

**SUPERVISORY CONTROL AND DATA ACQUISITION FOR A ROBOTIC
ARM VIA GRAFCET**

By

SYAHRIL IZWAN BANIRAM

FINAL PROJECT REPORT

Submitted to the Electrical & Electronics Engineering Programme
in Partial Fulfillment of the Requirements
for the Degree
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)

Universiti Teknologi Petronas
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

© Copyright 2007
by
Syahril Izwan Baniram, 2007

CERTIFICATION OF APPROVAL

SUPERVISORY CONTROL AND DATA ACQUISITION FOR A ROBOTIC ARM VIA GRAFCET

by

Syahril Izwan Baniram

A project dissertation submitted to the
Electrical & Electronics Engineering Programme
Universiti Teknologi PETRONAS
in partial fulfilment of the requirement for the
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)

Approved:



Dr. Nordin Saad
Project Supervisor

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK

June 2007

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.



Syahril Izwan Baniram

ABSTRACT

The aim of this report is to focus on the basic programming structure of PLC and its application in industry. This report emphasizes understanding of Grafset language through the basic knowledge of ladder diagram method. A robotic arm will be programmed to do some specific tasks. The main programming language to be used would be Grafset. Notably, since the language is relatively new, several aspects of the programming method would be required to be explored. Specific software named AUTOMGEN is available and it provides some features inclusive of pneumatic/electrical/hydraulic simulation module for any automation application and SCADA 3D process simulation. Programming of the Grafset would be conducted by using State-by-State (SS) approach and classic truth table simplification. The truth table is to be constructed based on the timing diagram of the motions of the robotic arm. Previous work on the robotic arm controlled via PLC programming using ladder diagram had shown a successful achievement. In this project, Grafset language is opted to be used for programming because of its concurrency for both simplicity and easy understanding. It would expect that a clear description of programming using Grafset could be achieved and the general features of programming a PLC could be understood.

ACKNOWLEDGEMENT

I am indebted to many individuals who are helping me in completing my Final Year Project where the presence is the essence to make my training successful. They are people of my respects who involve directly or indirectly throughout this project.

Thank to Allah S.W.T for His permission for letting me complete this project within time constraint. My big gratitude goes to my supervisor, Dr. Nordin Saad, for guiding me throughout this project. His continuous help in assisting me to do this project stimulate me to finish all the tasks allocated without any doubt. His invaluable idea generation and priceless advices is my milestone for my successfulness.

Not forget Mr. Azhar Zainal Abidin, Lab technologist, helping me in preparing the robot's components, prepare the lab sessions, share his experiences in conducting the PLC and help me in troubleshooting some problems during the lab experiment.

I also extend my special thanks to Mr. Carl Swanerpoel from Adroit Co. who has led the way of this project with unwavering support, along with the strong endorsement from his senior colleague Mr. Freddy Mare who has come along from South Africa for a precious moment of idea sharing.

The SCADA for Robotic Arm via Grafset also gained immensely from the hard work of my colleague Mr. Faiz Fauzi who has helped me prepare several materials regarding the subject. Tons of thank to all of people who is directly or indirectly involved in realizing this project. May this portion is enough to appreciate all of them.

TABLE OF CONTENTS

ABSTRACT.....	iv
ACKNOWLEDGEMENT.....	v
LIST OF TABLES.....	viii
LIST OF FIGURES.....	xi
LIST OF ABBREVIATIONS.....	x
CHAPTER 1 INTRODUCTION.....	1
1.1 Background of Study.....	1
1.2 Problem Statement.....	2
1.2.1 Programming Grafcet.....	3
1.2.2 SCADA Interface.....	4
1.3 Objective.....	4
1.4 Scope of Study.....	4
1.4.1 Understanding Grafcet.....	4
1.4.2 Designing SCADA Interface.....	4
CHAPTER 2 LITERATURE REVIEW AND THEORY.....	5
2.1 Theory of Grafcet.....	5
2.1.1 Grafcet Concept and Conditions.....	7
2.1.2 Basic Notions.....	7
2.2 Theory of Pick-and-Place Robot.....	10
2.2.1 Robotic Arm Operation.....	10
2.2.2 Control Objectives.....	12

CHAPTER 3 METHODOLOGY.....	13
3.1 Procedure Identification.....	13
3.2 Tools.....	15
3.2.1 Software.....	15
3.2.2 Hardware.....	15
3.3 Project Work.....	18
3.3.1 Project Architecture.....	18
3.3.2 Protocol.....	18
3.3.3 PLC-Server Connection.....	19
3.3.4 I/O Assignment.....	20
3.3.5 Programming Grafcet.....	22
CHAPTER 4 RESULT AND DISCUSSION.....	24
4.1 Findings.....	24
4.1.1 Programming Techniques.....	24
4.1.2 Programming Code.....	26
4.1.3 Client-Server Communication.....	29
4.1.4 Control System.....	35
4.2 Discussion.....	36
4.2.1 Programming Techniques.....	36
4.2.2 Programming Code.....	39
4.2.3 Client-Server Communication.....	40
4.2.4 Control System.....	42
CHAPTER 5 CONCLUSION AND RECOMMENDATIONS.....	44
5.1 Conclusion.....	44
5.2 Recommendations.....	46
REFERENCES.....	47
APPENDICES.....	48
Appendix A Full Program Code.....	49
Appendix B OMRON SYSMAC PLC Address.....	50
Appendix B PLC COMMUNICATIONS & GRAFCET.....	51

LIST OF TABLES

Table 2.1: Logic Form of Step and Transition.....	9
Table 3.1: Inputs and Outputs (Inputs-outputs) Assignment.....	20
Table 3.2: Bit Representation.....	22
Table 4.1: Truth Table for Bit Transitions.....	26

LIST OF FIGURES

Fig. 2.1: Steps.....	7
Fig. 2.2: Transitions.....	8
Fig. 2.3: Directed Arcs.....	8
Fig. 2.4: Physical Layout of the Robot.....	10
Fig. 2.5: Mechanism of the Robot.....	11
Fig. 3.1: Flow Chart Representation of the Project Planning.....	14
Fig. 3.2: General Overview of Project.....	18
Fig. 3.3: Relay Circuit of Representing the VDIN.....	21
Fig. 3.4: State-by-State (SS) Transitions Method.....	22
Fig. 4.1: Full Presentation of SS Method.....	25
Fig. 4.2: Implementation of DPSS.....	27
Fig. 4.3: Emergency Switch Code Generated.....	27
Fig. 4.4: Condition of Mode Operation and Emergency.....	28
Fig. 4.5: SET and RESET function.....	28
Fig. 4.6: Tagging Pane of OMRON OPC Server.....	30
Fig. 4.7: Login Security Page.....	30
Fig. 4.8: General Overview Window.....	31
Fig. 4.9: Specific Instrument Window.....	31
Fig. 4.10: Whole System Status Window.....	32
Fig. 4.11: Alarm Window.....	32
Fig. 4.12: Trending Window.....	33
Fig. 4.13: Timing Diagram of Typical Operation.....	34
Fig. 4.14: Architecture of Operation (SS Frame Transition).....	37
Fig. 4.15: SCADA System Architecture.....	41
Fig. 4.16: Block Diagram of Closed Loop Operation.....	42
Fig. 4.17: Block Diagram of Open Loop Operation.....	43

LIST OF ABBREVIATIONS

DIN	-	Digital Input
DPSS	-	Dual Pulses Safety Switch
DA	-	Double Acting
FYP	-	Final Year Project
HMI	-	Human-Machine Interface
LS	-	Limit Switch
MIS	-	Management Information System
OOPCS	-	OMRON OPC Server
OPC	-	OLE for Process Control
OLE.DB	-	Object-Linking and Embedding for Database
OLE	-	Object-Linking and Embedding
PLC	-	Programmable Logic Controller
RLL	-	Relay Ladder Diagram
RA	-	Rotary Actuator
SCADA	-	Supervisory Control and Data Acquisition
SS	-	State-by-State
TT	-	Truth Table
VDIN	-	Virtual Digital Input

CHAPTER 1

INTRODUCTION

This chapter explains briefly about the project's architecture, objective, and coverage. Most of the contents inside this chapter may help reader to understand how this project works. In other words, this portion gives a clear picture of what has happened through out the project and how this project is going to be conducted.

1.1 Background of Study

This study is about to implement Grafcet programming language into a simple robotic arm operation. As a powerful industrial programming tool, Grafcet has been developed to enhance the capabilities of machine operation beyond its limits. It provides users a very comprehensive and flexible method of modeling the Grafcet by the synchronous data-flow language.

The Grafcet is a specialized computer language in which the control-command is needed to preprogram. Control-command consists of two subsystems that consist of the operative part or controlled part (OP) and controlling part (CP). All the physical equipments such as motors, actuators, valves and so on are part of the OP. The OP is controlled by CP which computes the condition of the OP with information gained from sensors and executes operation under preprogrammed command [1].

The inputs-outputs (inputs and outputs) of a PLC can be controlled via an interface called SCADA Interface. SCADA stands for Supervisory Control and Data Acquisition. SCADA Interface System is a common process automation system which is used to gather data from sensors and instruments located at remote site and to transmit and display this data at a central site for either control or monitoring purposes [8].

The integrated features allow users to communicate with the hardware Inputs-outputs via a protocol. Most of SCADA Packages supports all the Open Standards which are:

- OLE (Object-Linking and Embedding)
- OPC (OLE for Process Control – server and client)
- OLE.DB (OLE for Database connectivity)
- Server Side VB Scripting
- Server Side JAVA Scripting
- Wide Area Transfer for real time data – WWW

1.2 Problem Statement

In this project, it is required to implement a general monitoring and control process of a Pick-and-Place robot via Grafset programming language. Strategic modifications are expected to ensure the quality and efficiency of the project. In FYP 1 (Final Year Project 1), the project is mainly focusing on the programming of Grafset. It is all about generating the code to do the operation of pick and place and some additional control features. During FYP 2 (Final Year Project 2), an interface is designed. Briefly, the interface is able to communicate with the PLC in a very comprehensive and reliable way.

1.2.1 Programming Grafcet

It is convenient to introduce a simple way of generating the Grafcet program. These include:

- Timing Diagrams
- Boolean Expression
- Truth Table

The program generated is based on an operation of a robotic arm which can do picking and placing operation. This Pick-and-Place Robotic Arm is widely used in automation industry for such operation. The robot is equipped with various actuators located at different angle and position. The actuators involve in this operation only accept digital input and output. 5 electro-pneumatic actuators are integrated in place and assembled in different angles and position.

1.2.2 Supervisory Control and Data Acquisition (SCADA) Interface

To make the communication between human and machine more reliable, an intelligent interface must be introduced. The interface is used to facilitate the end user to do the process control. Operator may also gather data from the PLC by observing the report and trending that are provided by this interface.

In this project, inputs-outputs of the robotic arm are fully controlled. Student is expected to generate an interface that can trace data from the PLC, manipulates and represents every bit of the inputs-outputs into graphical interface. By doing this, the whole operation can be graphically presented for better operating purposes.

1.3 Objectives

Author is expected:

- To design and program a PLC to perform the picking and placing job of robotic arm.
- To have exposure to the environment of PLC and its application in wider scope of industries.
- To implement the GRAFCET programming language and investigate its advantages.

1.4 Scope of Studies

Estimating the project duration need to be done beforehand by simply investigating the scope that must be covered. It cannot be too wide and too narrow. The scope must be defined wisely and considering time constraint that is specified.

1.4.1 Understanding Grafcet Architecture and Program the Grafcet Code

There are many other languages existed and viable to do the programming of PLC, but the GRAFCET promises better time consuming, easily debugged, and simplicity in understanding the program's structure. Robotic Arm is programmed via Grafcet language.

1.4.2 Designing a SCADA Interface

All data captured from the PLC can be supervised and manipulated. Thus, SCADA Interface is introduced. It helps operator to control the system and deliver the information regarding the conditions and activities occur inside of PLC or at remote site.

CHAPTER 2

LITERATURE REVIEW

2.1 Theories of Grafcet

Grafcet is a standardized graphical language [2] and a modeling tool that can be applied to describe the behaviors of discrete event systems (dynamic systems that evolve according to the asynchronous occurrence of events). It was primarily developed for Programmable Logic Controllers (PLC), which are specialized computers used in industrial automation [3]. Grafcet program is a directed-graph composed of steps and transitions.

Nowadays, Programmable Logic Controller (PLC) can handle several hundreds of inputs-outputs, communication link between networks, operations of regulatory and continuous and so on in such a way that they are transparent and clear for the machines' user. In programming the PLC, there are certain logic control rules that need to be specified. There are two kinds of models. The first model is related to the typical models of asynchronous sequential circuits, state diagrams, or state tables. The second model corresponds to the representation as a possible implementation such as Relay Ladder Diagram (RLL) which is the most famous language that is specifically implementing the representation of relay systems.

The need of an intelligent language that can cope up with a large number of complex operations became very apparent in early 1970's. In the other words, it was required to represent large logic controllers with many inputs and outputs. Detail study indicated that, some aspects are needed to be looked into in dealing with large number of inputs-outputs. These are as follows:

Aspect 1:

Describe the sequence of states of a discreet-event system that may contain a very large number of states.

Aspect 2:

Simplicity and easy understanding is a must because some systems may be worked independently.

Aspect 3:

Typically, given a state of the system, only a few inputs affect the state, and only a few outputs may change. Thus, only the behavior relating to these input changes need to be described.

Aspect 4:

Know what has happened to the inputs-outputs of a logic controller, i.e., the changes in input may changed the corresponding output.

The Grafcet comes into play as it complies with all the aspects as mentioned above. It is a graphical tool which allows modeling of sequential behavior and concurrency (Aspects 1 and 2). The Grafcet model was defined by a French group [4]. It has been briefly defined in [5] and then detailed in [6].

2.1.1 Grafcet Concept and Condition

The behavior of a logic controller originates from its programming environment and depends on two types of data that is the process to be controlled or operator and other system or better known as the 'conditions and events'. An event occurs as the preset condition is true (expressed as a Boolean variable) or met.

2.1.2 Basic Notions

The Grafcet is a graph having two types of nodes, i.e., steps and transitions. Each step is followed by one transition. The directed arcs connect a step to a transition or vice versa.

Steps

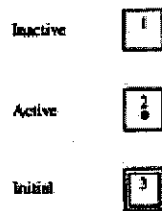


Fig. 2.1: Steps

A Grafcet contains at least one step and one transition as seen in Fig. 2.1. It is represented by a square and double squares for initial step and indicated as either active (see a token in each step) or inactive (no token) states.

Transitions

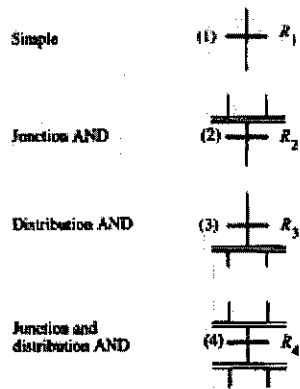


Fig. 2.2: Transitions

It is represented by a bar located between each step. Notice the vertical arc without arrow (see Fig. 2.2). It is running from top to bottom but different in some exclusive cases. In typical operation, the transition symbol is a bar, but somehow there is some other cases that double bars is used at preceding or/and following transition.

Directed Arcs

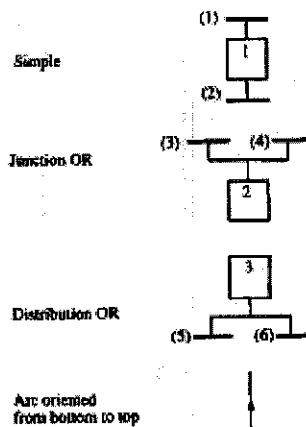


Fig. 2.3: Directed Arcs

From Fig. 2.3, directed arc connects either a step to a transition or transition to a step. Refer to arcs (3)-2 and (4)-2, when two or more directed links join the same step, they are grouped together. But when they leave the same step as shown, they shared a common departure point (refer to arcs 3-(5) and 3-(6)). It can be seen from the same figure, the arc oriented from bottom to top. It means the process or the token is passed by following the flow or the arrow indicated.

An active contain one and only one token, and an inactive step does not contain any token. The inputs of the logic controller are associated with the transition which means input and transition is the same things. The outputs are associated with the steps after the transition is fired. A transition is firable if both the following conditions are met [4]:

- All the steps preceding the transition are active (given that the transition is enabled).
- The receptivity of the transition is true.

In other words, it can be said that the step preceding and the transition must be true before the system proceeds with the next step. In logic form, the condition is denoted as in Table 2.1:

Table 2.1: Logic Form of Step and Transition

Step	Transition	Output
0	0	0
0	1	0
1	0	0
1	1	1

2.2 Theories of Pick-n-Place Robotic Arm

The powerful indexing capability of the OMRON PLC pneumatic rotating actuator is used to rotate a pick and place system into position so parts can be transferred from one operation to another. The system uses an OMRON CQM1H CPU21 programmable logic controller to select the position of the rotary table by activating digital inputs on the robot.

2.2.1 Brief Operation

A simple programmed function for the robot's movements was generated. See Fig. 2.5 for the movement's illustration.

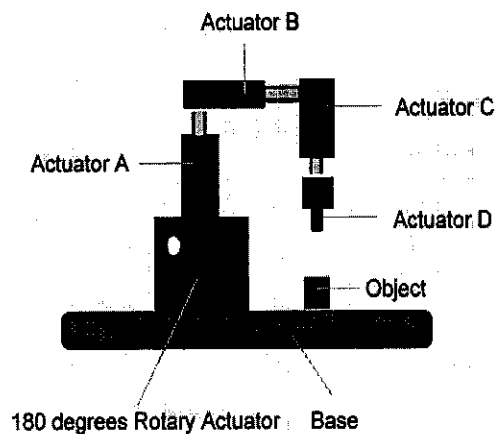


Fig. 2.4: Physical Layout of the Robot

The actuators of all parts of the robot are joined independently except the one with a clamper finger. There are total of 5 actuators involve and located at different angle and places as seen in Fig. 2.4.

Actuator A, B, and C extend and retract when they are electrically activated depending on the program structures. Actuator D acts as a clammer to the object where it has two dependant fingers to do the clamping. There is a rotary actuator with a capability of 180 degrees rotating angle located on the bottom of the arm. It is the main part of the robot to shift the object from one location to another location.

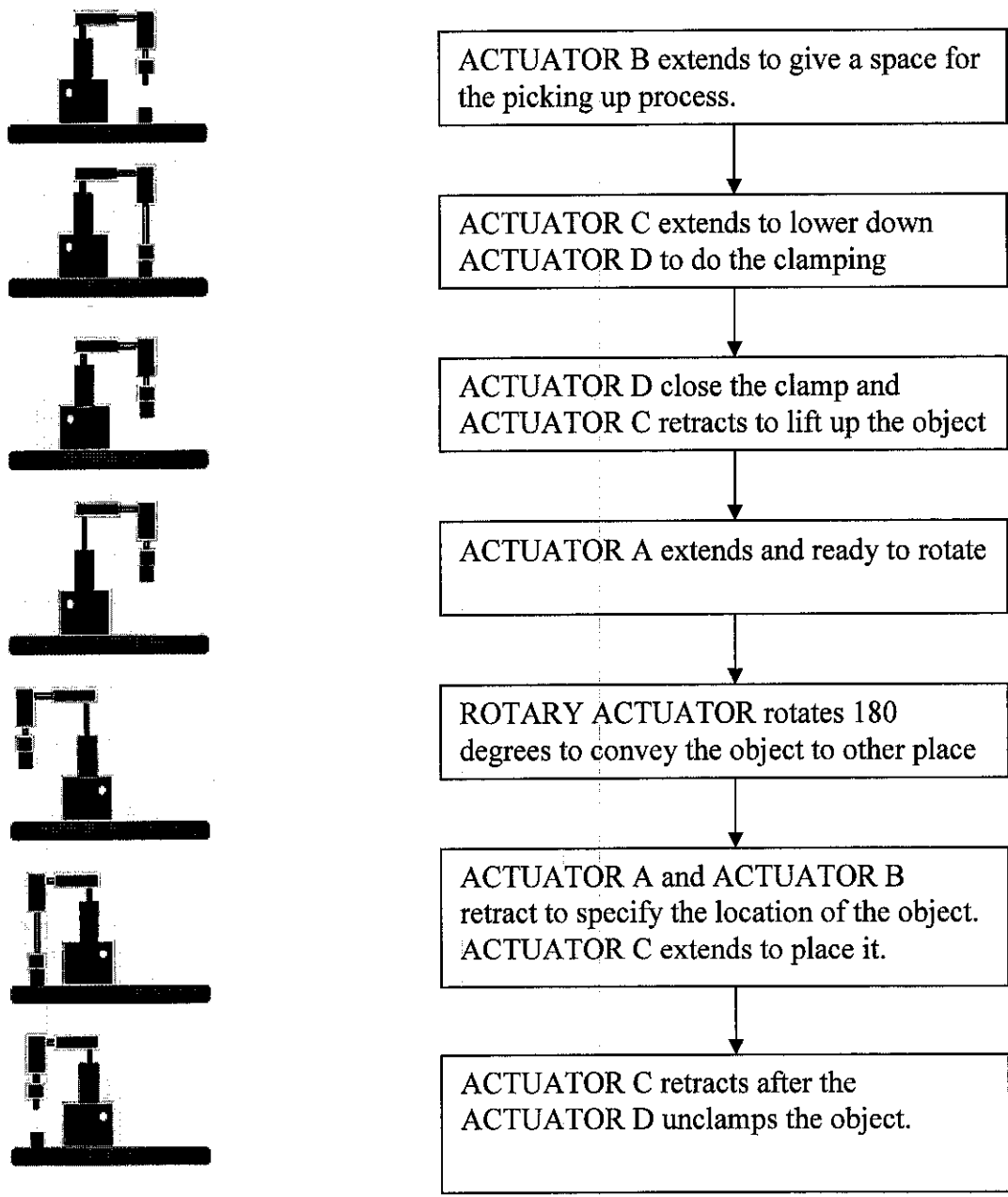


Fig. 2.5: Mechanism of the Robot

2.2.2 Control Objectives

In robotic automation, the use of pick and place robot is crucial to guarantee operation is running at the least time or delay, zero error, accident free, and minimized human intervention. Therefore, it needs to identify the strategies that can be used to achieve the control objectives.

Safety is the most important element that must be considered when identifying the control objectives. It lies on the top priority than others. Environmental protection, equipment protection, smooth operation, product quality, profit, and diagnosis are the other control objectives which commonly have been identified for industry. However, it depends on the decision of the company which is most prior to their industry.

This project has come out with a system which has considered some of the control objectives stated before.

- Safety: Dual Pulse Safety Switch implementation
- Equipment Protection: Intrusion Alert and Power trip

CHAPTER 3

METHODOLOGY

This work is devoted towards the development of a supervisory control and data acquisition of robotic arm operation via Grafset. For a better result, step by step procedures and preparations play important role. Thus, this portion generally briefs reader of how the project is conducted, the tools that are used, and other supporting elements to reach the objective.

3.1 Procedure Identification

Title for the FYP is provided by course coordinator during FYP 1 (First Semester). The title is reviewed by looking its scope of study and main objective of the project. The importance to select the best suited project is that, to know how far the project could be finished by the specified period of two semester study, the availability of tools and materials within the allocated budget and its potential for future modification.

Fig. 3.1 shows the flow chart representation of the steps undertaken throughout the two semesters (FYP 1 and FYP 2) that have evolved during this project study.

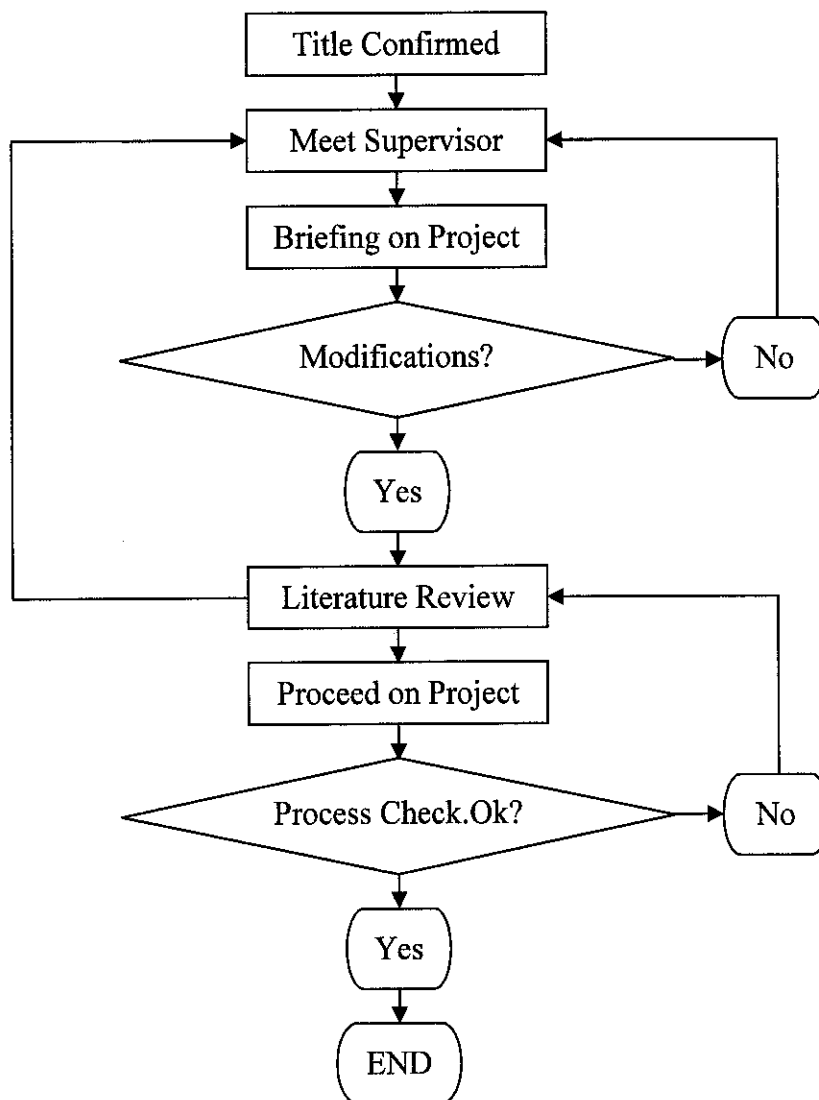


Fig 3.1: Flow Chart Representation of the Project Planning

3.2 Tools

This section discusses the tools (hardware and software) that have been utilized during the development of the supervisory control and data acquisition of the robotic arm.

3.2.1 Software

AUTOMGEN Licensed Software (Server)

This software is provided by UTP and available at Control Laboratory. To operate the software, it needs a licensed dongle attached to a computer. Communication with PLC and Grafset programming are established by using this software.

Adroit (Interface)

Adroit consisting of several different types of windows, makes up the client portion of the client-server architecture.

3.2.2 Hardware (APTS 502 Robotic Trainer Kit)

Actuators

Two types of actuators are used for this project. They are:

- Double-acting (DA) Cylinder- Used for expressing movement of the Robotic Arm.
 - Actuator Assignment: Actuator A, B, C, and D (see Fig. 2.4)
- Rotary Actuator (RA) – Used for expressing waist movement of the arm. It transports an object from one place to another.
 - Actuator Assignment: Actuator E (see Fig. 2.4)

Specifications:

DA Cylinder

- Cylinder Type: Double Acting
- Piston Type: Non-magnetic
- Return Friction: 20.00
- Extend Friction: 20.00
- Piston Diameter: 5.00 mm
- Rod Diameter: 2.00 mm
- Stroke Length: 8.00 cm

RA Actuator

- Degrees of Movement: 180 degrees
- Pressure Setting: 1

Directional Valves

All of the actuators use the same directional valve, which is 5/2 (14) valve. It is equipped with solenoids at both ends of the valve (energized and de-energized positions).

Specifications:

5/2 way (14) Valve

- No. of ports/positions: 5/2
- Controls Calculation (Left/Right): Maximum
- Selected Position: Standard
- Electrical Controls: Solenoid at both ends

Sensors

Limit Switch is used to replace the function of the sensors. It is assigned as LS# in the symbol list in Appendix 2.

Specifications:

LS1 & LS2

- Type of switch: Push Button
- Contact: Mechanical

3.3 Project Work

3.3.1 Project Architecture

Communication between the three main devices is established and visualized as seen in Fig. 3.2.

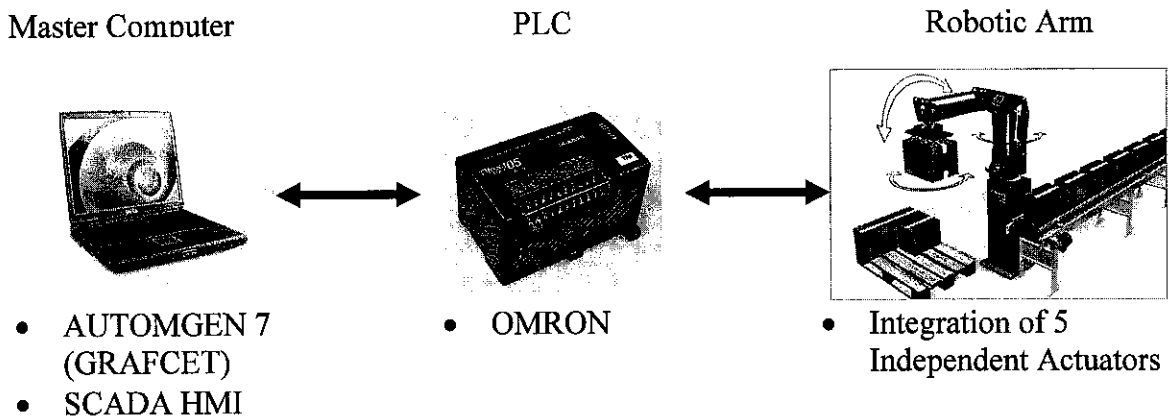


Fig. 3.2: General Overview of Project

3.3.2 Protocol

Before anything can start, the protocols involved in the system have to be recognized. From the very bottom of the system, there is a standard protocol implemented by the robotic arm PLC Device up to the SCADA level. The protocol allows communication between both levels.

PLC Protocol

There is a standard protocol used by the PLC to communicate with server at the master computer. From there, all the information of activities occurs inside the PLC are logged. These inputs-outputs data will then be manipulated by the software (AUTOMGEN) via Grafcet to monitor and/or control a process.

SCADA Protocol

Theoretically, connection between SCADA Interface and server can be established since there is a common protocol that the server could be complied with. In this project, OMRON OPC Server is used as a medium to communicate between the OMRON PLC and ADROIT SCADA. The OPC is a general protocol that most of PLCs used for making a communication to SCADA Interface.

3.3.3 PLC Server Connection

The PLC is connected to the server (computer) via RS232 communication cable. It is important to make sure that the server is configured to match the device's (target) specification. It is the primary way of making the server recognizes the device. Things that have to assured are:

- a. Device Name
- b. Device Type
 - CPU Type
- c. Network Type
 - Baud Rate
 - Communication Port
 - Data Bits
 - Parity
 - Stop Bits

AUTOMGEN is used as a server for the PLC and there is a feature of it allows user to program the Grafcet. Post-processors are actually software modules of AUTOMGEN that is used to translate pivot code files generated by the compiler into the executable files on a target [7]. The connection manual is referred to ensure all the settings are properly made according to the specification of the PLC.

3.3.4 Inputs-outputs Assignment

The inputs and outputs of the actuators are assigned directly to the internal address of OMRON PLC. They are directive (see Table 3.1) since all inputs-outputs assignment are same. User is able to define the inputs-outputs up to 16 Bit depending on the capability of the PLC.

Table 3.1: Inputs and Outputs (Inputs-outputs) Assignment

Actuators	Tag Name (AUTOMGEN)	Tag Name (SCADA)	Tag Name (OOPCS)	SCADA Virtual DIN
A	o0	SUPPORTARM	IR100.00	IR100.40
B	o1	TOPARM	IR100.01	IR100.41
C	o2	CLAMPERARM	IR100.02	IR100.42
D	o3	CLAMPER	IR100.03	IR100.43
E	o4	ROTATINGARM	IR100.04	IR100.44
LS1	i0	SWITCH01	IR000.00	IR000.00
LS2	i1	SWITCH02	IR000.01	IR000.01

All the inputs-outputs of the PLC are addressed to the corresponding symbol created inside the AUTOMGEN. The OOPCS (OMRON OPC Server) has different form of address assignment compared to AUTOMGEN. For input variable, the starting address should be written as IR followed by preset data address from the PLC. This form of assigning address is supported by Adroit.

Virtual Digital Input (VDIN) Technique

SCADA VDIN is the addressing technique introduced by the author to represent an input address by controlling the internal relays of the PLC which are available within the device. The PLC has got only 16 bits digital input and 16 bits digital output where all the inputs-outputs are already assigned to the physical layer device (valves and actuators). The inputs of physical device cannot be controlled at the control room or SCADA interface, but they can be changed if the input is controlled by changing the bit of outside the range of 0 to 15 bits. Therefore, the VDIN has been assigned as shown in the last column of Table 3.1.

Let say ARM01 is assigned with the output address of IR100.01. Since this is a digital output address, operator can not changed it by simply toggling the digital output. This is because the output acts only as a status indicator. As a result, it needs another digital output to control the ARM01 by ORing it with its output address. The equation below shows the situation of virtual DIN. The condition is defined as if either IR100.01 or IR100.40 is ON, the ARM01 will be ON as visualized in Fig. 3.3.

$$\text{IR100.01 OR IR100.40} = \text{ARM01}$$

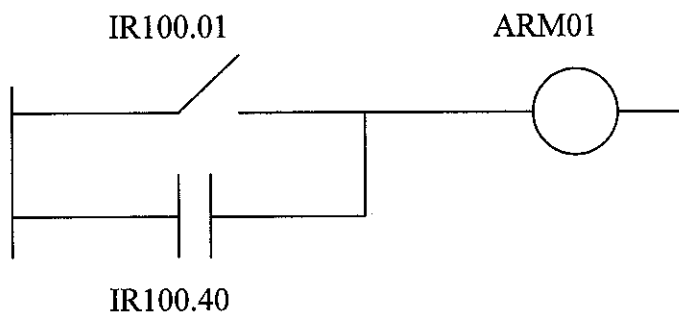


Fig 3.3: Relay Circuit of Representing the VDIN

3.3.5 Programming (Grafcet)

The process is verified by using State-by-State (SS) Transitions Method. See Fig. 3.4 for clearer picture. Each of the actuator has a specific address. All of them are digitally controlled which means the output is only 1 or 0. It is found that, by doing this method every single step taken can be easily traced and identified.

All the outputs are assigned to each bit as follows:

Table 3.2: Bit Representation

Actuators	A	B	C	D	E
Bit	0	1	0	0	0

And the Grafcet evolution is visualized in every state transition:

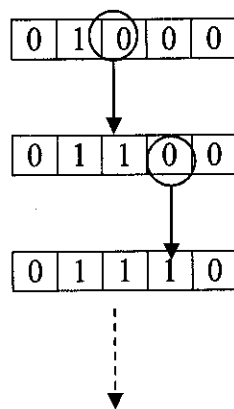


Fig. 3.4: State-by-State (SS) Transitions Method

The process is divided into 2 parts represent 2 locations. Port A for picking process and Port B for placing process. The arm is programmed to pick an arbitrarily object at point A and convey it along 180° rotation to point B and place it. After a certain period of time, t, it rotates back to the previous location (default position). So two states are already defined which are State A and State B.

The programming rung is now converged into two parts and it simplifies to proceed with the rest of the program. Port A is actually directed to the SET bit of Actuator E which is the Rotary Actuator and Port B is for its RESET. When Actuator E is SET, the arm will convey the clamped object to Port B. Then if Actuator E is RESET, the arm will change to position back to Port A.

The states from the SS are then been inserted into a table called Truth Table Representation as seen in Table 4.1. By doing this way, the transitions related to the whole process can be seen clearly.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Findings

Two significant approaches have been observed. The approaches define the evolution of Grafset program clearly. There are State-by-state Transition Method (SS) and Truth Table (TT) representation.

4.1.1 Programming Technique

SS Method

SS Method extracts the process into a simple bit representation. Each bit corresponds to the outputs of PLC (see Table 3.2). Fig. 4.1 below shows the evolution of bit transition by using SS method.

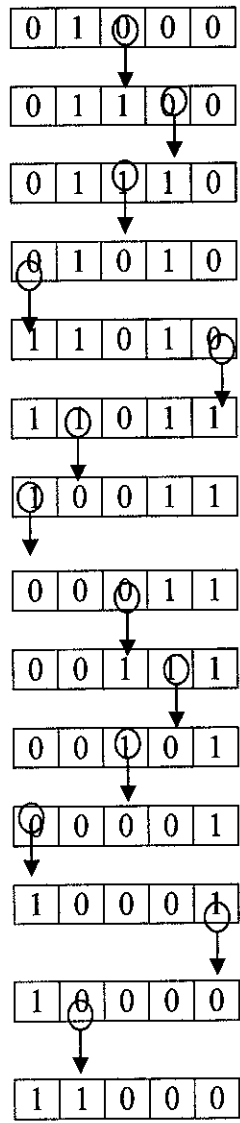


Fig. 4.1: Full Presentation of SS Method

TT Representation

Every row is related to every state at the SS Method. The circled bits are the occurrence of transition. Each transition has a delay as it changes from 0 to 1 or vice versa. So, there are 13 time delays. The bolded bits at the bottom are the states to default position. Table 4.1 shows how it works.

Table 4.1: Truth Table for Bit Transition

Actuators	A	B	C	D	E	Delays
PORT A	0	1	0	0	0	2 sec
	0	1	1	0	0	2 sec
	0	1	1	1	0	2 sec
	0	1	0	1	0	2 sec
	1	1	0	1	0	5 sec
Point of Rotation						
PORT B	1	1	0	1	1	2 sec
	1	0	0	1	1	2 sec
	0	0	0	1	1	2 sec
	0	0	1	1	1	2 sec
	0	0	1	0	1	2 sec
	0	0	0	0	1	2 sec
	1	0	0	0	1	5 sec
	1	0	0	0	0	2 sec
	1	1	0	0	0	2 sec
	0	1	0	0	0	2 sec

4.1.2 Program Code

Dual Pulse Safety Switch (DPSS)

To switch ON the machine, operator has to ensure the second switch is turned ON within some period of time, t . If the first switch is remained ON after the period of time, t , operator needs to turn ON the switch back to restart the time.

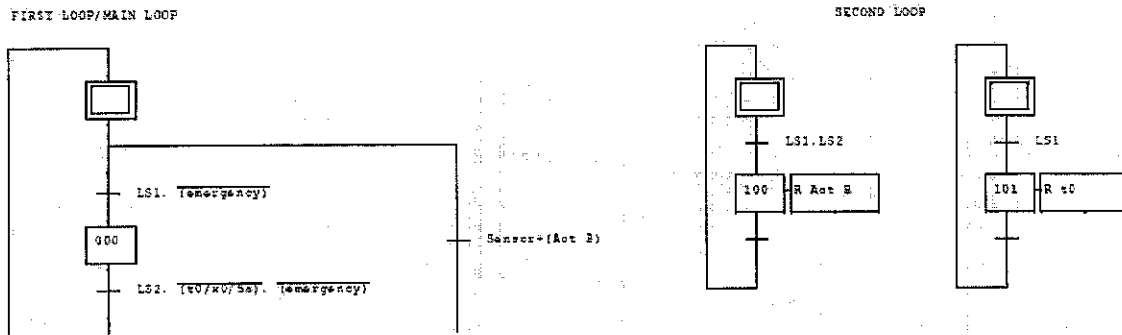


Fig. 4.2: Implementation of DPSS

LS1 and LS2 represent the first and second switch respectively of the interlock switches. By referring to the first loop of Fig. 4.2, timer, t_0 , and LS2 must be ON to set the transition or condition to 1 (neglecting the effect of “emergency” input). At the second loop, by repressing the LS1, the timer, t_0 , will be restarted (see the second loop).

Emergency Switch

In Grafcet, “emergency” input is introduced (see Fig 4.3).

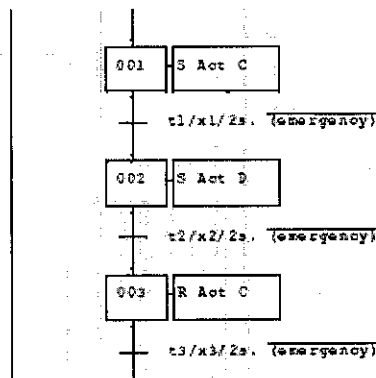


Fig. 4.3: Emergency Switch Code Generated

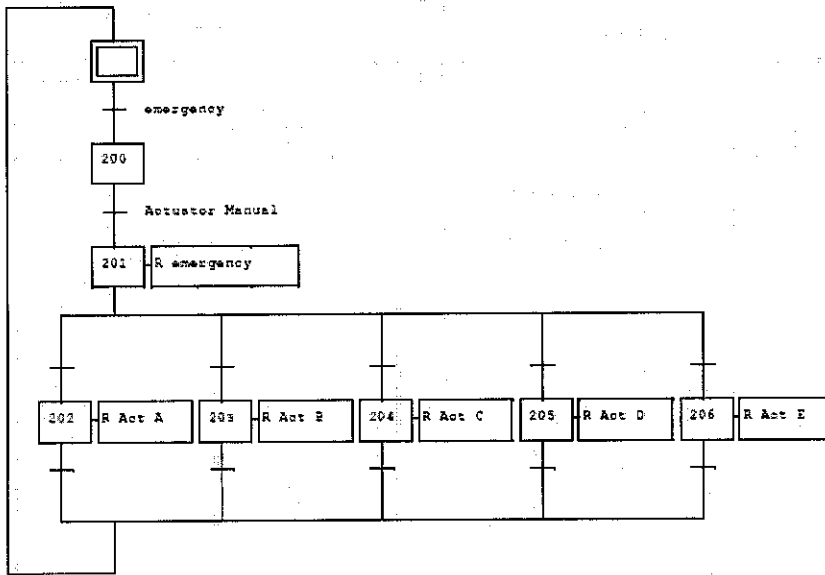


Fig. 4.4: Condition of Mode Operation and Emergency

Once the “emergency” input is ON, the operation mode is changed to manual and all the actuators are reset (zero state). Fig. 4.4 shows how the Grafcet code is generated for such application.

Sequence of Actuator

SET and RESET function are frequently used to every step. The SET is represented by ‘S’ and RESET by ‘R’ as shown in Fig. 4.5.

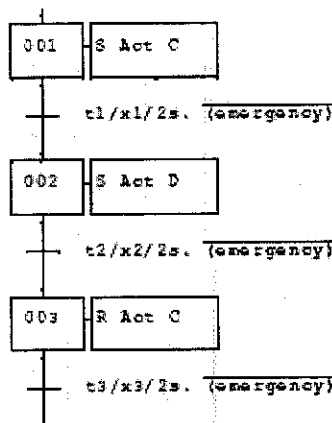


Fig. 4.5: SET and RESET function

Timer is assigned to each transition. When the time specified is elapsed, the following step will be activated. A delay of 5 seconds to every transition except the time delay for Actuator E is used frequently. It needs 10 seconds to complete a 180 degrees rotation.

Notice that **t2/x2/2s** is the syntax used to implement timer in Grafset. There are a few syntaxes available to be used.

t2/x2/2s

t2 - Refer to timer index
x2 - Refer to which step is affected by the timer
2s - The delay

4.1.3 Client-server

Client and Server interconnection must be established first. In order to do that, both server and client must be configured correctly.

Server (OMRON OPC Server)

The device's setting used for the system is defined first. Typical communication setting for OMRON:

- PLC Type: CQM1H CPU 52
- Port : COM1
- Baud Rate : 9600
- Format : 7,Even, 2, Stop
- Timeout: 500ms

Then, the tag for each input and output is defined as seen in Fig. 4.6.

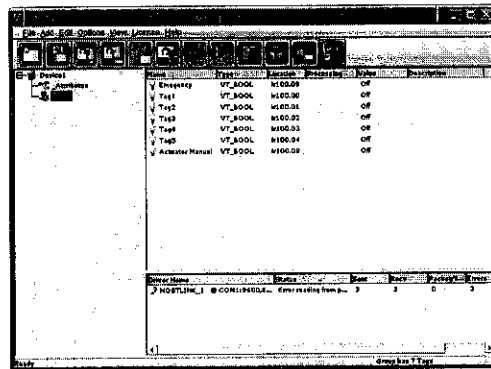


Fig. 4.6: Tagging Pane of OMRON OPC Server

Client (SCADA Interface)

Login Page

The Login page is designed with a security password according to the same security level of the operating system of the machine (this SCADA software only complied with Windows NT system base). Only authorized user is allowed to make any change of the system. See Fig. 4.7 for clear picture.

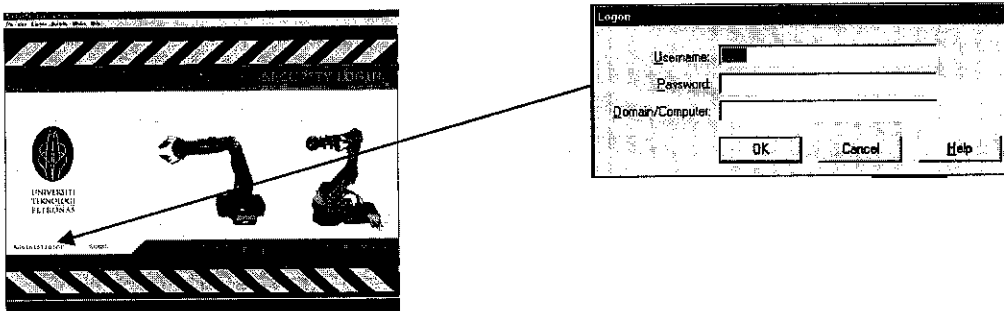


Fig. 4.7: Login Security Page

General Overview

After a successfully login, user will be directed to the general overview of the system. The page indicates the position of all the actuators involved in the system and the movement of the robotic arm in 3D presentation (refer to the middle pane of Fig. 4.8). It also provides user a simple trending report (see the bottom pane of Fig. 4.8)

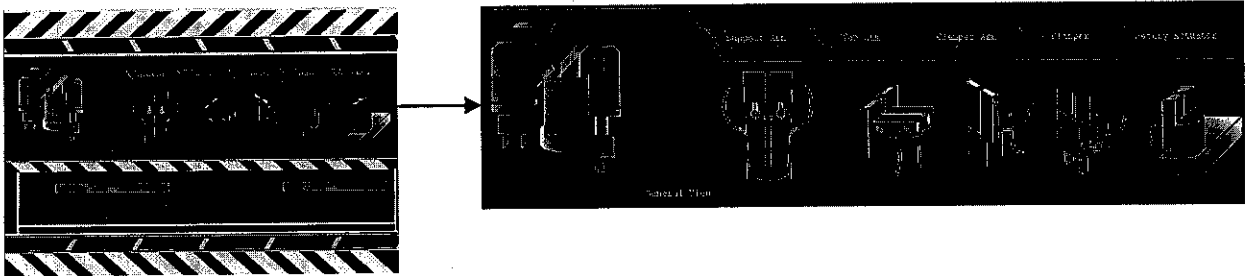


Fig 4.8 General Overview Window

Actuators

At the bottom pane of all the windows, it can be seen a few buttons which will link user to the windows of each actuator in this system. For example, when user clicks “SUPPORT ARM” button, the window of the correspondent link will appear as seen in Fig. 4.9.

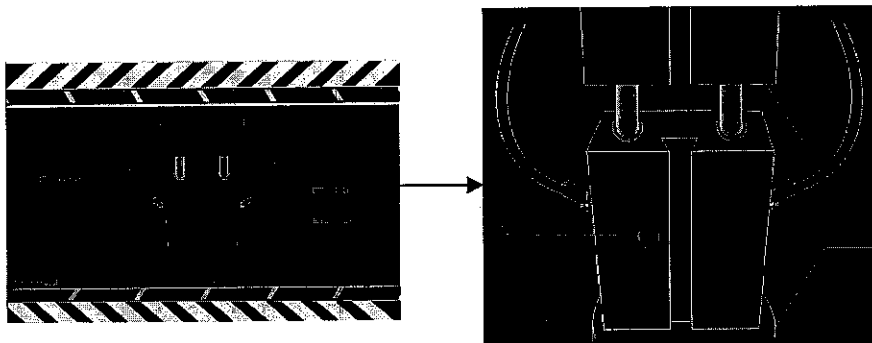


Fig. 4.9: Specific Instrument Window

So do the other buttons. User can simply check the specific position of the actuator. There are total 5 actuators' windows and each of them shows physical characteristic in 3D representation.

Status

To obtain all the data and status of all actuators, user can browse the link to “Status” located at the top pane of all windows (see Fig. 4.10 for status window).

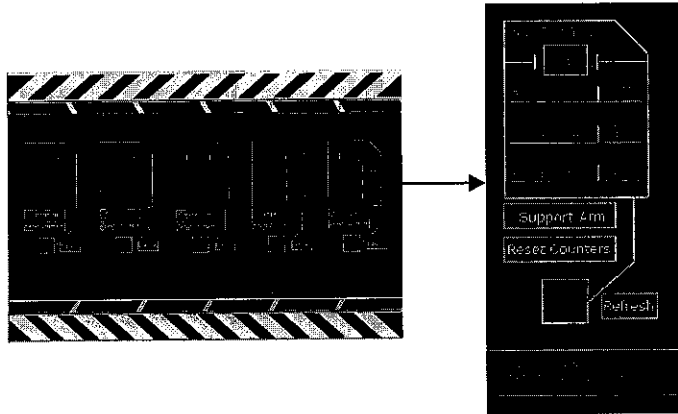


Fig. 4.10: Whole System Status Window

Reporting

There are two types of reporting implemented in this project which are alarm and trending. A total overview of all alarms, both Active and Alarm Log, are just a mouse click away for the operator. Right clicking on an active alarm in the module enables the operator to call up the Find function to immediately locate precisely where in the process the alarm originates (see Fig.4.11).

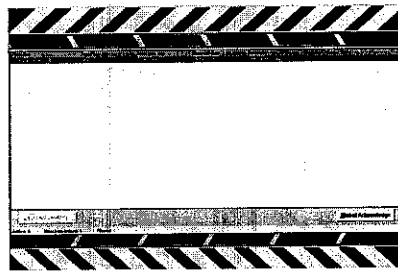


Fig. 4.11: Alarm Window

Trending allows user to specifically know the pattern of operation in real-time or even in the past. It is also known as a timing diagram but in the different characteristics. See both Fig. 4.12 and Fig. 4.13 for clarity.

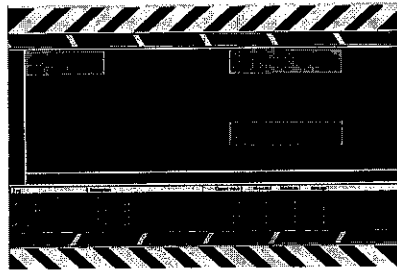


Fig. 4.12: Trending Window

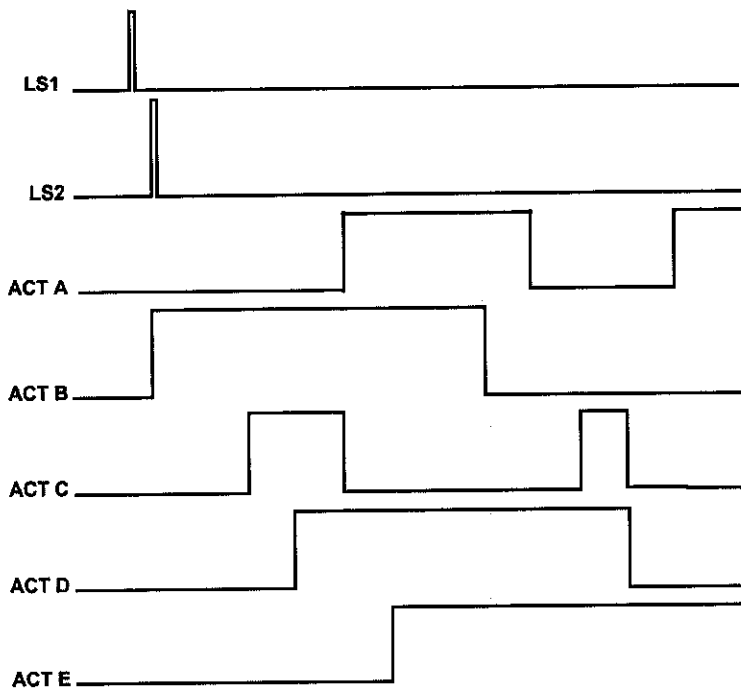


Fig. 4.13: Timing Diagram of Typical Operation

The timing diagram signifies the pattern or trending of the robotic arm process. If there is a fault occurs to the system operation, the trending window will show an irregular pattern. It is realized that sometimes, due to some technical failures, the pattern of the inputs-outputs is distracted and need a solution to troubleshoot it. As a result, a fault-finding solution was introduced. It was a graphical representation of the timing diagram. It states a standard pattern of normal operation and detects any abnormalities of the operation by simply observed the pattern.

4.1.4 Control System

The system has two modes of operation which are Manual Mode and Automatic Mode. Briefly, the manual mode is defined as an opened loop operation where the actuators are controlled by human supervision whereas the automatic mode is for the closed loop operation.

- **Manual Control:** Operator can observe the status of all equipments and may introduce changes to compensate. Adjustment made depends upon the person.
- **Automatic/Continuous Control:** This is a modulating control of digital inputs and outputs. No human intervention but the operation can be break whenever the Manual mode is activated.

4.2 Discussion

4.2.1 Programming Technique

The robot is programmed initially by using simple function of ladder diagram. The purpose of doing so is that, the box of functions used in ladder diagram to be implemented yet in the Grafset can be clearly seen.

The OMRON PLC has its advance control functions. The Inner Boards allows simple positioning, multi-point high-speed counter inputs, absolute rotary encoder inputs, analogue I/O, analogue settings, and serial communications for connection to standard serial devices. Connections can be easily made to general-purpose machine components and dedicated controllers. To connect with the server, CX-Programmer is used. The code generated by this software is ladder diagram.

In ladder diagram, each rung corresponding to Boolean equation is associated with the input and output variable. It may be convenient to do the ladder diagram after the Boolean equation is obtained. But it needs to do some tedious works to obtain the equation. Moreover, the sequential nature of the behavior is not apparent as compared to Grafset.

Grafset code has been developed for this robotic system. There is no specific method for programming Grafset. Therefore, the approaches by using SS Transition Method and TT Representation are introduced by the author.

State-by-State (SS) transitions Method

As what have discussed earlier, there are two parts (A and B) involve to complete the process of picking and placing. The method of SS transitions provides an easy way to understand and clarify how things work.

Grafcet is also known as a sequential function charts [1] where it can manage any complex sequences by assigning the input and output to its graphical box. A sequence also known as a cyclic control-command system has been adapted and implemented by using the State-by-State (SS) transitions Method as visualized in Fig. 4.14. This is a much simpler way as compared to other method that has been used by historically assembler-like languages, ladder diagram languages, Petri Nets, SA-RT, synchronous language, state charts and so on.

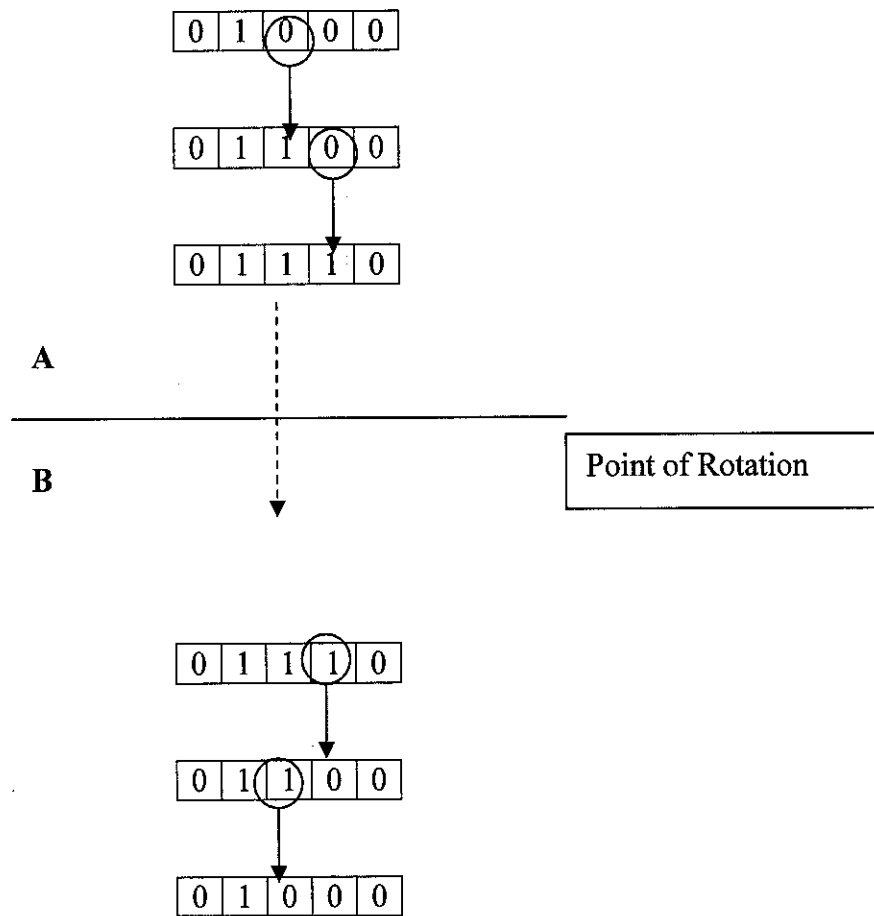


Fig. 4.14: Architecture of Operation (SS Frame Transition)

The transition of bit as referred to the SS method has an advantage. Obviously can be seen, there are 13 transitions to complete the process and each transition represents a specific condition that the system has to meet before it proceeds to another step. For each

and every transition, programmer can state any condition like a timer, divergence AND, divergence OR, and so on.

In this project, a timer condition is used for each of the transition. Hence, there would be 13 timers involve. Denote the total number of input transition, **I**, so **I**, is proportional to the condition, **C** (timer).

Truth Table Representation

The method is then converted into a truth table consisting of 5 independent outputs. A structured truth table is needed to show a better presentation of bit flow through out the process. In fact, it provides a simple representation as compared to the frame of SS transition (see Table 4.1).

From the truth table, the eclipsed elements are representing the transition bit. There are 12 transitions inclusive of a bit transition of the arm to return to default position, A. Each of the elements has a specific delay, *t*, to transit from 1-0 and vice versa. The delay is constant to all transitions.

By knowing the amount of bit transitions, the Grafcet sequences code can be generated easily. Denote the total bit transition is *n*, so the number of input transition, **I**, inside the Grafcet is given by:

$$\mathbf{I = n+1}$$

The ONE is added since the transitions of state B back to state A (the arm rotates back to its default position) is considered. As a result, there will be:

$$\begin{aligned} \mathbf{I} &= \mathbf{12 + 1} \\ &= \mathbf{13 \text{ input transitions}} \end{aligned}$$

4.2.2 Programming Code

Safety Environment (Dual Pulse Safety Switch)

The system is equipped with a safety control switch. In industry, the implementation of interlock switch is very popular for safety precaution. Machine will start operating when both of the interlock switches are turned ON. Concerning a situation where only the first switch is turned ON at one time. The machine will automatically operate whenever the second switch is accidentally turned ON. What will happen, if there is an intruder or maybe another operator is standing within the danger parameter and suddenly the machine moves?

To avoid that possibility from happen, an improved system of interlock switches is introduced. A Dual Pulse Safety Switch, founded by author, functions quite similar to the interlock switches except, it has a timer which can reset the state of first switch to the initial state (commonly is in OFF state).

Equipment Protection (Emergency Switch)

When emergency happens, whole system will be stopped automatically and the operation mode is turned back to manual mode. During that time, all actuators are reset and can be controlled manually. This method somehow will help in protecting the actuators by stopping the operation simultaneously.

Sequential Actuator

All the actuators is set to be sequentially ON by transition of time, t . After time, t is elapsed, the transition will ON. So, the step will be activated as the condition is met.

4.2.3 Client-Server

Server: PLC Communication

A communication between Adroit and PLC is established through a general protocol that allows both layers talk to each other. As a result, an INGEAR OMRON OPC Driver is found during the researching period. It is able to trace, gather, and pump data from and to the PLC and interact with Adroit without any failure. Thus, to ensure the communication between both of the layers is going well, a standard setting for OMRON is needed to be defined as follows:

Client: SCADA Interface

SCADA HMI interfaces the data traced from the PLC to be monitored, controlled, and manipulated in more user-friendly environment. It can ease users in understanding well and having a simple control method.

In this project, the work has been focusing on:

- Creating a security login page used for authorizing certain personals to operate the HMI.
- Creating an interface to do the control process.
- Alarm page used for alerting operator on any presence of alarm.
- Trending used for viewing a real-time pattern of operation.

SCADA stands for Supervisory Control and Data Acquisition. As the name indicates, it is not a full control system, but rather focuses on the supervisory level. As such, it is a purely software package that is positioned on top of hardware to which it is interfaced, in general via Programmable Logic Controllers (PLCs), or other commercial hardware modules. General overview of the system is shown clearly in figure 13.

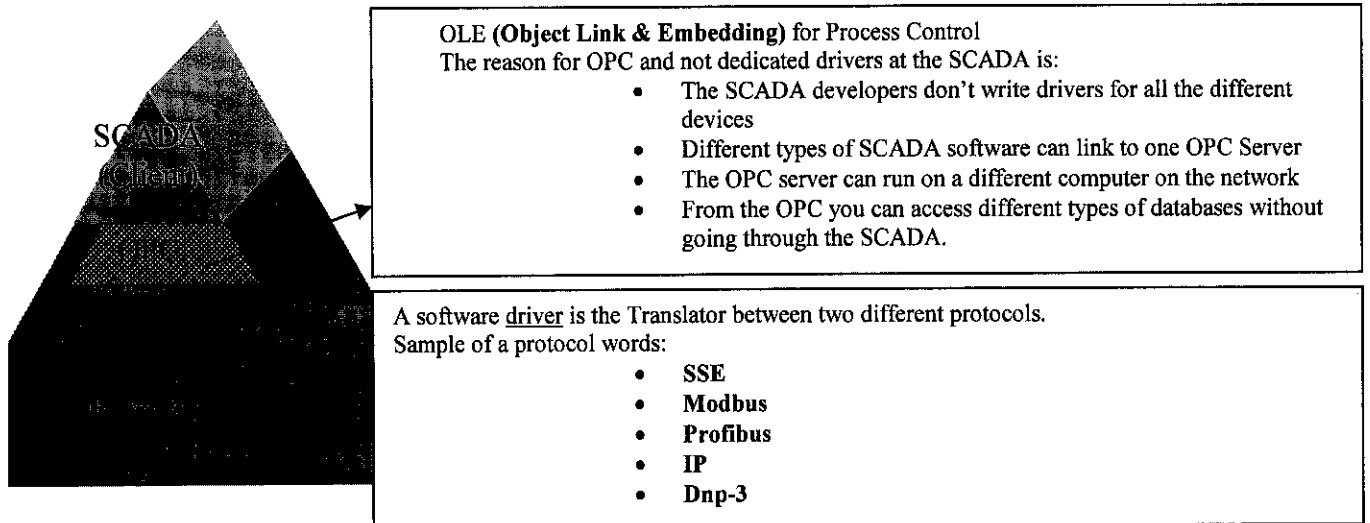


Fig. 4.15: SCADA System Architecture

SCADA systems are used not only in industrial processes: e.g. steel making, power generation (conventional and nuclear) and distribution, chemistry, but also in some experimental facilities such as nuclear fusion.

SCADA software is used to display and manipulate the data from and to the PLC, to a level where the data will make sense, to the persons using the system.

Different types of SCADA:

Adroit (Currently Used), Factory Link, Fix De-Max, In Touch, Cytec

This software has its own capabilities of handling a lot of IOs by using a common protocol where many PLCs as well as RTUs are able to communicate with. It is the OPC Protocol.

4.2.4 Control System

The system comprises two modes of operation which are open loop and closed loop operation.

Closed Loop

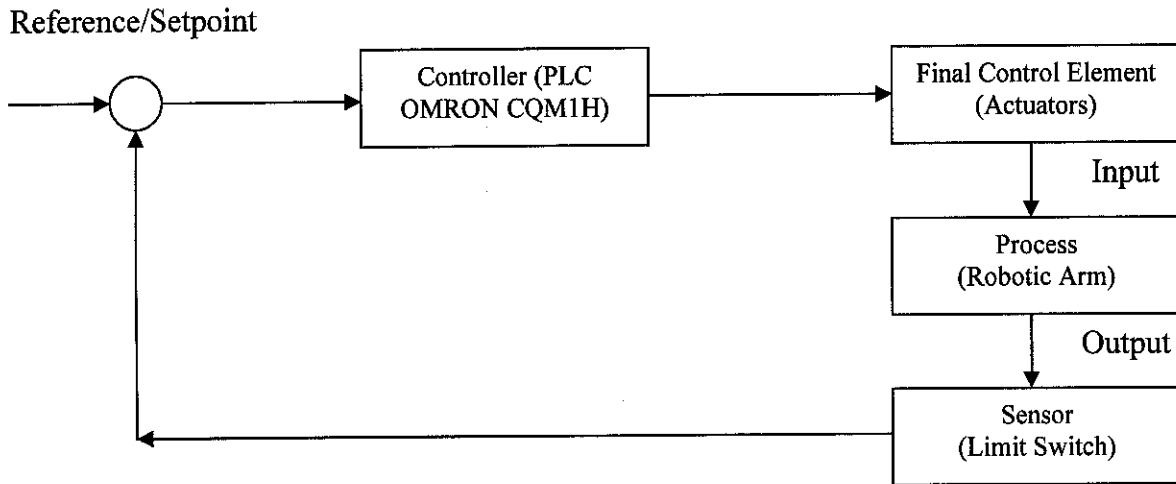


Fig. 4.16: Block Diagram of Closed Loop Operation

The block diagram (see Fig. 4.16) shows the relationship between input and output of the system. The presence of programmable logic controller (PLC) is to ensure the process is in feedback control. It is also known as an Automatic Mode. Only when sensor (in this project limit switch is used) senses an object, it will tell the controller that there is a change in input. Controller requires input of both sensor and desired reference of the variable to activate the final control element.

Final control element (actuator) is a device that exerts direct influence on the process. It accepts input from the controller which is then transformed into proportional operation on the process.

Open Loop



Fig. 4.17: Block Diagram of Open Loop Operation

Fig. 4.17 clarifies the flow of open loop operation that constitutes the Manual Mode operation. There is no controller presents at the system to do the automatic looping. This operation allows an authorized user to manually operate the robot.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

This project is mainly about a design strategy for movements of a robotic arm controlled via a powerful programming language called Grafset and generating an intelligent SCADA Interface. Grafset stresses out concurrencies which are how easy programs to be understood and its simplicity towards how the program is structured and generated. Programming by using Ladder diagram is very popular but have several setbacks for example a tedious way in obtaining the Boolean expression and may become difficult to understand when system is made of several parts (completely or partially independent). Hence, Grafset which gives more flexibility in terms of how its program is generated and troubleshoot is worth to be studied and explored. These have provide very useful piece of knowledge and information that truly applicable for the future work as an engineer.

The project is well managed and produces promising results. The main contributions of this work are:

- **Methods for Grafset Programming**
 - State-by-state transition method for process flow
 - Truth table representation for process modeling

- **System Architecture Representation of Multiple Layers**
 - The whole system is defined by a simple pyramid diagram as shown in Fig. 4.15. Almost all of the layers starting from the bottom up to the top level of the pyramid have been configured.

- System Security and Data Acquisition
 - The interface is designed with the operating system (OS) security. The Windows NT User Right policy manages the rights granted to groups and user account. A right authorizes a user to perform certain actions on the system.
 - Trending, alarm reporting and current status of the process are type of data collected from the PLC and displayed to the interface.

- Communication protocol and its compatibility
 - The adroit OMRON protocol driver provides serial connectivity using RS-232 to OMRON host link modules.
 - INGEAR OMRON OPC Server provides communication between OMRON PLC with any OPC SCADA Software.

SCADA Interface is used for interfacing the PLC and operator. Different SCADA software comes up with different features but having the main capability of monitoring and controlling data of certain process. The interface is designed by using software called Adroit originates from South Africa. The software is an example of what is known as a client-server architecture, or model. In this model, the “client”, which is typically the part of the application that interacts with the user, communicates with “server” which is the INGEAR OMRON OPC Server via an OPC Protocol.

5.2 Recommendations

The biggest portion of Supervisory Control and Data Acquisition (SCADA) system has been covered by author. It takes two semesters to complete the system within the scope of study. A smarter approach and more complicated problem are expected from the next batch to enhance this project to be more dynamic and gained better result.

The work presented in this report has contributed to an improved understanding of the procedure for the development of a supervisory control and data acquisition of a robotic arm via Grafset. Even though the approach offers some promising tools, however more work needs to be done on the way of programming the Grafset. For instance, the operation involve in this project has dealt only with discreet or digitally 1 and 0 state. For continuous process, the operation would be a bit more complicated as compared to the discreet operation. Thus, it is suggested to extend the scope of operation for future work.

The need of Management Information System (MIS) to be implemented into the SCADA System is apparently high. A set of tools must be introduced to facilitate the flow of information from the process management layer to the business management layer of an organization. The tools is able to retrieve data from server (OMRON OPC Server) belonging to more than one project, providing management with the ability to get an overall view of their operations. It will result in a profitable outcome and improve the process to be more productive.

REFERENCE

- [1] Journal by Philippe Le Parc, Dominique L'Her, Jean-Luc Scharbarg, and Lionel Marc'e, Grafcet Revisited with a Synchronous Data-Flow Language, IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, Vol. 29, No. 3, May 1999.
- [2] Object Management Group. Unified Modeling Language – UML Notation Guide Version 1.1. Ad/97-08-05.Framingham, Mass. November 1997.
- [3] Journal by Nathalie Gaertner and Bernard Thirion, Laboratory EEA, Group LSI, University of Haute, Grafcet: An Analysis Pattern for Event Driven Real-time.
- [4] Rene David, A Powerful Tool for Specification of Logic Controllers,IEEE Transaction on Control System Technology, Vol. 3 No. 3, September 1995
- [5] GRAFCET, "Normalisation de la representation du cahier des charges dun automatisme logique," *Final Report of AFCET Commission*, Aug. 1977, published in the *J. Automatique et Znfomtique Zndustrielles* (61-62), Nov.-Dec. 1977.
- [6] M. Blanchard, "Comprendre, maitriser et appliquer le Grafcet," Editions Cepadues, Toulouse, 1979.
- [7] Manual Reference of AUTOMGEN Post-processor, Release #4, 2004, www.irai.com
- [8] ADROIT 6 Manual References, South Africa.

APPENDICES

APPENDIX A: FULL PROGRAM CODE

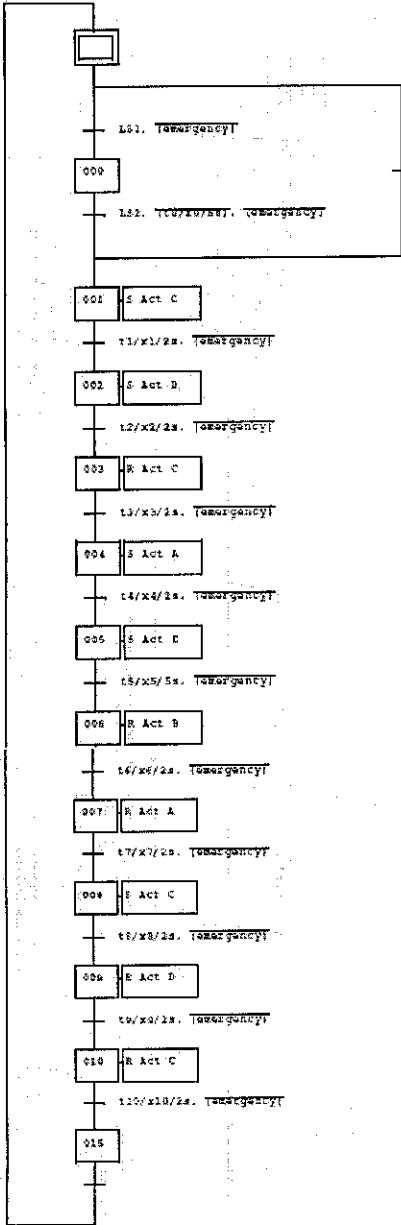
APPENDIX B: OMRON SYSMAC PLC ADDRESSES SUPPORTED BY ADROIT

APPENDIX C: PLC COMMUNICATIONS & GRAFCET

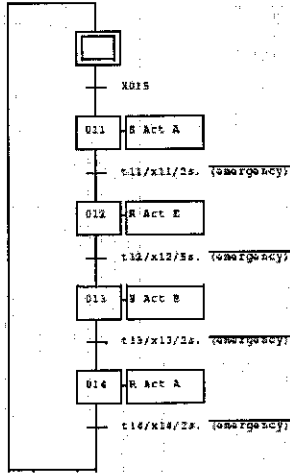
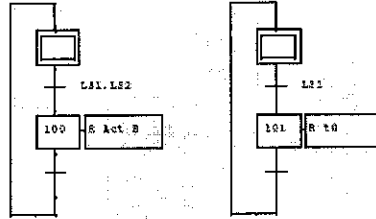
APPENDIX A

FULL PROGRAM CODE

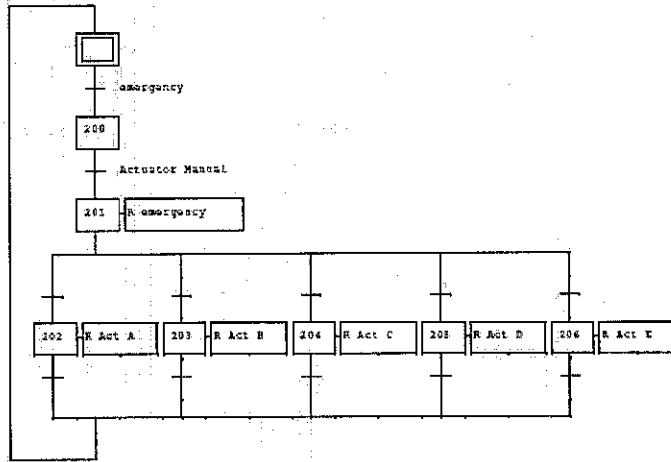
FIRST LOOP/MAIN LOOP



SECOND LOOP



FOURTH LOOP WITH MANUAL/AUTO MODE AND EMERGENCY SYSTEM APPROACH



APPENDIX B

OMRON SYSMAC PLC ADDRESSES SUPPORTED BY ADROIT

Device	Prefix	Bits	Min Range	Max Range	PLC Data Type	BOOL	INT	REAL	STRING
Internal Relay	IR	1*	000.00	8191.15	Bit	IR 000	IR 000	IR 0000	
		16	000	8191	BCD16		IR 000 B	IR 0000 B	
		16	000	8191	Unsigned Binary (Word)		IR 000 W	IR 0000 W	
		16	000	8191	Signed Binary (Integer)		IR 000 I	IR 0000 I	
		32	000	8190	Unsigned Binary (Double Word)		IR 000 D	IR 0000 D	
		32	00	8190	BCD32		IR 000 N	IR 000 N	
		32	000	4095	Signed Binary (Long)		IR 000 L	IR 0000 L	
Holding Relay	HR	1*	00.00	1599.15	Bit	HR 000	HR 000	HR 0000	
		16	00	1599	BCD16		HR 000 B	HR 00 B	
		16	00	1599	Unsigned Binary (Word)		HR 000 W	HR 00 W	
		16	00	1599	Signed Binary (Integer)		HR 000 I	HR 0000 I	
		32	00	1598	Unsigned Binary (Double Word)		HR 000 D	HR 000 D	
		32	00	1598	BCD32		HR 000 N	HR 000 N	
		32	00	1598	Signed Binary (Long)		HR 000 L	HR 000 L	
Auxiliary Relay	AR	1*	00.00	959.15	Bit	AR 000	AR 000	AR 000	
		16	00	959	BCD16		AR 000 B	AR 000 B	
		16	00	959	Unsigned Binary (Word)		AR 000 W	AR 000 W	
		16	00	959	Signed Binary (Integer)		AR 000 I	AR 000 I	
		32	00	958	Unsigned Binary (Double Word)		AR 000 D	AR 000 D	
		32	00	958	BCD32		AR 000 N	AR 000 N	
		32	00	958	Signed Binary (Long)		AR 000 L	AR 000 L	
Link Relay	LR	1*	00.00	1023.15	Bit	LR 000	LR 000	LR 000	
		16	00	1023	BCD16		LR 000 B	LR 000 B	
		16	00	1023	Unsigned Binary (Word)		LR 000 W	LR 000 W	
		16	00	1023	Signed Binary (Integer)		LR 000 I	LR 000 I	
		32	00	1022	Unsigned Binary (Double Word)		LR 000 D	LR 000 D	
		32	00	1022	Signed Binary (Long)		LR 000 L	LR 000 L	
		Timer/ counters (preset)	TCP	16	000	2047	BCD16		TCP 000 B
16	000			2047	Unsigned Binary (Word)		TCP 000 W		
16	000			2047	Signed Binary (Integer)		TCP 000 I		
Timer/ counters (current)	TCC	16	000	2047	BCD16		TCC 000 B		
		16	000	2047	Unsigned Binary (Word)		TCC 000 W		
		16	000	2047	Signed Binary (Integer)		TCC 000 I		
Data Memory	DM	1	0000.00	9999.15	Bit	DM 000			
		16	0000	9999	BCD16		DM 000 B	DM 000 B	
		16	0000	9999	Unsigned Binary (Word)		DM 000 W	DM 000 W	
		16	0000	9999	Signed Binary (Integer)		DM 000 I	DM 000 I	
		32	0000	9998	Unsigned Binary (Double Word)		DM 000 D	DM 000 D	
		32	0000	9998	Signed Binary (Long)		DM 000 L	DM 000 L	
		32	0000	9998	Float			DM 000 F	
32	0000	9998	IEEE754Float			DM 000 R			

(*) Bit writes are done using a Read-Mask-Write algorithm to achieve an effective bit write. The Read-Mask-Write algorithm reads a word from PLC memory, masks in the bit to be written and immediately writes the entire word to PLC memory. Therefore if the PLC program modifies the bits between the Adroit drivers read and subsequent write operations, the changes made by the PLC program will be lost.

APPENDIX C

AUTOMGEN 7 PLC COMMUNICATIONS & GRAFCET

INSTALLATION

1. If you are installing from the AUTOMGEN CD-ROM, put the CD in your CD-ROM drive. Installation is automatically launched.

2. If this does not occur, launch the « Setup.exe » executable located in the CD-ROM root directory. The CD-ROM contains AUTOMGEN7, ACROBAT READER (for access to on-line documentation) CROSSROADS (a 3D conversion utility program) and DIRECTX 8 (for managing 3D display).

3. If you are installing it from files downloaded from Internet, launch the execution from the downloaded executables. The Internet site can also be used for downloading ACROBAT READER, CROSSROADS and DIRECTX8 modules.

Network installation

AUTOMGEN can be installed in a network without any problems.

1. Execute the installation process on the «server» PC (make sure you have all the access rights at the time of installation).

2. To launch AUTOMGEN on client PC's, create a shortcut to the « autom7.exe » executable in the AUTOMGEN installation directory on the server PC.

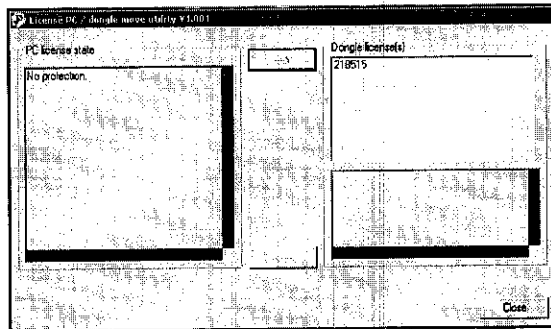
3. To make post-processors appear in the target tab on Client PC's, install the post-processors on client PC's then uninstall AUTOMGEN on client PC's (this is to create only lines in the « Target » windows).

LICENSE

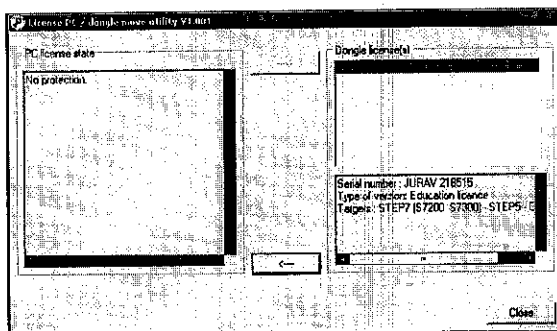
1. Click on the CD-ROM and locate the file "AKEY7MOV"



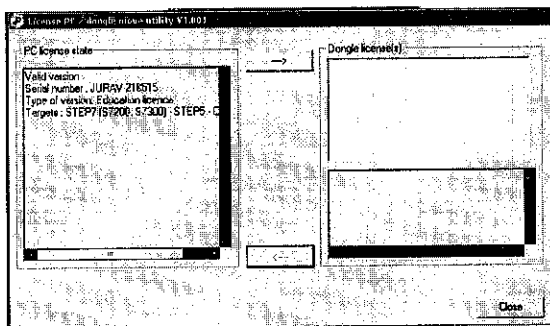
2. Copy the file and paste it into root directory, i.e., C:\Program Files\IRAI\AUTOMGEN7101
3. Plug in the dongle license into the PC parallel port.



4. Double click on "AKEY7MOV", click on the 'Dongle License(s)' column and highlight the license.



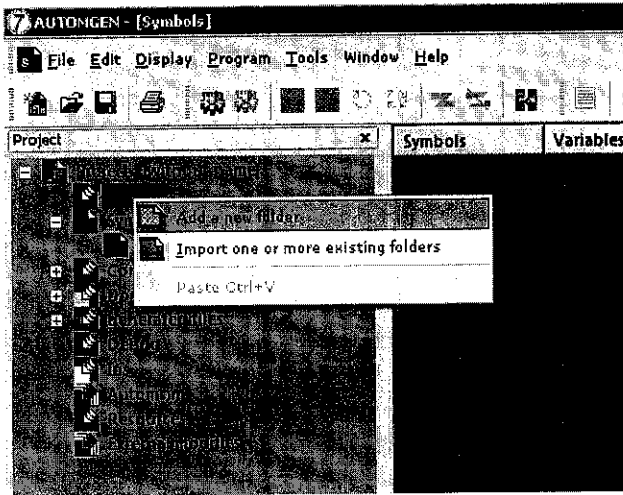
5. Then, click the arrow to transfer the license into the PC.



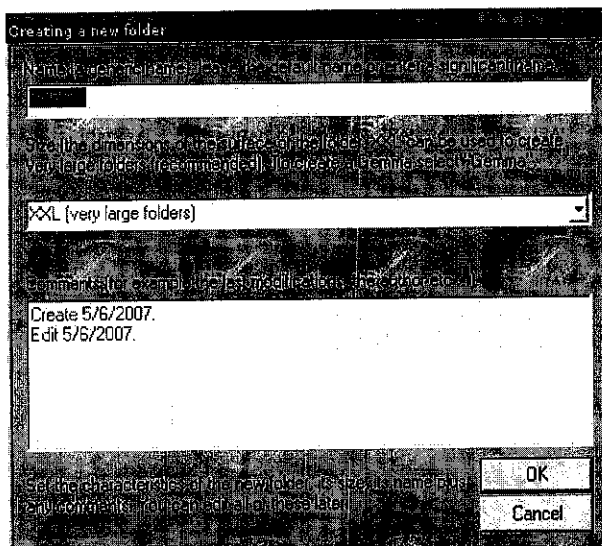
6. Before closing the AUTOMGEN, make sure the license has been transferred back into the dongle by clicking the opposite arrow.

DESIGNING

1. Right Click at the 'Folders'-Add New Folder



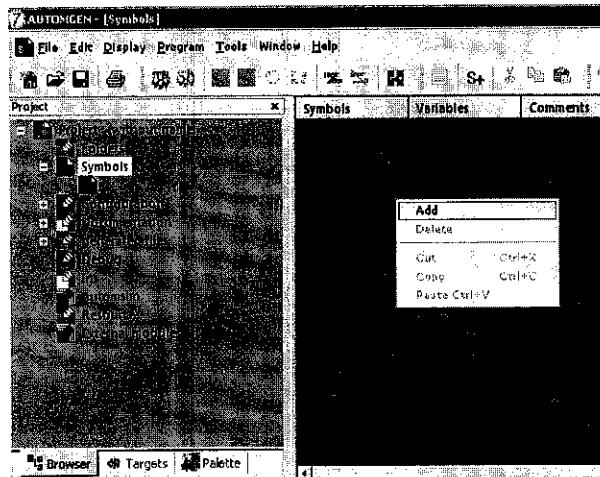
2. Name the folder for better indexing.



3. Creating a symbol table

The list of symbols provides the correspondence between « symbol » names and variable names. A project may only have one symbol table.

With the right side of the mouse click on the « Symbols» element on the browser and select « Create a symbol table » from the menu.



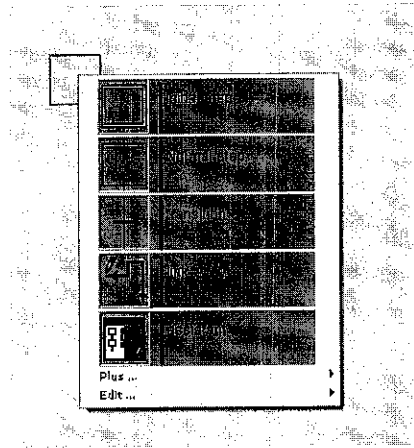
Symbol Properties

- *Name the symbol.*
- *Associated variable slot defines only the supported address of corresponding protocol can be written here.*

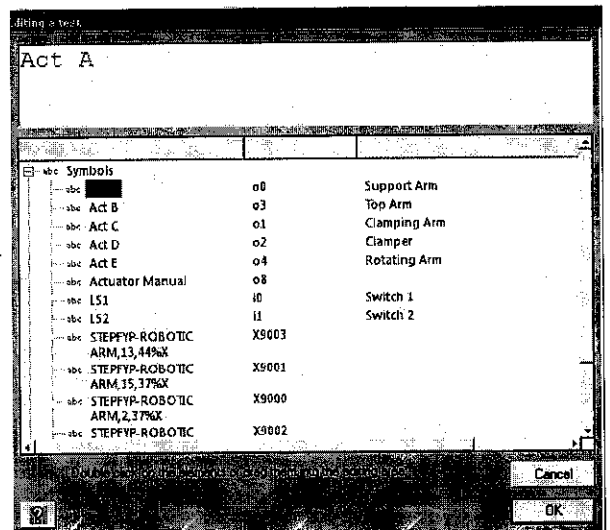
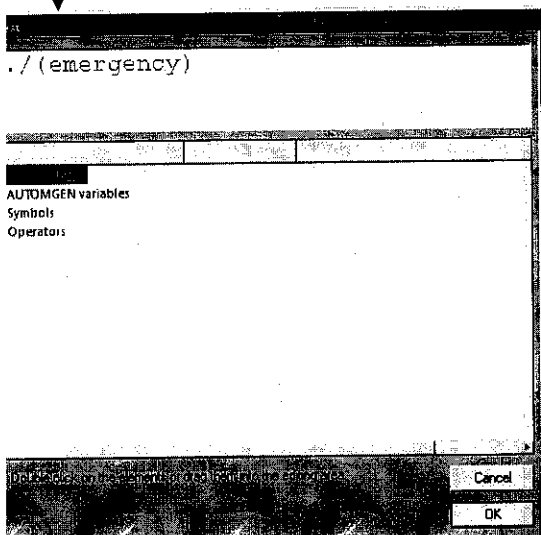
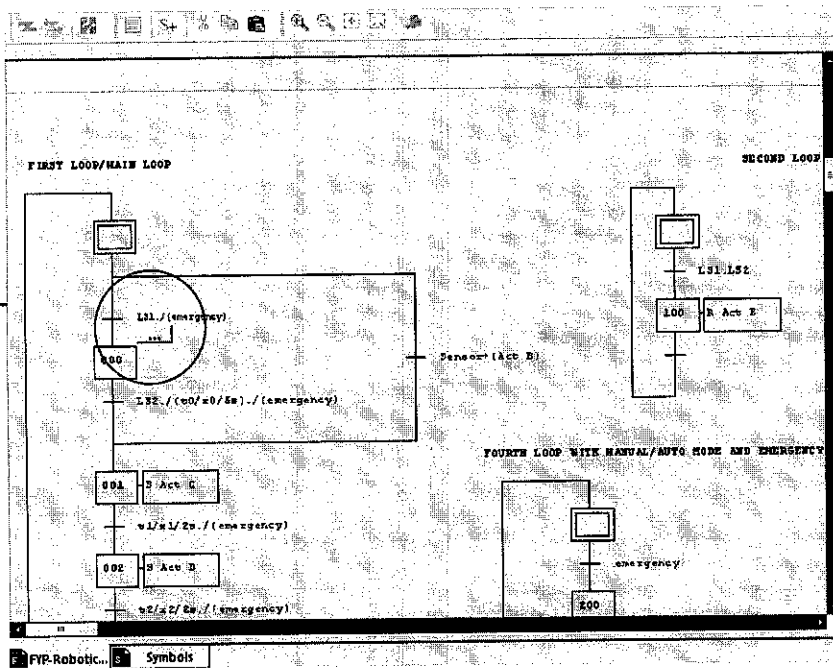
Automatic symbols

It can be a nuisance to have to set the attribution in each symbol and a variable, particularly if the precise attribution of a variable number is not very important. Automatic symbols are a solution to this problem, they are used to let the compiler automatically generate the attribution of a symbol to a variable number. The type of variable to use is provided in the name of the symbol.








4. Start designing the Grafcet by simply right-clicking the work space and choose the Grafcet notions. (Refer to the Grafcet block for the syntaxes).













- After the Grafset code is complete, assigned the transitions (input) and output corresponds to the input. Click at the transition and browse the input from the symbol list.

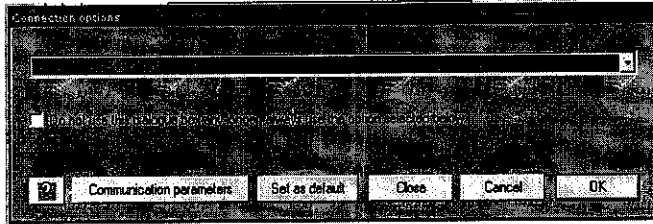


Grafcet Block

Aspect	Associated key	Generic name	Comments	Languages
	[B]	Step	Normal step	Grafcet
	[C]	Initial step without activation	Initial step without activation	Grafcet
	[D]	Initial step	Initial step	Grafcet
		Macro-step	Only available in the shortcut menu	Grafcet
	[T]	Transition	Transition	Grafcet
	[K]	Left limit of an « And » divergence	Compulsory to the left of an « And » divergences	Grafcet
	[L]	Supplementary branch of an « And » divergence or an « And » convergence	Do not use as a left or right limit of an « And » divergence	Grafcet

	[M]	Right limit of an « And » divergence	Compulsory to the right of an « And » divergence	Grafcet
	[N]	Extension of an « And » divergence	If placed in the [K], [L], [M], [P] or [O],[P],[Q],[L] blocks	Grafcet
	[O]	Left limit of an « And » convergence	Compulsory to the left of an « And » convergence	Grafcet
	[P]	Supplementary branch of an « And » convergence or an « And » divergence	Do not use as a left or right limit of an « And » convergence	Grafcet
	[Q]	Right limit of an « And » convergence	Compulsory to the right of an « And » convergence	Grafcet
	[R]	« Or » divergence	Do not use as a limit of an « Or » convergence	Grafcet
	[S]	« Or » convergence	Do not use as a limit of an « Or » divergence	Grafcet
	[U]	Skip or repeat left step	« Or » convergence or divergence	Grafcet
	[V]	Skip or repeat right step	« Or » convergence or divergence	Grafcet
	[SPACE] on an [E] block	Link towards the top	For relooping and repeating steps	Grafcet

3. Compile the program. (Program-Compile) or Alt-C
4. Establish connection with the PLC.



5. Make sure the setting of communication parameters is same as the series of the processor.

