5 0.02 x1 0.03 x2 0.00 x0 0.01 x3 1.01 u0 1.03 u2 1.02 u1
0.04 x4 0.05 x5 0.03 x2 0.00 x0 0.02 x1 0.01 x3 1.01 u0 1.02 u1 1.03 u2
0.04 x4 0.05 x5 0.01 x3 1.05 u4 1.06 u5 1.04 u3
0.04 x4 0.05 x5 0.00 x0 1.01 u0 0.01 x3
0.04 x4 0.05 x5 0.02 x1 1.02 u1 0.01 x3
0.04 x4 0.05 x5 0.03 x2 1.03 u2 0.01 x3
0.04 x4 0.05 x5 0.01 x3 1.06 u5 1.04 u3 1.05 u4
0.05 x5 0.04 x4 0.01 x3 1.05 u4 1.04 u3 1.06 u5

Figure 3.5 : Input of high water level, x1 and normal steam pressure, x3

Based on Figure 3.5, when the water level in the Deaerator storage tank is high and the steam pressure in exchange packing is normal, the high water level sensor,x1 and normal steam pressure, x3 will detect Hence,the overflow water valve,u1 will on and the steam pressure reducing valve, u3 will hold. Thus, the water is vented to the drainer. The steam pressure will always normal as the water level is high.

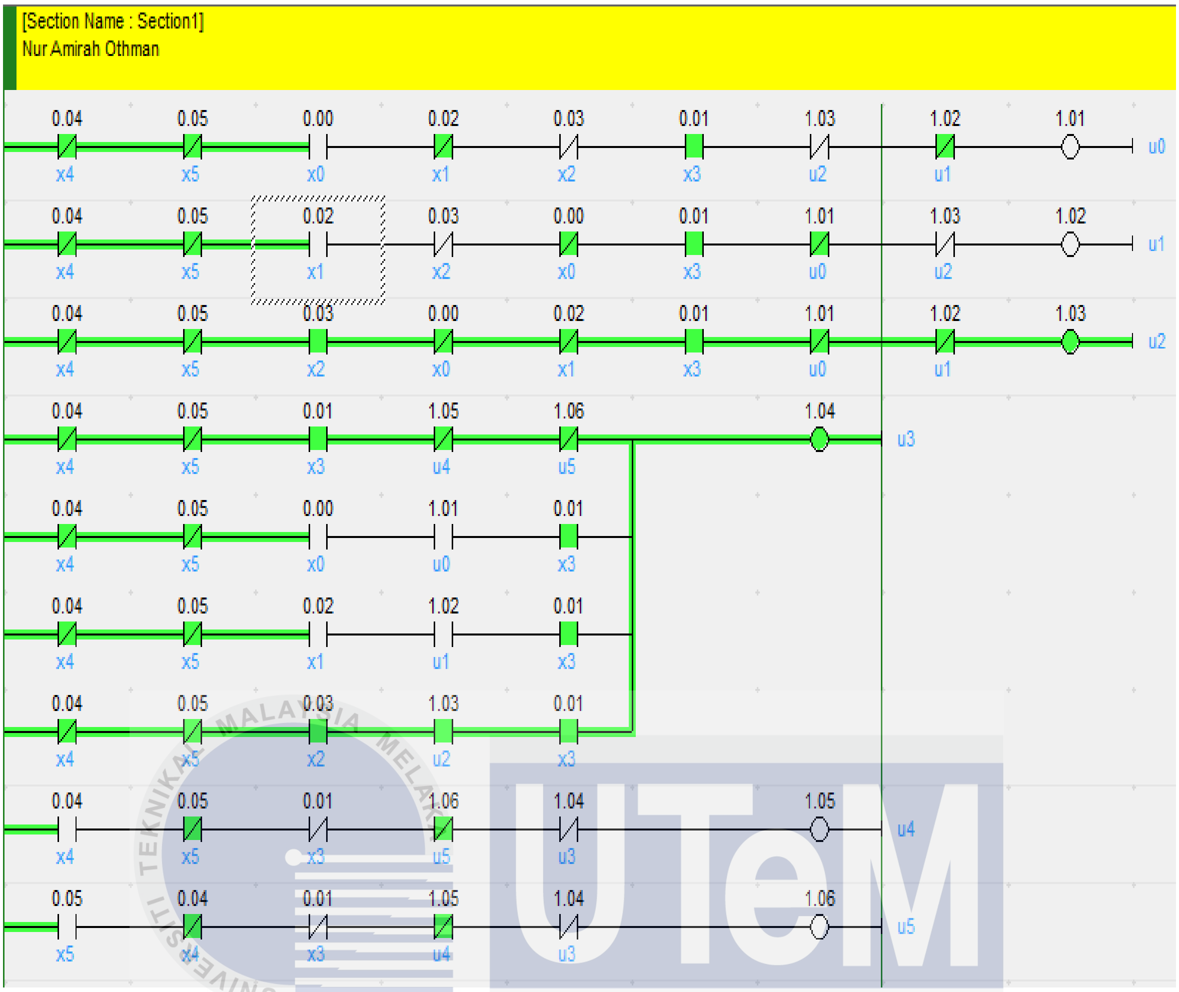
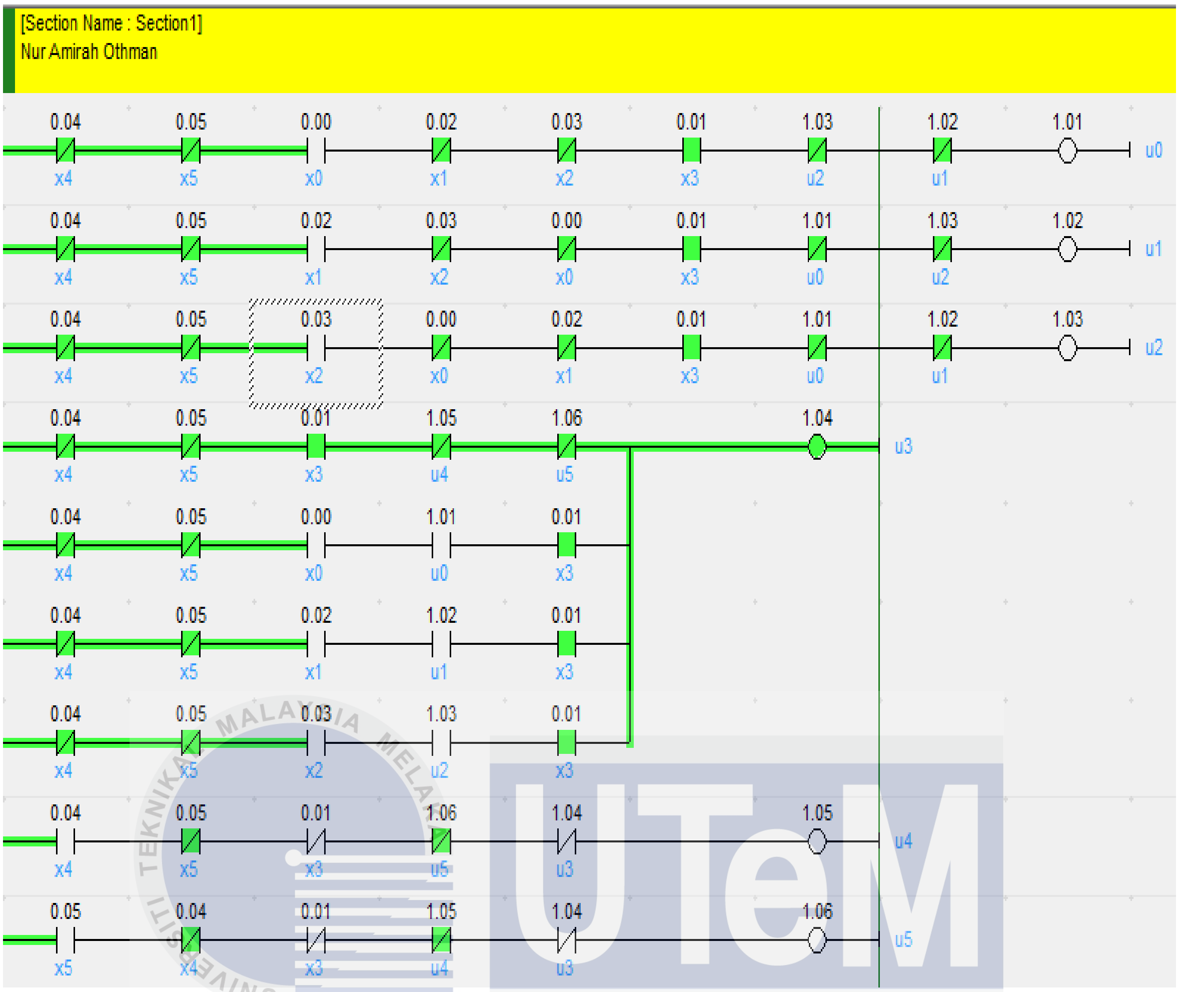

[Section Name : Section1]
Nur Amirah Othman
0.04 x4 0.05 x5 0.00 x0 0.02 x1 0.03 x2 0.01 x3 1.03 u2 1.02 u1 1.01 u0
0.04 x4 0.05 x5 0.02 x1 0.03 x2 0.00 x0 0.01 x3 1.01 u0 1.03 u2 1.02 u1
0.04 x4 0.05 x5 0.03 x2 0.00 x0 0.02 x1 0.01 x3 1.01 u0 1.02 u1 1.03 u2
0.04 x4 0.05 x5 0.01 x3 1.05 u4 1.06 u5 1.04 u3
0.04 x4 0.05 x5 0.00 x0 1.01 u0 0.01 x3
0.04 x4 0.05 x5 0.02 x1 1.02 u1 0.01 x3
0.04 x4 0.05 x5 0.03 x2 1.03 u2 0.01 x3
0.04 x4 0.05 x5 0.01 x3 1.06 u5 1.04 u3 1.05 u4
0.05 x5 0.04 x4 0.01 x3 1.05 u4 1.04 u3 1.06 u5

Figure 3.6 : Input of low water level, x2 and normal steam pressure, x3

Based on Figure 3.6, when the water level in the Deaerator storage tank is low and the steam pressure in exchange packing is normal, the low water level sensor,x2 and normal steam pressure, x3 will detect Hence,the make-up water valve,u2 will on and steam pressure reducing valve, u3 will hold. Thus, the water is pumped to the Deaerator storage tank. The steam pressure will always normal as the water level is low.



[Section Name : Section1]
Nur Amirah Othman

Figure 3.7 : Input of normal steam pressure, x3

Figure 3.7 show the working process of the steam pressure reducing valve, when the steam pressure is normal, normal steam pressure sensor,x3 will detect and thus, the steam pressure reducing valve will hold.

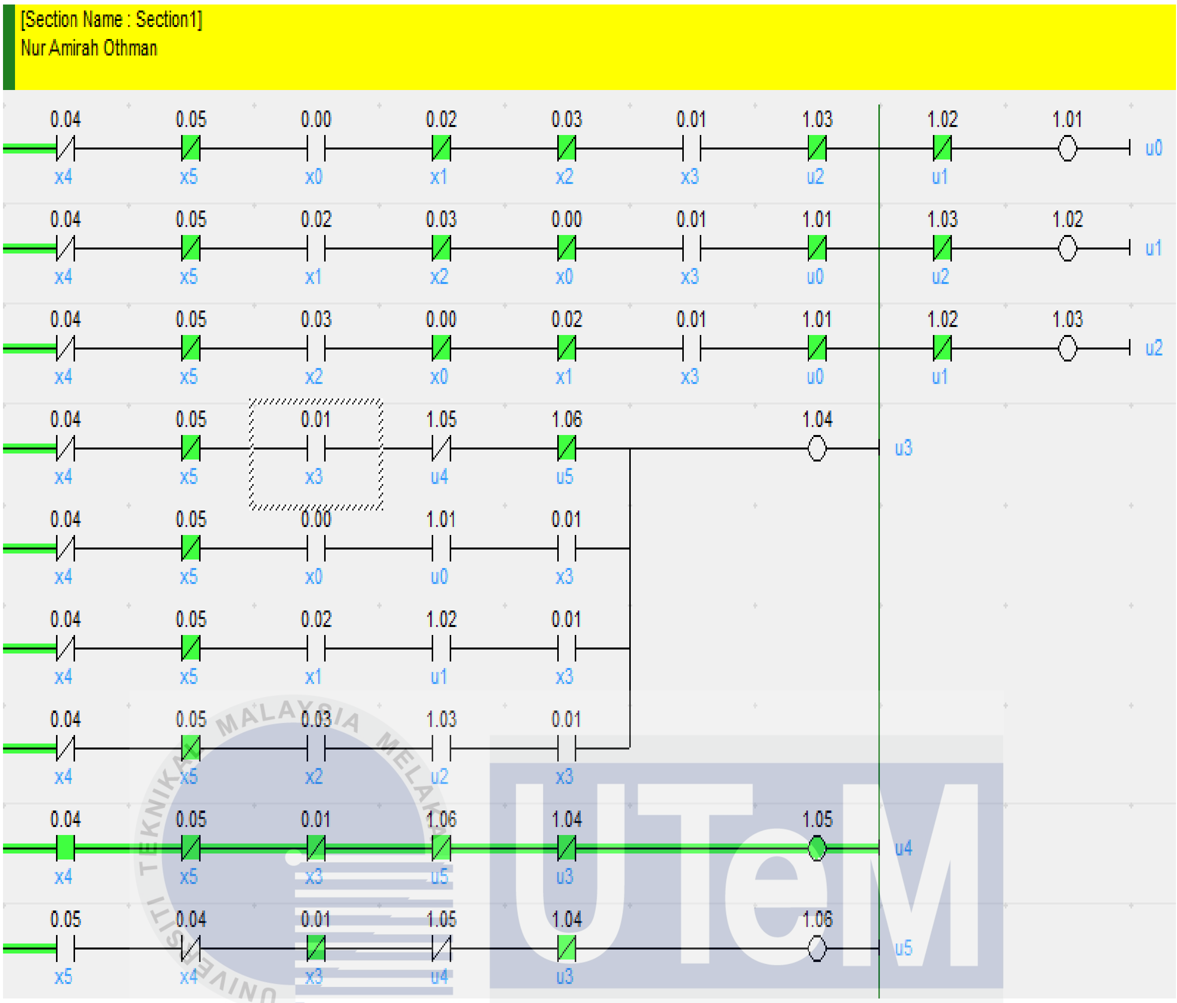
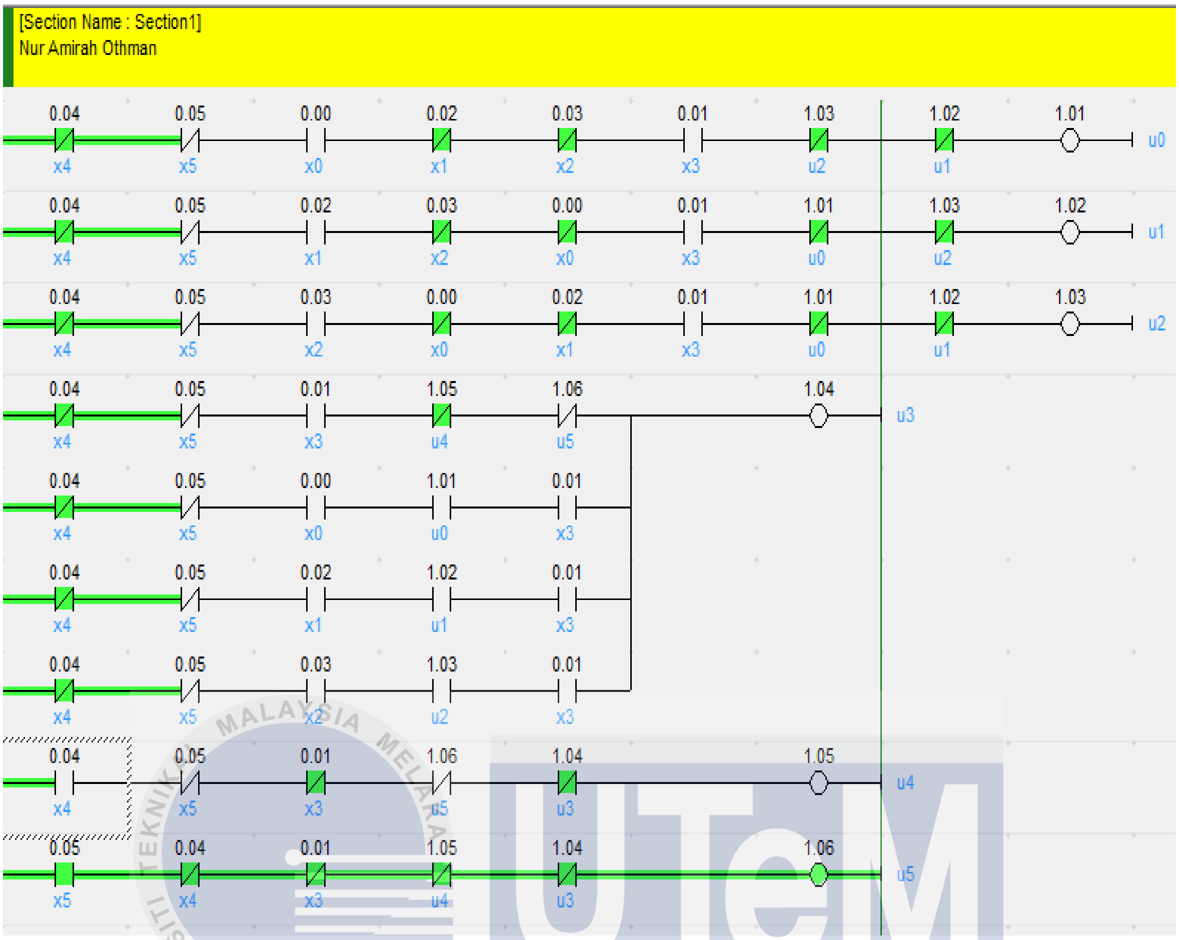

[Section Name : Section1]
Nur Amirah Othman
0.04 x4 0.05 x5 0.00 x0 0.02 x1 0.03 x2 0.01 x3 1.03 u2 1.02 u1 1.01 u0
0.04 x4 0.05 x5 0.02 x1 0.03 x2 0.00 x0 0.01 x3 1.01 u0 1.03 u2 1.02 u1
0.04 x4 0.05 x5 0.03 x2 0.00 x0 0.02 x1 0.01 x3 1.01 u0 1.02 u1 1.03 u2
0.04 x4 0.05 x5 0.01 x3 1.05 u4 1.06 u5 1.04 u3
0.04 x4 0.05 x5 0.00 x0 1.01 u0 0.01 x3
0.04 x4 0.05 x5 0.02 x1 1.02 u1 0.01 x3
0.04 x4 0.05 x5 0.03 x2 1.03 u2 0.01 x3
0.04 x4 0.05 x5 0.01 x3 1.06 u5 1.04 u3 1.05 u4
0.05 x5 0.04 x4 0.01 x3 1.05 u4 1.04 u3 1.06 u5

Figure 3.8 : Input of high steam pressure, x4

Based on Figure 3.8, when the steam pressure is high, the high steam pressure sensor,x4 will detect and thus, the steam pressure reducing valve will move upward, u4 to modulate or reduce the steam pressure. The boiler feed water pump,u0, overflow water valve, u1 and make-up water valve, u2 will not involve in the process of reducing the steam pressure. Thus, u0, u1, u2 are automatically off.


[Section Name : Section1]
Nur Amirah Othman

0.04 x4 | 0.05 x5 | 0.00 x0 | 0.02 x1 | 0.03 x2 | 0.01 x3 | 1.03 u2 | 1.02 u1 | 1.01 u0
0.04 x4 | 0.05 x5 | 0.02 x1 | 0.03 x2 | 0.00 x0 | 0.01 x3 | 1.01 u0 | 1.03 u2 | 1.02 u1
0.04 x4 | 0.05 x5 | 0.03 x2 | 0.00 x0 | 0.02 x1 | 0.01 x3 | 1.01 u0 | 1.02 u1 | 1.03 u2
0.04 x4 | 0.05 x5 | 0.01 x3 | 1.05 u4 | 1.06 u5 | 1.04 u3
0.04 x4 | 0.05 x5 | 0.00 x0 | 1.01 u0 | 0.01 x3
0.04 x4 | 0.05 x5 | 0.02 x1 | 1.02 u1 | 0.01 x3
0.04 x4 | 0.05 x5 | 0.03 x2 | 1.03 u2 | 0.01 x3
0.04 x4 | 0.05 x5 | 0.01 x3 | 1.06 u5 | 1.04 u3 | 1.05 u4
0.05 x5 | 0.04 x4 | 0.01 x3 | 1.05 u4 | 1.04 u3 | 1.06 u5

Figure 3.9 : Input of low steam pressure, x5

Based on Figure 3.9, when the steam pressure is low, the high steam pressure sensor,x5 will detect and thus, the steam pressure reducing valve will move downward, u5 to modulate or reduce the steam pressure. Similarly to Figure 3.8, the boiler feed water pump,u0, overflow water valve, u1 and make-up water valve, u2 will not involve in the process of reducing the steam pressure. Thus, u0, u1, u2 are automatically off.

Overall operation of the safety control logic is summarized in the table below :

Table 3.14 : Summarization of control logic for deaerator system

Activation Input	**Output of Controller**								
	u0 & u3		u1 & u3		u2 & u3		u3	u4	u5
	u0	u3	u1	u3	u2	u3			
x0	ON	ON	OFF	-	OFF	-	-	-	-
x1	OFF	-	ON	ON	OFF	-	-	-	-
x2	OFF	-	OFF	-	ON	ON	-	-	-
x3	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
x4	OFF	OFF	OFF	OFF	OFF	OFF	OFF	ON	OFF
x5	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	ON
u0	-	ON	OFF	-	OFF	-	-	-	-
u1	OFF	-	-	ON	OFF	-	-	-	-
u2	OFF	-	OFF	-	-	ON	-	-	-
u3	-	-	-	-	-	-	-	OFF	OFF
u4	-	OFF	-	OFF	-	OFF	OFF	-	OFF
u5	-	OFF	-	OFF	-	OFF	OFF	OFF	-

After the control logic has been designed then Boolean form is created before performing the formal verification in which using SMV model checker. Hence, from the ladder diagram as in Figure 3.4 – Figure 3.9, below are the following Boolean expressions for the deaerator control logic.

$$
\begin{aligned}
du0 &= \overline{x4}\,\overline{x5}\mathrm{x0\ x3}\overline{u1}\,\overline{u2}\,\overline{x1}\,\overline{x2} \\
du1 &= \overline{x4}\,\overline{x5}\,x1\,\overline{x2x0}x3\,\overline{u2}\,\overline{u0} \\
du2 &= \overline{x4}\;\overline{x5}\,x2\,\overline{x0x1}x3\,\overline{u0}\,\overline{u1} \\
du3 &= \overline{x4}\,\overline{x5}\,x3\,\overline{u4}\,\overline{u5}\ \lor\ \overline{x4x5}x0\ u0\ x3 \\
&\quad \lor\ \overline{x4x5}x1\ u1\ x3\ \lor \overline{x4x5}\ x2\ u2\ x3 \\
du4 &= x4\,\overline{u5x3u5u3} \\
du5 &= x5\,\overline{u4}\,\overline{x3u4}\,\overline{u3} \qquad (3.3)
\end{aligned}
$$

3.3 Procedure 4 : Verification through SMV Model Checker

The verification of logical behavior for deaerator system can be determined by the Boolean rules. In SMV model checker, the transition states are written in terms of the initial and state variables using next and case variable. Modeling language will create by using the SMV model of access module control rules. The codes for the verification process consist of a model expression, controller expression and the specification expressed in temporal logic using CTL rules. The temporal logic is used express the ordering of the event of time by the all the operators that specify properties such as “property *p* will eventually hold” [20]. The properties used in this project are reachability and resettability. The model checker wills automatically checking formally whether a design satisfies some requirements. A coding part of SMV model checker for deaerator system is attached in Appendix A.

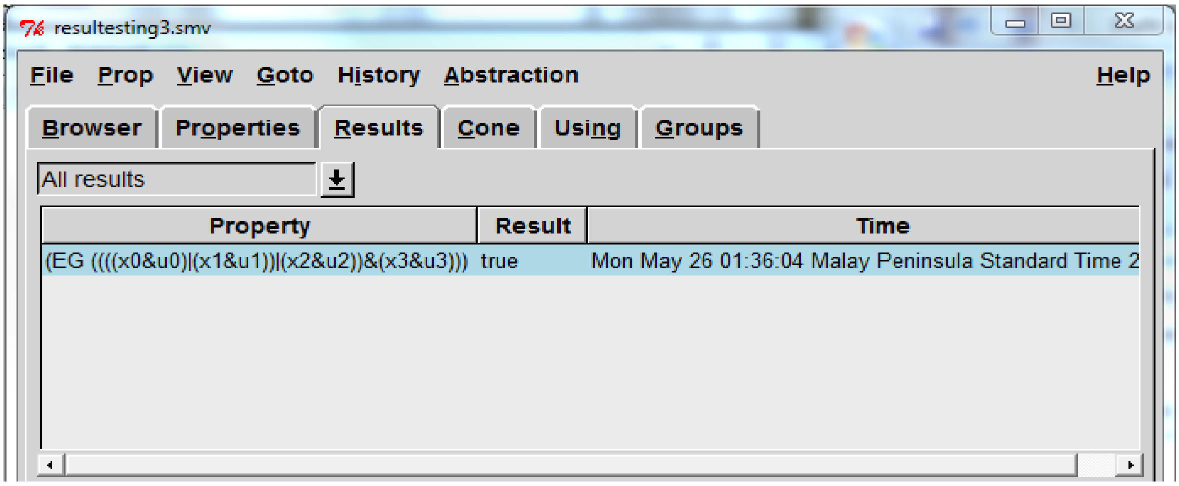
CHAPTER 4

RESULT AND DISCUSSION

By the model checking through SMV model checker, the modeling language will be used in order to verify the model of mathematical model for operational of deaerator system whether it describes the actual behavior of the system or not. Figure 4.1 – Figure 4.6 show the observation is made while verifying the control system and Figure 4.7 show the counterexample part. The temporal properties used in this project are reachability and resettability.

4.1 Reachability

Model checking such as " Property *x* may never hold", will go to a reachability analysis of the state space of the transition system. The reachability analysis will either find a state which property *x* hold or conclude that the state does not never exists [21]. The reachability analysis uses the temporal property of EG *p* where *p* is the negation of the given good property [22]. E represents existential quantification over the traces and G represent indicates an infinite trace along which the property *p* holds globally. Figure 4.1 – Figure 4.4 show the verification execution result applying the several condition of specification through rechability analysis.



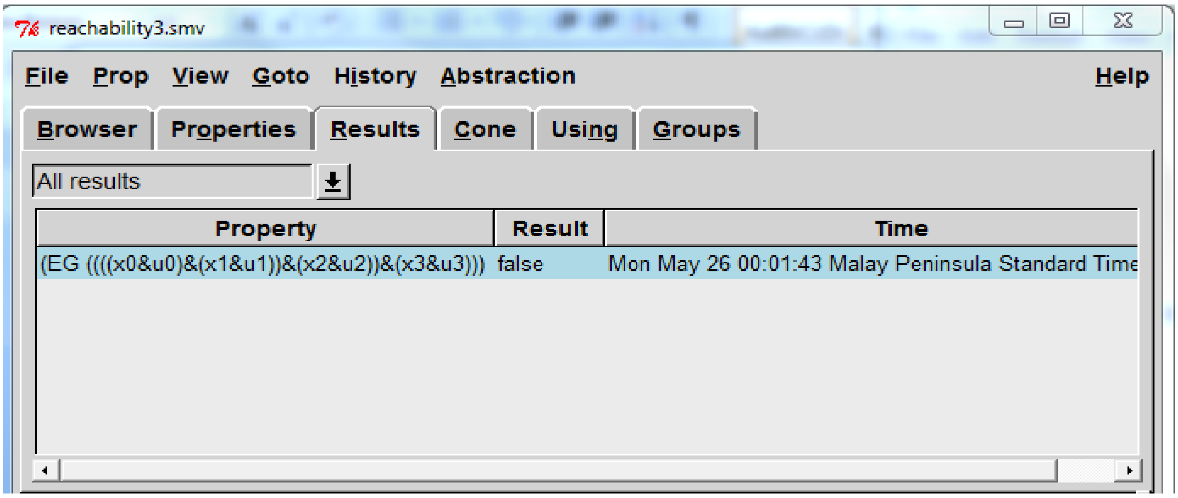
Property	Result	Time
(EG ((((x0&u0)|(x1&u1))|(x2&u2))&(x3&u3)))	true	Mon May 26 01:36:04 Malay Peninsula Standard Time 2

Figure 4.1 : Verification result 1

Figure 4.1 show the specification as in (4.1) :

$$\text{SPEC EG } (((x0 \ \& \ u0) \mid (x1 \ \& \ u1) \mid (x2 \ \& \ u2)) \ \& \ (x3 \ \& \ u3)) \qquad (4.1)$$

In the computation through specification in (4.1), it’s state that there exist some path along *(EG)*, which is only one output will be opened either boiler feed water pump,u0 , overflow valve, u1 or make-up valve, u2. A ‘ | ’ notation used in this specification is a OR gate. These three outputs, u0, u1 and u2 cannot be opened at the same time. Besides, ‘&’ notation used in this specification is AND gate. In this specification also, it specifies that the normal steam pressure, x3 will always sweep with every condition of water level by activation of steam pressure reducing valve hold,u3. Hence, by the specification description, the model checker executed the TRUE verification result. This verification result is same as the operational control logic designed in Figure 3.4 – Figure 3.7 which is the outputs cannot open at the same time. Thus, it is a safe specification for deaerator model system.



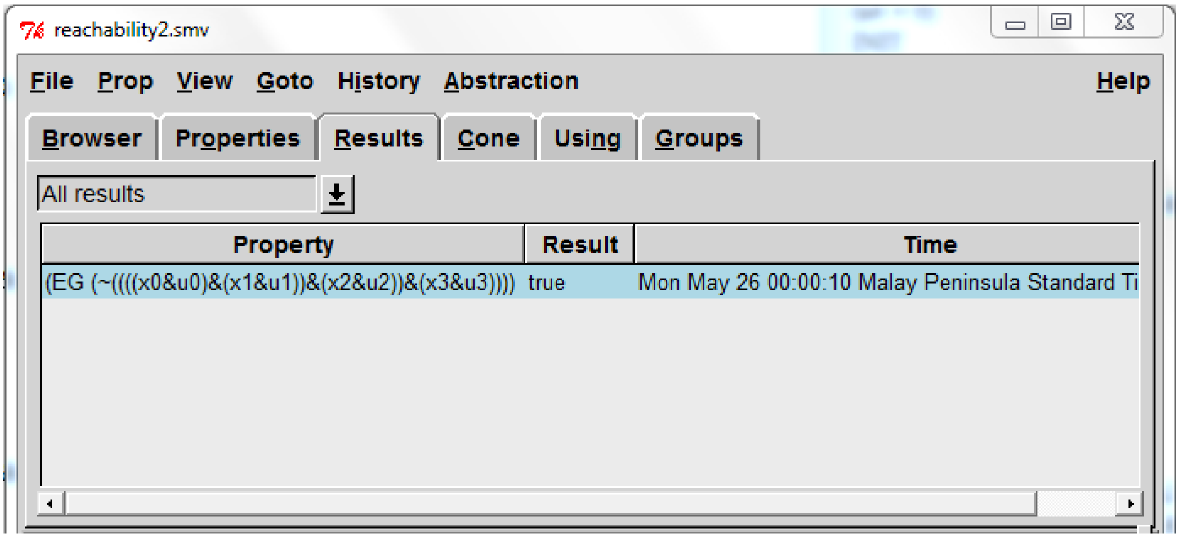
Property	Result	Time
(EG ((((x0&u0)&(x1&u1))&(x2&u2))&(x3&u3)))	false	Mon May 26 00:01:43 Malay Peninsula Standard Time

Figure 4.2 : Verification result 2

Figure 4.2 show the specification used as in (4.2) :

SPEC EG ((x0 & u0) **&** (x1 & u1) **&** (x2 & u2) **&** (x3 & u3)) (4.2)

In the computation through specification in (4.2), it show that there exists some path along *(EG),* which all the inputs and the outputs is detected at the same time during the computation path by using AND gate notation by **'&'**. The inputs are normal steam pressure sensor,x3 ,normal water level sensor,x0 , high water level sensor ,x1 and low water level sensor,x2. While the outputs are boiler feed water pump,u0 , overflow water sensor,u1, make-up water valve,u2 and hold steam pressure reducing valve,u3. By comparing to the model system and control logic in methodology part, the condition will never happen as the specification stated. It is not a safe condition for deaerator system. Thus, the model checker executed FALSE verification result.



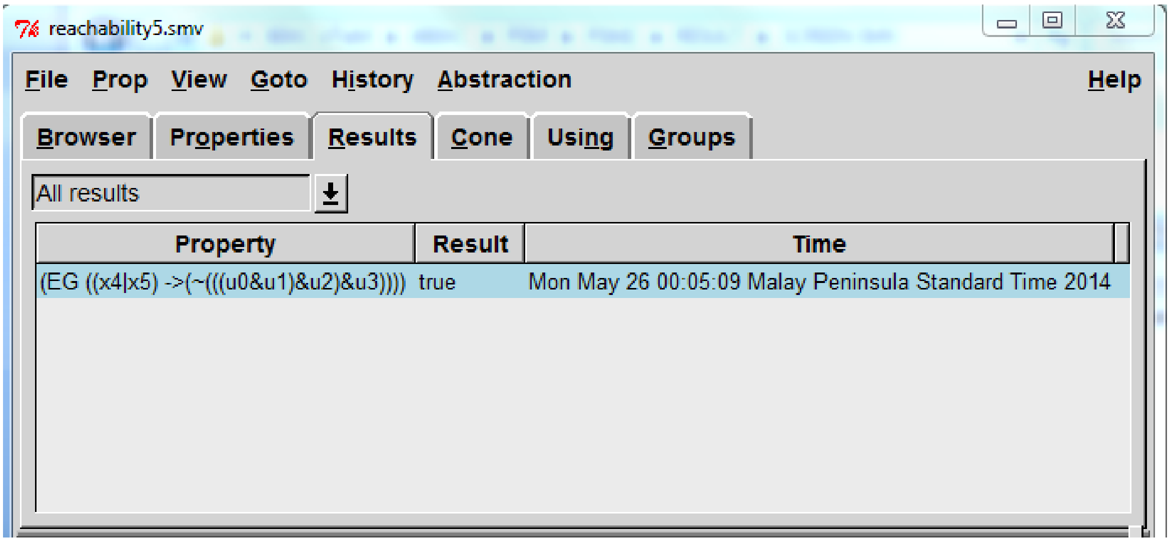
Property	Result	Time
(EG (~((((x0&u0)&(x1&u1))&(x2&u2))&(x3&u3))))	true	Mon May 26 00:00:10 Malay Peninsula Standard Ti

Figure 4.3 : Verification result 3

Figure 4.3 show the specification used as in (4.3);

SPEC EG ~((x0 & u0) & (x1 & u1) & (x2 & u2) & (x3 & u3)) (4.3)

The computation through specification in (4.3) is the same as in Figure 4.2 but the specification in (4.3) used the "~" notation in front of the expression. It means that the specification will never happen in the system. This specification can be proved by the model system and control logic in methodology part. Thus, the model checker executes TRUE verification result. Thus, the specification used is safe for the deaerator system.



Property	Result	Time
(EG ((x4|x5) ->(~(((u0&u1)&u2)&u3))))	true	Mon May 26 00:05:09 Malay Peninsula Standard Time 2014

Figure 4.4 : Verification result 4

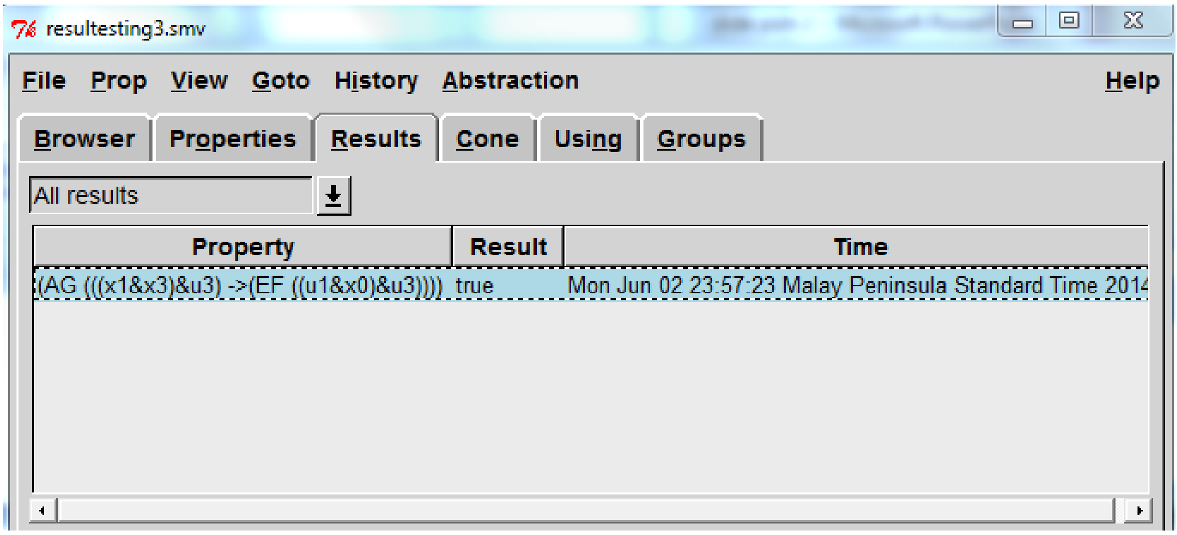
Figure 4.4 show the specification used as in (4.4):

SPEC EG ((x4 | x5) -> ~ (u0 & u1 & u2 & u3)) (4.4)

In the computation through specification in (4.4), it show that there exists some path along *(EG)*, which either high steam pressure,x4 or low steam pressure,x5 is detected, the boiler feed water pump,u0, overflow valve, u1, make-up valve, u2 and hold steam pressure reducing valve,u3 must cannot be opened. A '~' notation used in this specification state the it cannot happen in the system. Thus, the model checker executed TRUE verification result. This verification result is same as the operational control logic designed in Figure 3.8 and Figure 3.9. Hence, it is a safe specification for deaerator system.

4.2 Resettability

The resettability property use AG EF *p* in CTL that require branching time semantic [23].), "AG *p*" describes that condition *p* is always or globally Second, "EF *p*" describes that there exists some path that eventually in the future satisfies *p*. Besides, resettability state that it will eventually resettable no matter happen in the middle of operation. Figure 4.5 and Figure 4.6 show the verification execution result applying the several condition of specification by using resettability analysis.



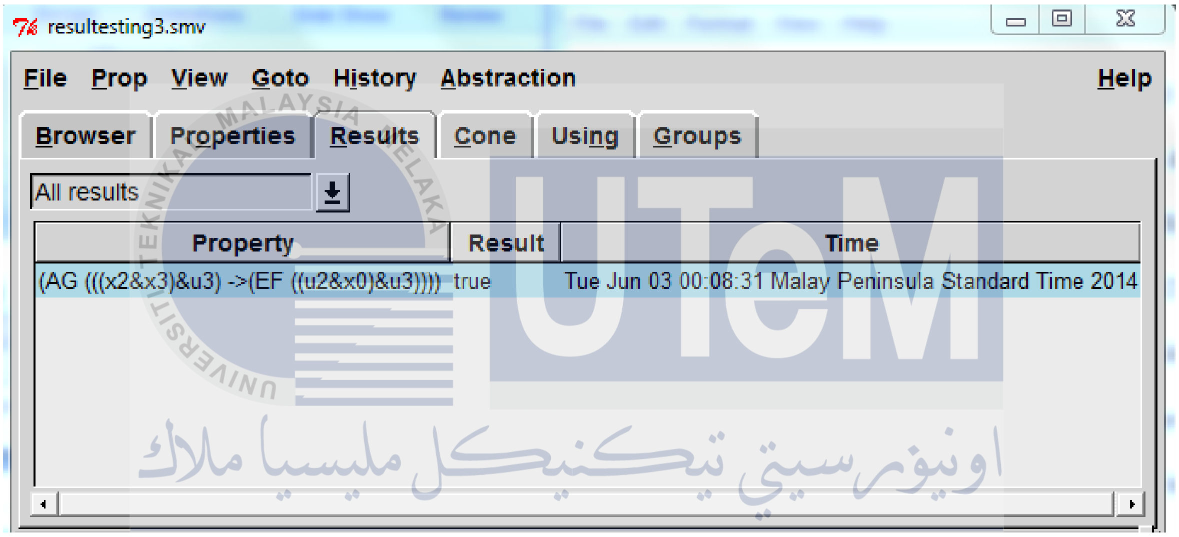
Property	Result	Time
(AG (((x1&x3)&u3) ->(EF ((u1&x0)&u3))))	true	Mon Jun 02 23:57:23 Malay Peninsula Standard Time 2014

Figure 4.5 Verification result 5

Figure 4.5 show the specification used as in (4.5):

$$\text{SPEC AG ((x1 \& x3 \& u3) -> EF (u1 \& x0 \& u3))} \tag{4.5}$$

In the computation through specification (4.5) show that it always happen *(AG)* when high water level,x1 detected, THEN it will exist in the future *(EF)* that overflow water valve, u1 will automatically open in oder to retrive back to the normal water level. Thus, by the specification, the model checker show TRUE verification result. The verification result is same to the control logic designed in Figure 3.5 and the model system in Table 3.7. Thus, it is safe specification for deaerator system.



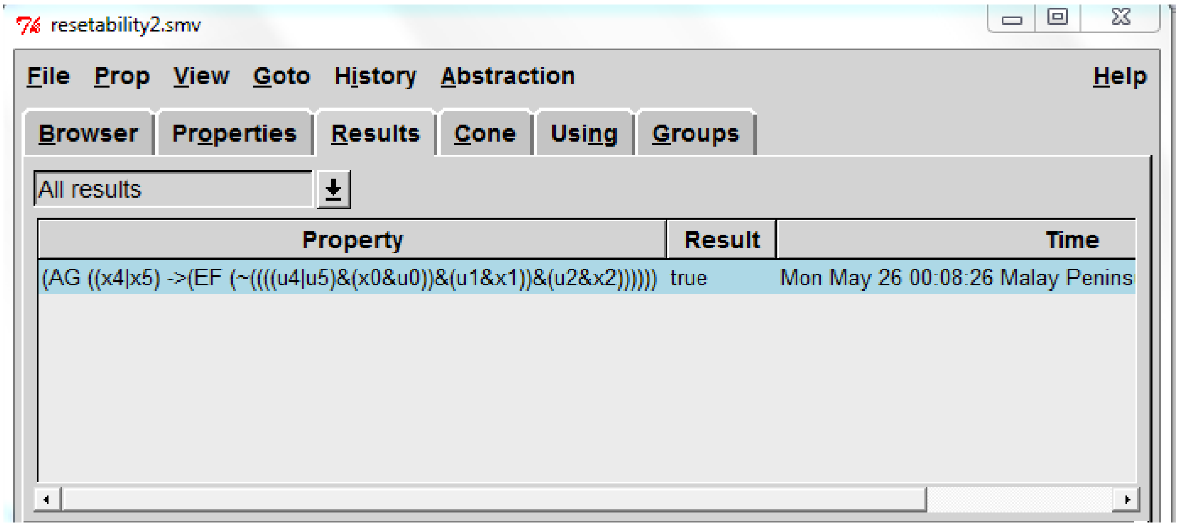
Property	Result	Time
(AG (((x2&x3)&u3) ->(EF ((u2&x0)&u3))))	true	Tue Jun 03 00:08:31 Malay Peninsula Standard Time 2014

Figure 4.6 Verification result 6

Figure 4.6 show the specification used as in (4.6):

$$\text{SPEC AG ((x2 \& x3 \& u3) -> EF (u2 \& x0 \& u3))} \tag{4.6}$$

In the computation through specification (4.6) show that it always happen *(AG)* when low water level,x2 detected, THEN it will exist in the future *(EF)* that make-up water valve, u2 will automatically open in order to retrive back to the normal water level. Thus, by the specification, the model checker show TRUE verification result. The verification result is same to the control logic designed in Figure 3.6 and the model sytem in Table 3.7. Thus, it is safe specification for deaerator system.


resetability2.smv
File Prop View Goto History Abstraction Help
Browser Properties Results Cone Using Groups
All results

Property	Result	Time
(AG ((x4|x5) ->(EF (~((((u4|u5)&(x0&u0))&(u1&x1))&(u2&x2))))))	true	Mon May 26 00:08:26 Malay Penins

Figure 4.7 : Verification result 7

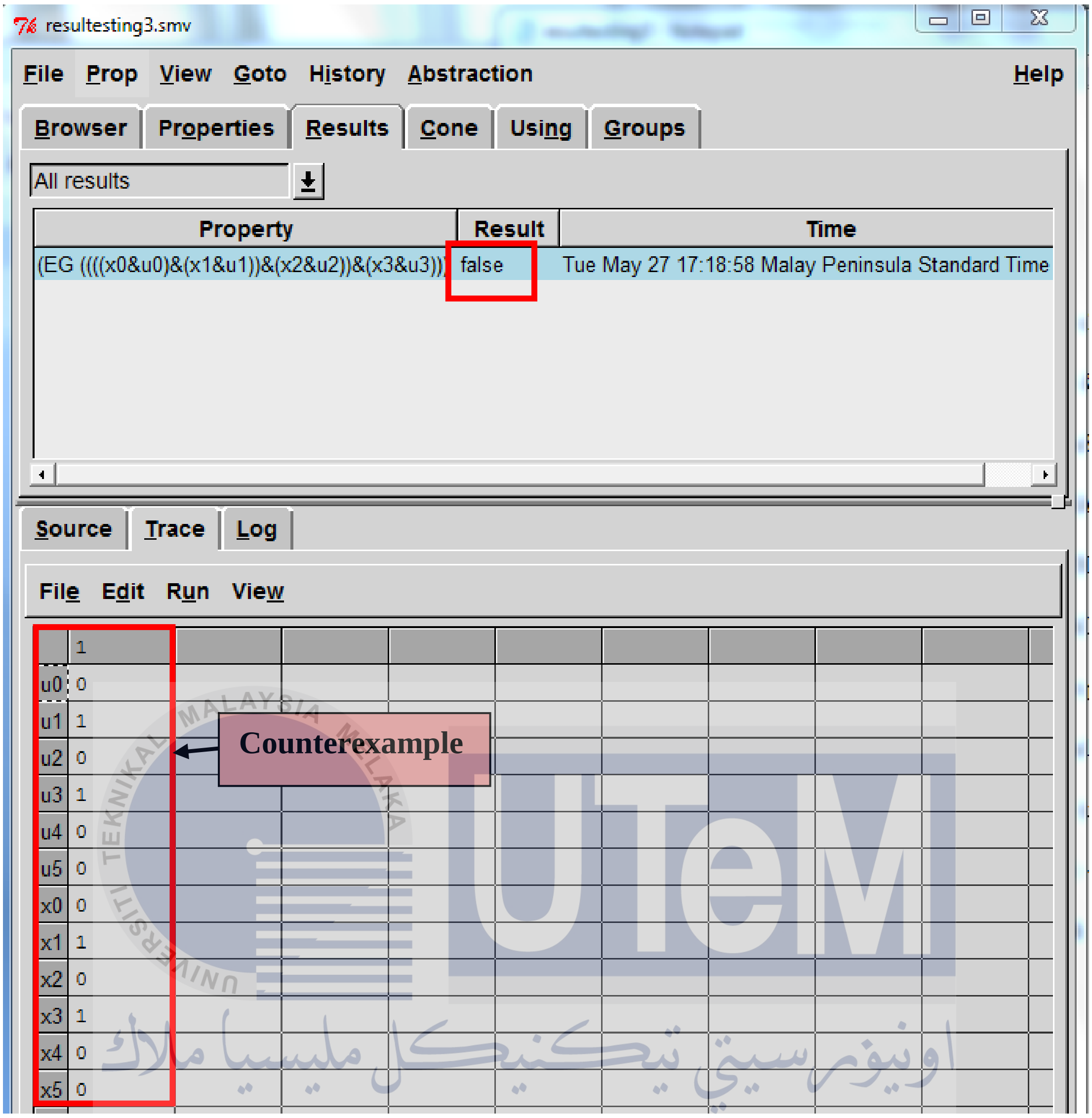
Figure 4.7 show the specification used as in (4.7):

SPEC AG ((x4 | x5) -> EF! ((u4 | u5) & (x0 & u0) &
(u1 & x1) & (u2 & x2))) (4.7)

In the computation through the specification in (4.6), it specify that in all path when the high steam pressure or low steam pressure is detected, it will eventually not happen for the boiler feed water pump,u0 , overflow valve,u1 and make-up valve,u2 to open. The model checker show TRUE verification result. This specification is same as in control logic designed in Figure 3.8 and Figure 3.9.

4.3 Counterexample of SMV software

SMV model checker produce counterexample for specification that do not hold in the given model [24]. In counterexample, if none of the behaviors of the system violates the given specification, the model of the system will correct. Otherwise, the model checker will automatically execute the counterexample of the model system to show why the specification is false. Below is a example of FALSE veriffication result and its counterexample.



Property	Result	Time
(EG ((((x0&u0)&(x1&u1))&(x2&u2))&(x3&u3))	false	Tue May 27 17:18:58 Malay Peninsula Standard Time

	1
u0	0
u1	1
u2	0
u3	1
u4	0
u5	0
x0	0
x1	1
x2	0
x3	1
x4	0
x5	0

Figure 4.8 : Counterexample

From the specification in Figure 4.8 :

SPEC EG ((x0 & u0) & (x1 & u1) & (x2 & u2) & (x3 & u3)) (4.8)

The specification (4.8) state that boiler feed water pump,u0 , overflow valve,u1 , make-up valve,u1 and steam pressure reducing valve,u3 is open simultaneously. The model checker show FALSE veriffication result because in the model and controller, that condition will never happen. From counterexample, only u1 and u3 are open while u2 and u0 are close. u2 and u0 do not hold in the model and controller. Hence, this proved that the specification in Figure 4.2 is FALSE veriffication and violate from the system behavior.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

In this chapter, the conclusion of the project is conclude and some suggestions recommendation for further development of this project.

5.1 Conclusion

Model checking tools using SMV model checker uses the finite state machine for the verification of the system.. It is based on the mathematical expression of the variable to be verified and check that the model system using Boolean form satisfies to the actual system or not. The model checker will show true if the model system correct while false is shown if the model system is false. After model of mathematical expression for the system, it is easy to check with the different condition and absolutely will see the change of condition with the different input set.

After making the model, it will be easy to check and see the small changes in the signal and condition of the model will change the behavior of model. The computational method like model checking is used to verify the correctness of the model of an automation system through specifying properties of CTL and describe it in temporal logic. The temporal logic will show either the specification is violates or not. Temporal logic is a specification language based on the temporal logic for expressing into formal verification. All the temporal logic used for the verification must to be fully understand before start to make a checking. Beside, in the model checking, the design of control logic is very important to make the model functioned properly. The control logic and the model must be operated together in order to make sure the specification created satisfies both control logic and the model. For the model design, it is importance to check the equation of mathematical expression before performing the verification. As a conclusion, the reachability and resettability property used in this project are performed well based on the theoretical result and justification of the verification result.

5.2 Recommendations

The model checking technique is interesting methods that can be make a more research in the future. There are many things that need to be explored in this formal verification. In the future, it is more interesting challenge if this project is continue to the Boiler operation part. The complex specification will be produced by applying the Boiler operation part in this project. Besides, as a recommendation, it can be recommending that to use the NuSMV as a model checker tools for the future. NuSMV model checker can check the LTL properties and can be compared to the CTL temporal logic. In addition, it is become interesting and full of challenge to research and focus more on the counterexample part in any automation system. It is because the study of process for counterexample can gain a new knowledge that need to explore in mathematical and computer science.

REFERENCES

[1] D.Lindsley, “Steam and water circuits”, *Power-plant Control and Instrumentation*, United Kingdom, Control Engineering Series 58,2000, pp 17.

[2] B.Woodruff, B.Lammers and F.Lammers , “Auxillary Steam-Plant Equipment”, *Steam Plant Operation*,8th ed,Amerika, The McGraw-Hill,2005, pp 662.

[3] “*Formal Method*”, The Institution of Engineering and Technology,2011

[4] M.Clarke, “The Birth of Model Checking”, Department of Computer Science Carnegie Mellon University Pittsburgh, PA,USA,

[5] Q. Tran, A. Mulepati, “Validation of Automation Systems using Temporal Logic Model Checking and Groebner Bases”, 2010, World Academy of Science, Engineering and Technology, Lamar University, U.S.A.

[6] A.Artale, “Computational Tree Logic (CTL)”, 2010, Faculty of Computer Science, Free University of Bolzano Room 2 03.

[7] A. Juarez-Dominguez, N. Day, “Working with Cadence MSV models in Moscow ML”, 2012, Cheriton School of Computer Science University of Waterloo, Canada.

[8] A.Raji, P.Dhaussy,and B.Aizier, “Automating Context Description for Software Formal Specification”, 2010, Universite Europeene de Bretagne.

[9] L.Sun and M.Leng, “Formal Derivation of the second kind Stirling numbers with PAR method”,2010

[10] P.Olgivie, “Formal Method in Requirements Engineering”,2000

[11] M.Asztalos, I.Madari, and T.Vajk, “Formal Verification of Model Transformations an Automated Framework”, 2010, Department of Automation and Applied Informatics.

[12] J.Valkonen, M.Koskimies and V.Petterson, “Formal Verification of Safety I&C System Design: Two Nuclear Power Plant Related Applications”, 2008, Department of Information and Computer Science.

[13] K. Heljanko, “A Symbolic Model Checking Approach to Verifying Satellite Onboard Software”,2011, Department of Information and Computer Science

[14] S.Inchi, “Arithmethic Boolean Expression Manipulator Using BDDs”,1996

[15] I.Opris, “A deaerator model”, Department of Energy Use and Generation

[16] *Operation, Maintenance and Parts Manual*, Cleaver-Brooks, Aqua-Chem, Inc,1998, pp.8

[17] J. Gunnarson, " Algebraic Methods for Discrete Event System – A Tutorial", Proc. Of IEE WODES'96,Edinburgh (GB), pp. 18-30,1996

[18] ScienceDaily, "*Mathematical Model*", http://www.sciencedaily.com/articles/m/mathematical_model.htm

[19] Saifulza Alwi and Yasutaka Fujimoto, "*On A Safety of Sequential Control System Based on Grobner Bases Computation*", Department of Electrical and Computer Engineering, 2010

[20] Y.Crame, "Boolean models and methods in Mathematic Computer Science and Engineering, Ecylopedia of Mathematic and its approaches, pp. 559, 2010

[21] G.Kart Msc, "Comparing the LTSmin and NuSMV reachability tools via automatic translation of their respective input languanges" , University of Tanta, pp.6, 2013

[22] K.Vasanta Lavshmi, "Checking Temporal Properties of Presburger Counter System using Reachability Analysis", India Institute Department, pp. 2

[23] Z.Hassan, "Incremental, Inductile CTL Model Checking", ECFE Department, pp.2, 2011

[24] D. Wijesekera, " Relating Counterexample to Test Cases in CTL Model Checking Specification", Department of Information and Software, pp.2, July 2007.

APPENDICES

APPENDIX A : Codes for formal verification using SMV Model Checker

```
MODULE main
VAR
x0 : boolean;   -- normal water level
x1 : boolean;   -- high water level
x2 : boolean;   -- low water level
u0 : boolean;   -- boiler feed water pump
u1 : boolean;   -- overflow water valve
u2 : boolean;   -- make-up waer valve
x3 : boolean;   -- normal steam pressure
x4 : boolean;   -- high steam pressure
x5 : boolean;   -- low steam pressure
u3 : boolean;   -- steam pressure reducing hold
u4 : boolean;   -- steam pressure reducing move upward
u5 : boolean;   -- steam pressure reducing move downward
INIT
(x0 = 0)
INIT
(x1 = 1)
INIT
(x2 = 0)
INIT
(u0 = 0)
INIT
(u1 = 1)
INIT
(u2 = 0)
INIT
(x3 = 1)
INIT
(x4 = 0)
INIT
(x5 = 0)
INIT
(u3 = 1)
INIT
(u4 = 0)
INIT
(u5 = 0)

SPEC EG (((x0 & u0) | (x1 & u1) | (x2 & u2)) & (x3 & u3)) -- REACHABILITY
SPEC EG ~((x0 & u0) & (x1 & u1) & (x2 & u2) & (x3 & u3)) -- REACHABILITY
SPEC EG ((x0 & u0) & (x1 & u1) & (x2 & u2) & (x3 & u3)) --REACHABILITY
SPEC EG ((x4 | x5) ->  ~(u0 & u1 & u2 & u3)) -- REACHABILITY
SPEC AG ((x1 & x3 & u3) -> EF ( u1 & x0 & u3)) -- RESETTABILITY
SPEC AG ((x2 & x3 & u3) -> EF ( u2 & x0 & u3)) – RESETTABILITY
SPEC AG  ((x4 | x5) -> EF! ((u4 | u5) & (x0 & u0) & (u1 & x1) & (u2 & x2))) –
RESETTABILITY

MODULE storagetank
ASSIGN
next(x0) :=
case
        (~x0 &  ~x1 & ~x2 & x3 & ~u0 & u1 & ~u2 & u3) | (~x0 & ~x1 & ~x2 & x3 & ~u0 & ~u1
& u2 & u3) |
        (~x1 & ~x2 & x3 & ~u1 & u3 & ~x0 & ~u0 & u2)  : 1;
1 : x0;
esac;
next(x1) :=
case
        (~x0 & ~x1 & x3 & ~u0 & ~u1 & u2 & u3 & ~x2) | (x0 & ~x1 & ~x2 & x3 & u0 & ~u1 &
~u2 & u3) |
        (~x0 & ~x1 & ~x2 & x3 & ~u0 & ~u1 & u2  & u3)    : 1;
1 : x1;
esac;

next(x2) :=
case
        (~x0 & ~x1 & x3 & ~u0 & ~u1 & u2 & u3 & ~u2) | (~x0 & ~x1 & ~x2 & x3 & u0 & ~u1
& ~u2 & u3): 1;
1 : x2;
esac;
next(x3) :=
case
        (~x0 & x1 & ~x2 & x3 & ~u0 & u1 & ~u2 & u3) | ( ~x0 & ~x1 & ~x2 & x3 & ~u0 & u1 &
~u2 & u3) |
        (x0 & ~x1 & ~x2 & x3 & u0 & ~u1 & ~u2 & u3) | (~x0 & ~x1 & ~x2 & x3 & u0 & ~u1 &
~u2 & u3) |
        (~x0 & ~x1 & x2 & x3 & ~u0 & ~u1 & u2 & u3) | ( ~x0 & ~x1 & ~x2 & x3 & ~u0 & ~u1
& u2 & u3) |
        (x0 & ~x1 & ~x2 & x3 & u0 & ~u1 & ~u2 & u3) | ( ~x0 & ~x1 & ~x2 & x3 & ~u0 & ~u1
& u2 & u3) :1;
1 : x3;
esac;
MODULE exchangepacking
ASSIGN
next(x3) :=
case
        (~x3 & ~x4 & ~x5 & ~u3 & u4 & ~u5) | (~x3 & ~x4 & ~x5 & ~u3 & ~u4 & u5) : 1;
1 : x3;
esac;

next(x4) :=
case
        (~x3 & ~x4 & ~x5 & u3 & ~u4 & ~u5) | ( ~x3 & ~x5 & ~u3 & u4 & ~u5 & ~x4) |
        (~x4 & ~x5 & u3 & ~u4 & ~u5 & ~x3) | (~x3 & ~x4 & ~u3 & ~u4 & u5 & ~x5) | (x3 &
~x5 & ~x4 & u3 & ~u4 & ~u5) : 1;
1 : x4;
```

```
esac;
next(x5) :=
case
            (~x3 & ~x4 & ~x5 & u3 & ~u4 & ~u5) | (~x3 & ~x5 & ~u3 & u4 & ~u5 & ~x4) | (x3 &
~x4 & ~x5 & u3 & ~u4 & ~u5) |
            (~x3 & ~x4 & ~u3 & ~u4 & u5 & ~x5) | ( ~x4 & ~x5 & u3 & ~u4 & ~u5 & ~x3) : 1;
1 : x5;
esac;
MODULE controller
ASSIGN
next (u0) :=
case
            ( ~x4 & ~x5 & x0 & ~x1 & ~x2 & x3 & ~u1 & ~u2) :1 ;
1 : u0;
esac;
next(u1) :=
case
            (~x4 & ~x5 & x1 & ~x2 & ~x0 & x3 & ~u2 & ~u0) : 1 ;
1 : u1;
esac;
next(u2) :=
case
            ( ~x4 & ~x5 & x2 & ~x0 & ~x1 & x3 & ~u0 & ~u1) : 1 ;
1 : u2;
esac;
next(u3) :=
case
            ((~x4 & ~x5 & x3 & ~u4 & ~u5) | (~x4 & ~x5 & x0 & x3 & u0) | (~x4 & ~x5 & x1 & x3
& u1) | (~x4 & ~x5 & x2 & x3 & u2)): 1;
1 : u3;
esac;
next(u4) :=
case
            (x4 & ~x5 & ~x3 & ~u5 & ~u3 ) : 1 ;
1 : u4;
esac;
next(u5) :=
case
            (x5 & ~x4 & ~x3 & ~u4  & ~u3 ): 1 ;
1 : u5;
esac;
```

Bibliography Form

Personal Data

Name : NUR AMIRAH BINTI OTHMAN

IC No : 910424-11-5622

Date and Place of Birth : 24 APRIL 1991 (TERENGGANU)

Sex : FEMALE

Nationality : MALAYSIAN

Permanent Address : PT 30494, TAMAN KALUNGAN IMPIAN, 21200
BANGGOL AIR LILIH, KUALA TERENGGANU,
TERENGGANU

Contact No : 019 - 6756094

Academic Qualification

Primary School	Name 1. SEKOLAH KEBANGSAAN KIJAL Place : KEMAMAN, TERENGGANU 2. SEKOLAH KEBANGSAAN PAKA Place : PAKA, TERENGGANU	Year start : 1998 Year end : 2002 Year start : 2003 Year end : 2003
Secondary School	Name 1. SEKOLAH MENENGAH AGAMA SULTAN ISMAIL DUNGUN, TERENGGANU Place : DUNGUN, TERENGGANU 2. SEKOLAH MENENGAH KEBANGSAAN PAKA, DUNGUN, TERENGGANU Place : PAKA, TERENGGANU	Year start : 2004 Year end : 2006 Year start : 2007 Year end : 2008
High Academic Qualification (matriculation, polytechnic, university)	Name 1. KOLEJ MATRIKULASI PAHANG Place : GAMBANG, PAHANG	Year start : 2009 Year end : 2010
High Academic Qualification	Name 1. UNIVERSITI TEKNIKAL MALAYSIA MELAKA Place : DURIAN TUNGGAL, MELAKA	Year start : 2010 Year end : 2014

Title of Project : A FORMAL METRHOD FRAMEWORK FOR AUTOMATED VERIFICATION OF A DEAERATOR SYSTEM

Supervisor : DR SAIFULZA BIN ALWI @ SUHAIMI