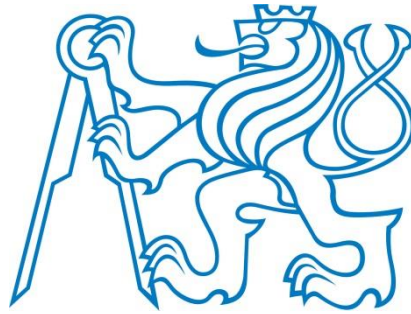


CZECH TECHNICAL UNIVERSITY IN PRAGUE



FACULTY OF MECHANICAL ENGINEERING

DEPARTMENT OF INSTRUMENTATION AND CONTROL ENGINEERING

Master's thesis in Instrumentation and Control Engineering

**CONTROL AND VIZUALIZATION OF THE SPECIAL PNEUMATIC DRIVE
APPLICATION**

YUTIAN HU

Prague, 2018

SUPERVISOR: ING. MARIE MARTINÁSKOVÁ, Ph.D.



MASTER'S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Hu Yutian** Personal ID number: **463551**
Faculty / Institute: **Faculty of Mechanical Engineering**
Department / Institute: **Department of Instrumentation and Control Engineering**
Study program: **Mechanical Engineering**
Branch of study: **Instrumentation and Control Engineering**

II. Master's thesis details

Master's thesis title in English:

Control and vizualization of the special pneumatic drive application

Master's thesis title in Czech:

Řízení a vizualizace pro systém se speciálním pneumatickým pohonem

Guidelines:

1. Prepare the analysis of the application chosen
2. Design control system in the SW environment FLuidSIM for the virtual PLC in the GRAFCET language
3. Design implementation of the part of the control algorithm for the PLC SIMATIC S7-200
4. Design visualisation of the part of the system in the SCADA systém Reliance
5. Debug the whole system in the real time

Bibliography / sources:

1. Documentation from Siemens company for the PLC S7-200
2. Documentation from GEOVAP company for the SCADA system Reliance
3. IEC 61131-3

Name and workplace of master's thesis supervisor:

Ing. Marie Martinásková, Ph.D., U12110.3

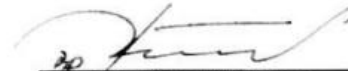
Name and workplace of second master's thesis supervisor or consultant:

Date of master's thesis assignment: **18.04.2018** Deadline for master's thesis submission: **15.06.2018**

Assignment valid until: _____


Ing. Marie Martinásková, Ph.D.
Supervisor's signature

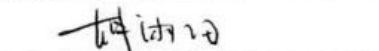

Head of department's signature


prof. Ing. Michael Valášek, DrSc.
Dean's signature

III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

18. 4. 2018
Date of assignment receipt


Student's signature

I declare that I have worked out this thesis independently assuming that the results of the thesis can also be used at the discretion of the supervisor of the thesis as its co-author. I also agree with the potential publication of the results of the thesis or of its substantial part, provided I will be listed as the co-author.

Prague, 2018

.....

Signature

ACKNOWLEDGMENT

I would like to thank my supervisor, Ing. Marie Martínšková Ph.D, who has provided me with valuable guidance in every stage of the writing of this thesis. Without her enlightening instruction, impressive kindness and patience, I would not be able to complete my thesis. I appreciate that she gave me the permission to use the Lab 109 for my experiments.

And grateful acknowledgment belongs to Ing. Mgr. Jakub Jura, Ph.D. for his help with PLC and visualization support, Ing. Vladimír Hlaváč, Ph.D. for his consultations and advice. I would like to thank other colleagues from the department of instrumentation and control engineering. All of the hardware devices are supported by Laboratory of Industrial Automation, Faculty of Mechanical Engineering, Czech Technical University in Prague.

I also would like to thank my family for their understanding, and encouragement.

ANNOTATION

Yutian Hu: Control and visualization of the special pneumatic drive application

Master's thesis (supervisor: Ing. Marie Martín áskov á Ph.D.). Department of Instrumentation and Control Engineering, Faculty of Mechanical Engineering – Czech Technical University in Prague, Prague 2018, 69 pages.

Pneumatic manipulation systems are widely used in automation industry because they are safe, reliable and easy to control. This thesis presents the application of pneumatic manipulation system with special drives and different types of actuators from FESTO. The whole process includes actuator sequential design, simulation design with GRAFCET language, PLC programming, hardware connection, communication establishment, and SCADA visualization design. The ultimate goal of this application is to perform those pneumatic actuators moving as its designed sequences in real-time and monitor the current state of the process by using Reliance SCADA software.

LIST OF CONTENTS

1	INTRODUCTION	1
1.1.	HISTORY AND BACKGROUND	1
1.2.	PROBLEM DESCRIPTION.....	1
1.3.	OVERVIEW OF CHAPTERS	2
2	SOFTWARE USED DESCRIPTION	4
2.1.	FLUIDSIM® PNEUMATICS	4
2.2.	SCADA SYSTEM	4
2.3.	RELIANCE SCADA	5
2.4.	SIMATIC STEP 7- MICRO/WIN	5
3	GRAFSET LANGUAGE.....	7
3.1.	GRAPHIC REPRESENTATION OF THE ELEMENTS	7
3.1.1.	STEPS	7
3.2.	GRAPHIC REPRESENTATION OF SEQUENCE STRUCTURES	10
3.2.1.	LINEAR SEQUENCE.....	10
3.2.2.	ALTERNATIVE BRANCHING.....	11
3.2.3.	PARALLEL BRANCHING.....	11
3.3.	STRUCTURING OF GRAFCETS AND PARTIAL GRAFCETS.....	13
3.3.1.	COMPULSORY COMMANDS	13
3.3.2.	ENCLOSING STEP	14
4	SPECIAL DRIVES AND COMPONENTS USED	16
4.1.	SPECIAL PNEUMATIC DRIVES AND SENSORS.....	16
4.1.1.	SEMI-ROTARY DRIVE	16
4.1.2.	LINEAR DRIVE	18
4.1.3.	MAGNETIC SENSOR	19
4.2.	NORMAL PNEUMATIC ACTUATORS.....	20
4.2.1.	DOUBLE ACTING CYLINDER	20
4.2.2.	DOUBLE SOLENOID VALVE	21
4.2.3.	ONE WAY FLOW CONTROL VALVE	22
4.2.4.	VACUUM SUCTION NOZZLE.....	22
4.2.5.	SUCKER.....	23

4.2.6.	LIMIT PROXIMITY SENSOR	23
5	COMPUTER SIMULATION DESIGN AND PROGRAM	25
5.1.	MODES DESCRIPTION	28
5.2.	HARDWARE SIMULATION CONNECTION	31
5.3.	LADDER PROGRAM SIMULATION	32
6	LABORATORY 109 SIMULATION	36
6.1.	PLC SELECTION	36
6.1.1.	<i>S7-200 SERIES</i>	37
6.1.2.	<i>CP 243-1 COMMUNICATIONS PROCESSOR FOR INDUSTRIAL ETHERNET</i>	38
6.2.	I/O TABLE	39
6.3.	HARDWARE APPLICATION OF STATION 1.....	41
6.4.	HARDWARE APPLICATION OF STATION 2.....	42
7	VISUALIZATION OF THE SYSTEM.....	43
7.1.	OPC SERVER AND CLIENTS.....	43
7.2.	ETHERNET WIZARD AND PC ACCESS	43
7.3.	COMMUNICATION WITH RELIANCE SCADA	46
7.4.	DATA EXCHANGE POSSIBILITIES BETWEEN CONTROLLERS	47
7.4.1.	<i>SCRIPT METHOD</i>	47
7.4.2.	<i>EVENT METHOD</i>	48
7.4.3.	<i>HARDWARE METHOD</i>	48
7.5.	VISUALIZATION DESIGN IN RELIANCE.....	49
7.6.	RUNTIME WINDOW OF THE PROJECT.....	51
8	CONCLUSION	53
	REFERENCES	55
	APPENDIX A – PLC CODES	56
A.1	PROGRAM CODE OF STATION 1	56
A.1	PROGRAM CODE OF STATION 2	61
	APPENDIX B – RELIANCE 4 SCRIPT CODES.....	67
B.1	SCRIPT CODE OF EXCHANGING DATA.....	67
B.2	SCRIPT CODE OF VIRTUAL STATION 1	68

LIST OF FIGURES

FIGURE 1: A STEP	8
FIGURE 2: INITIAL STEP	8
FIGURE 3: ACTION CONNECTED TO A STEP	8
FIGURE 4: TRANSITION	9
FIGURE 5: EXAMPLE OF A LINEAR SEQUENCE	10
FIGURE 6: EXAMPLE OF ALTERNATIVE SEQUENCE.....	11
FIGURE 7: EXAMPLE OF PARALLEL SEQUENCE	12
FIGURE 8: EXAMPLE OF PARALLEL SEQUENCE	13
FIGURE 9: EXAMPLE OF COMPULSORY COMMANDS	14
FIGURE 10: EXAMPLE OF ENCLOSING STEP.....	15
FIGURE 11: ENCLOSED STEP PROPERTIES	15
FIGURE 12: PICTURE AND SYMBOL OF A SEMI-ROTARY DRIVE	17
FIGURE 13: ROTARY DRIVE APPLICATION IN MPS	17
FIGURE 14: ROTARY DRIVE APPLICATION IN MPS	18
FIGURE 15: PICTURE AND SYMBOL OF A SEMI-ROTARY DRIVE	18
FIGURE 16: HANDLING SYSTEM	19
FIGURE 17: PICTURE AND SYMBOL OF A MAGNETIC SENSOR	19
FIGURE 18: A MAGNETIC SENSOR IN THE SLOT OF THE LINEAR DRIVE	20
FIGURE 19: PICTURE AND SYMBOL OF A DOUBLE ACTING CYLINDER	20
FIGURE 20: DOUBLE ACTING CYLINDER AS A FEEDER IN MPS	21
FIGURE 21: PICTURE AND SYMBOL FOR A DOUBLE SOLENOID VALVE.....	22
FIGURE 22: PICTURE AND SYMBOL OF A ONE WAY FLOW CONTROL VALVE	22
FIGURE 23: PICTURE AND SYMBOL OF A VACUUM SUCTION NOZZLE	23
FIGURE 24: PICTURE AND SYMBOL OF A SUCKER	23
FIGURE 25: PICTURE AND SYMBOL OF A LIMIT SENSOR	24
FIGURE 26: SCHEME OF THE PRODUCTION LINE.....	25
FIGURE 27: MANIPULATION SYSTEM IN FLUIDSIM®.....	26
FIGURE 28: STATE DIAGRAM OF THE SYSTEM.....	27
FIGURE 29: MAIN CONTROL PART	28
FIGURE 30: MANUAL MODE	29
FIGURE 31: RESET MODE.....	29

FIGURE 32: AUTOMATIC MODE IN GRAFCET	30
FIGURE 33: HARDWARE CONNECTION 1 IN FLUIDSIM®	31
FIGURE 34: HARDWARE CONNECTION 2 IN FLUIDSIM®	31
FIGURE 35: HARDWARE CONNECTION 3 IN FLUIDSIM®	32
FIGURE 36: STATION 1 IN FLUIDSIM®	33
FIGURE 37: STATION 2 IN FLUIDSIM®	33
FIGURE 38: MAIN CONTROL IN LADDER LOGIC	33
FIGURE 39: MANUAL MODE IN LADDER LOGIC	34
FIGURE 40: STATION 1 MAIN CONTROL IN LADDER LOGIC	35
FIGURE 41: STATION 1 (LEFT) AND STATION 2 (RIGHT) IN LAB 109	36
FIGURE 42: EXTENSION MODULE	37
FIGURE 43: ETHERNET MODULE CP243-1	38
FIGURE 44: STATION 1 IN LAB 109.....	41
FIGURE 45: STATION 2 IN LAB 109.....	42
FIGURE 46: ETHERNET CONTROL BLOCK.....	44
FIGURE 47: ETHERNET WIZARD IN MICRO/WIN	44
FIGURE 48: SETTING IN PC ACCESS.....	45
FIGURE 49: VARIABLES IN PC ACCESS	45
FIGURE 50: TEST CLIENT IN PC ACCESS	46
FIGURE 51: LIST OF TAGS FROM THE REAL PLC	46
FIGURE 52: DEVICE MANAGER IN RELIANCE 4.....	47
FIGURE 53: CREATING EVENTS IN RELIANCE 4.....	48
FIGURE 54: EVENT-RELATED TAG	48
FIGURE 55: RELIANCE SCADA SYSTEM OF THE PROCESS WITH TOP VIEW	49
FIGURE 56: RUN TIME WINDOW FOR THE STATION 1 FROM THE SCADA RELIANCE PROJECT.....	51

LIST OF TABLES

TABLE 1: LIST OF USED COMPONENTS..... 16

TABLE 2: INITIAL POSITION OF EACH ACTUATOR..... 27

TABLE 3: SIGNAL DESCRIPTION..... 34

TABLE 4: DATA OF PLCs AND MODULES IN THE LAB..... 37

TABLE 5: I/O TABLE FOR STATION 1 39

TABLE 6: I/O TABLE FOR STATION 2 40

LIST OF ABBREVIATION

PLC	Programmable Logic Controller
LAD	Ladder Logic
HMI	Human Machine Interface
SCADA	Supervisory Control And Data Acquisition
PC	Personal Computer
OPC	OLE for Process Control
TSAP	Transport Service Access Point
I/O	Inputs and Outputs
MPS	Modular Production System

1 INTRODUCTION

1.1. History and background

Pneumatics is the discipline that deals with mechanical properties of gases such as pressure and density and applies the principles to use compressed gas as a source of power to solve engineering problems. The most widely used compressed gas is air, and thus its use has become synonymous with the term pneumatics.

The use of air as an energy transfer medium can be traced back more than 2000 years, and the varying areas of application reflect the changes in technology since then. The industrial use on a larger scale began 1888 when a 1,500 kW central compressor station was installed in Paris to supply the city with compressed air (Neermann 1989). With the evolution of electric power, this form of energy transfer became obsolete, but the competition between fluid power systems and electric systems is still going on [1].

If the high production of components or devices is needed, an automatic production line is necessary to be introduced to realize the operations. For the application in this thesis, a set of sequential operations could be achieved by pneumatic manipulation stations. Every pneumatic actuator performs a specific operation and the product is processed step by step as it moves along the line in the pre-defined production sequences.

1.2. Problem description

Master's thesis guidelines:

- Prepare the analysis of the application chosen
- Design pneumatic manipulation system in the SW environment FluidSIM®® for the virtual PLC in the GRAFCET language
- Design implementation of the part of the control algorithm for the PLC SIMATIC S7-200

- Design visualization of the system in the SCADA system Reliance
- Debug the whole system in the real-time

To design the sequential movements, it takes some time to come up with an idea of how to connect those pneumatic actuators together in a reasonable way when the type of actuators is limited. Also, it is only possible to build the stations in two axes.

Adding more modes of the machine is something new to me. I was only experienced with automatic mode. So it requires more research into the definition of different modes and functionalities, the way of switching mode, categories of emergency stop, and how to realize mode selection in GRAFCET. It is necessary to study GRAFCET language deeply.

The first try on Reliance SCADA software was not successful because the software is not very smart. For example, it does not have enough pictures provided in its library, meaning that I have to create pictures and make it with a transparent background. Plus, the size of the picture in the software cannot be adjusted. Therefore, the size must be exactly same as what I want when it is created. But I decided to use normal vectors to represent models of actuators, which is simple, clear and much faster to do.

When the implementation of the real system in Lab 109 started, I realized that all of seven actuators cannot fit in the same experiment table. They must be placed in two separate places, which would increase the difficulty of control and communication between stations since they are controlled by two PLCs that are unknown to each other.

1.3. Overview of chapters

The whole thesis contains 9 chapters and 2 appendixes.

Chapter 2 introduces the technology and software that are used in the application.

Chapter 3 is about GRAFCET language, which has information from the basic elements of the representation and different structures about the program with real examples. The simulation of

the pneumatic manipulation system in FluidSIM® has used some higher level GRAFCET blocks and functions. They are explained in Chapter 3.

Chapter 4 lists all of the pneumatic components that appeared in the application with principle explanation, pictures, and its symbols in FluidSIM®.

Chapter 5 introduces the simulation of the pneumatic manipulation system in two different programming languages (GRAFCET and LAD). It consists of the logic of the program, modes description and the instruction of operations.

Chapter 6 is about how it is implemented in Lab 109. Pictures of stations, information about controllers and I/O tables can be found there.

Chapter 7 talks about visualization mostly. It explains how the communication is established and how the project in Reliance SCADA works.

2 SOFTWARE USED DESCRIPTION

2.1. FluidSIM® Pneumatics

The software FluidSIM® has pneumatics and hydraulics. In our pneumatic manipulation system, FluidSIM® Pneumatics is used for designing and running the simulation of the system. It is a tool that teaches and helps people to understand how pneumatic systems work. It offers different kinds of actuators and components from pneumatics and electrical field. One of the main features of this software is that every pneumatic actuator in the library has its text description, picture. And the principle of pneumatic components can be illustrated by their animations when the simulation is running.

In FluidSIM®, simulations of pneumatics circuits are based on real models. Parameters of components are able to be changed. For example, the length of the double acting cylinder, positions of sensors, the structure of valves, etc. FluidSIM® supports programming in GRAFCETs language and LAD to operate sequential actions, which are used primarily in our simulation.

2.2. SCADA system

A SCADA (or supervisory control and data acquisition) system means a system consisting of a number of control terminal units collecting field data connected back to a master station via a communications system. The master station displays the acquired data and also allows the operator to perform remote control tasks [2]. For example, SCADA system is used in electric power, automation production, petroleum, chemical for gathering data and monitoring that everything is working under control. The peripheral hardware and actuators mostly are PLCs and actuators with sensors of the plants. People are able to monitor the whole process such as the position of the valve and issue commands to the plant through user interfaces of the SCADA control system. In order to achieve remote control by using SCADA systems, operators can send signals to devices and actuators that are connected with communication field bus.

A SCADA system works by operating with signals that communicate via channels to provide the user with remote controls of any equipment in a given system. In the meantime, it generates a database that has tags throughout the manipulation system. System inputs and outputs value are linked to those tags that are supervised or controlled by the SCADA system in the centralized control room.

Nowadays a lot of water, electric, and manufacture companies are using manual labor to perform measurements and adjustments. However, those tasks could be easily automated by using SCADA systems. With implement of automation plant, labor costs are possible to be cut as well as minimize errors with measurements or adjustments.

2.3. Reliance SCADA

Reliance is a professional SCADA/HMI system designed for the visualization and control of industrial processes and for achieving automation. The runtime software is a program designed to run a visualization project on the end user's computer. It is used to control and monitor the visualized industrial process. The server modules provide data to connected clients, i.e., either to client instances of the runtime software or to the thin clients [3].

1. Increasing the quality and productivity of the production process
2. Minimizing technological breakdowns by early warning the system operator
3. Reliability of the application due to the redundancy of data communication lines
4. Ability to subsequently analyze the cause of a possible technological breakdown

2.4. SIMATIC STEP 7- MICRO/WIN

SIMATIC STEP 7- Micro/WIN is the industry software that can configure and program Siemens programmable logic controllers. It serves for defining variables and addresses, PLC programming, programs uploading and downloading, debugging, monitoring current states, etc. The languages for programming in STEP 7- Micro/WIN are from IEC 61131-3. Ladder diagram, Statement List, and Function Block Diagram can be used in STEP 7- Micro/WIN.

The language being used in our real application manipulation system is Ladder logic. Ladder logic is one of the most frequently used programming languages for PLCs and it is very close to old-fashioned relay ladder logic. It is using graphical relays and coils to represent its circuits.

STEP 7 for applications on SIMATIC S7-300/S7-400, SIMATIC M7-300/M7-400, and SIMATIC C7 with a wider range of functions:

- Can be extended as an option by the software products in the SIMATIC Industry Software
- The opportunity of assigning parameters to function modules and communication processors
- Forcing and multi-computing mode
- Global data communication
- Event-driven data transfer using communication function blocks
- Configuring connections

3 GRAFCET LANGUAGE

This chapter introduces important properties and functionality of each component of GRAFCET language and how the GRAFCETs of the application are created and simulated with FluidSIM®.

GRAFCET is a graphical chart with an associated interpretation. The chart is made up of a finite set of step nodes and transition nodes, connected by directed actions. Such a chart may describe the behavior of the whole pneumatic manipulation system or any part of it [5]. Also, GRAFCET allows users to monitor the running program, so it is easy to read and troubleshoot the errors by highlighting the step that we are stuck.

The essential components of a GRAFCET are steps, transitions, and actions. One or more actions can be assigned to a step. Actions can be executed when the step is activated. A transition is where you can set conditions. If the conditions are met, the program will move to next step. Conditions and actions define how the sequential operations are linked together, regardless of the application of different types of hardware and professional software. The GRAFCET language is integrated into the software FluidSIM®. Besides those basic elements of GRAFCET, FluidSIM® offers higher level programming functions which will be introduced in this chapter later.

3.1. Graphic representation of the elements

3.1.1. Steps

Steps are either active or inactive, and actions can link to it. The actions of active steps are executed. The sequence of a GRAFCET is described by the transitions from a previous to a subsequent step. Steps and transitions have to alternate in the plan. The creation and simulation of GRAFCETs in FluidSIM® is illustrated in the following using simple examples. The example of linking steps is shown in Figure 1.

A step in GRAFCET represents one state of the manipulation process and it has two different states. The step could be activated or deactivated. It works from top to bottom. If the step is

active, the corresponding actions (Control commands) will be performed. Deactivated steps wait for calling or next loop.



Figure 1: A step

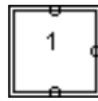


Figure 2: Initial step

3.1.2 Action

Actions are used to be commands that are sent to process outputs. One step is able to execute any number of commands at the same time. They need to be linked to a step. It is not necessary to connect every action to the step directly. The alternative way of doing it is to place actions next to others. The software would create the link automatically between actions, which can make the design work easier.

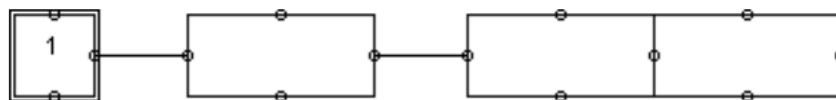


Figure 3: Action connected to a step

Actions are called in the list of output's textual names or it is possible to set its variable values. A programmer can decide the way of displaying actions. It could be either the textual description or the signal name shown within an action. If users hope to see the description of that command, to display the comment or description of the signal, users can check the option "Display description instead of formula", which can be found in the properties of the action.

3.1.3 Transition

Transitions are the key to transfer from the previous step to the next step. They are used to be Boolean conditions of the sequential operation, like the position sensors and START/STOP pushbuttons. According to the application, an on-delay and off-delay could be used to control the sequential movements precisely.

Depends on the structure of a GRAFCET, one transition is able to lead to one or several steps. The changing to the program state happens when all the previous steps are activated, and the Boolean condition of that transition has to be true. Once the current conditions are met, the following states are active and the previous steps will be deactivated. Transitions are added in a simple program shown in Figure 4.

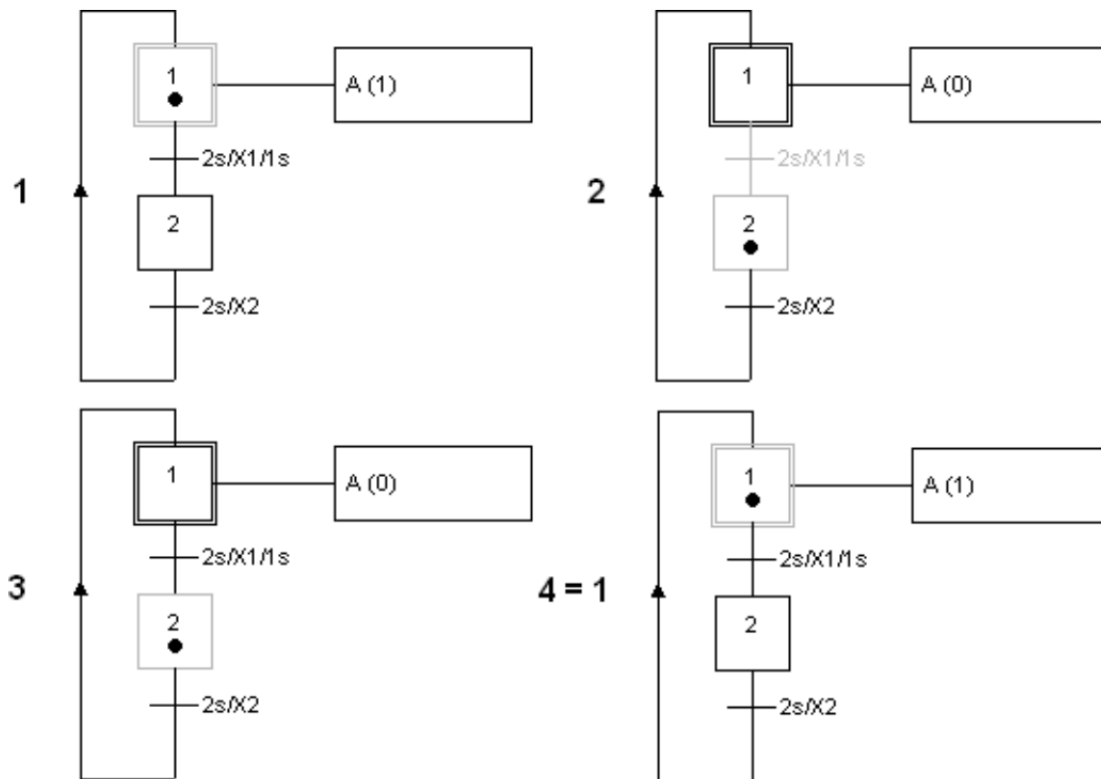


Figure 4: Transition

From above picture, we can see that how the steps changes in GRAFCET. In this application, “X1” and “X2” are the variables for the steps. The value of the variable will become to “1” when the step is active. The time delay can be added on the transitions with the following form: t1/ “Signal name”/ t2” where t1 and t2 should be substituted by numbers meaning how many seconds that we want to delay. And the signal should be a Boolean term. If there is no time delay to the signal, then the next following step will be activated immediately when the transition is fulfilled. The transition is in green when the conditions have value “1”. The Boolean state of an actuator in the action could be used as transition content. In this case, the signal name of the transition should be as same as the action name.

3.2. Graphic representation of sequence structures

There are three basic types of sequence structures, which are generated by combining steps and transitions.

3.2.1. Linear sequence

In linear sequence, every step is followed by one subsequent transition, which would be activated by means of a single step within the sequence in series, excluding the first step. And every step connects to only one subsequent transition, with the exception of the ending step.

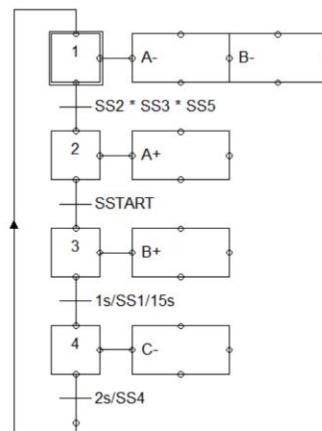


Figure 5: Example of a linear sequence

3.2.2. Alternative branching

For alternative branching, it has more than one transitions follow a step. The branch would be activated when the first sub-sequence transition condition is fulfilled. Due to the fact that precisely one sub-sequence has to be chosen in the case of alternative branching, the various transition conditions must be mutually exclusive.

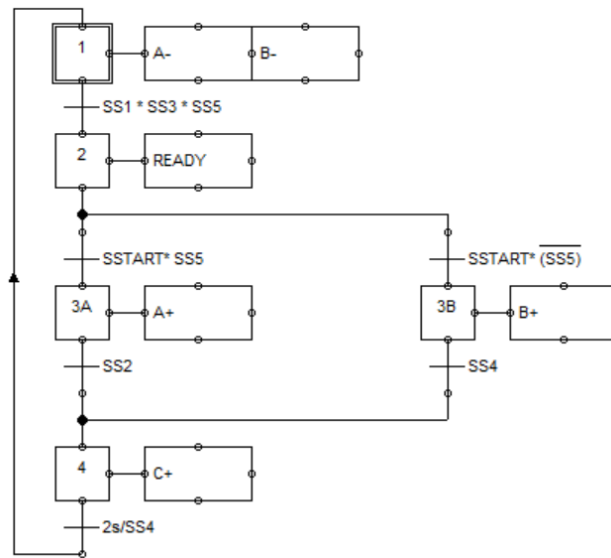


Figure 6: Example of alternative sequence

3.2.3. Parallel branching

Under the parallel branching, some sub-sequences will be activated simultaneously when a single transition condition is fulfilled. But each branch is working independently. The sub-sequences are merged back into the main primary sequence. The step underneath the second double line will not be carried out until all of the parallel sub-sequences have been fully processed and its transition conditions are met.

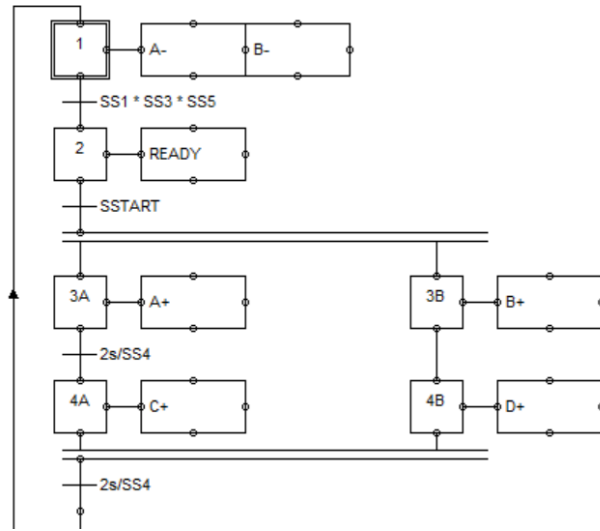


Figure 7: Example of parallel sequence

3.3. Structuring of GRAFCETS and partial GRAFCETS

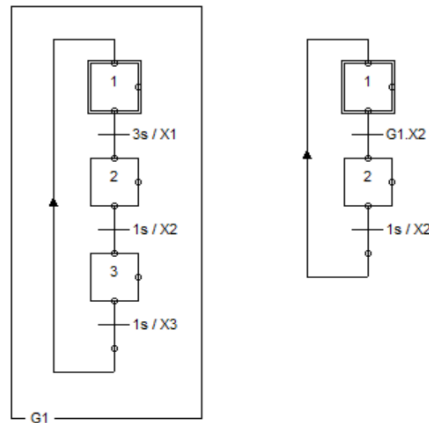


Figure 8: Example of parallel sequence

Partial GRAFCETS is a good solution to separate the program depending on its function, which makes a GRAFCET with one layer become several layers. Partial GRAFCETS are used to be activated when the active step comes to enclosing steps and forcing commands. The difference between the main GRAFCET and partial GRAFCET is that the latter one has a frame placed over that part and programmer has to assign a name to it. The name will be displayed at the bottom left of the frame. There is a “G” that is created automatically by FluidSIM®. The programmer can change the size of the frame to make sure that every element of the partial GRAFCET is placed within the frame. Otherwise, it would be problematic to run the program.

3.3.1. Compulsory commands

Compulsory commands or forcing commands are made for controlling certain steps without following the normal sequence flows. In FluidSIM®, we can find four forcing commands with different functions. To import compulsory commands from the library in FluidSIM®, drag a normal action first, and it will be on the list of types after double checking on it. The form of a different function of commands is shown in Figure 9.

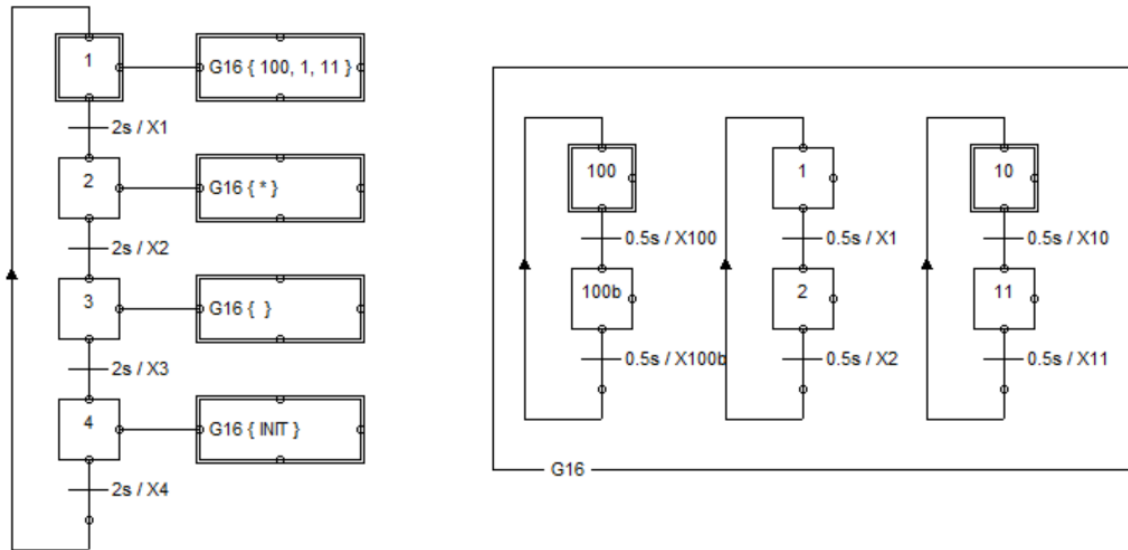


Figure 9: Example of Compulsory commands

G16 {100, 1, 11}: Jumping to certain steps. In the example above, the system jumps to steps 100, 1 and 11 of partial GRAFCET 16.

G16 {*}: Freezing the current situation. If this command is called during simulation, every component will maintain its current state. No other transitions will be carried out. This command can be applied to an emergency stop.

G16{ }: Removing active steps. The active in partial GRAFCET 16 steps will be set to blank when this command is working.

G16 {INIT}: Initialize all of GRAFCET immediately. In this example, every initial step (With double frame) will be activated in the partial GRAFCET 16.

3.3.2. Enclosing step

Enclosing steps offer an extra structure of GRAFCETs. The symbol of enclosing steps is an octagon with its step number in the center of the box. An enclosed step is used to link a partial GRAFCET with a frame. In FluidSIM®, the enclosing step and its corresponding partial GRAFCET use an identical name.

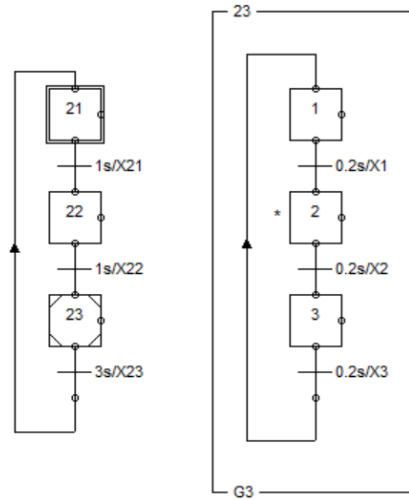


Figure 10: Example of enclosing step

One thing needs to be mentioned is that programmers have to manually set the “Activation link” in the enclosed step properties, which chooses the step that is expected to activate when enclosing step is being called. The chosen step has an asterisk next to it as shown in the example.

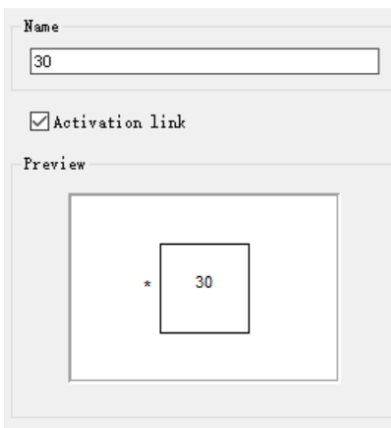


Figure 11: Enclosed step properties

4 SPECIAL DRIVES AND COMPONENTS USED

Name	Number
Double acting cylinder	2
Semi-rotary drive	3
Vacuum suction nozzle	2
Linear drive	1
5/2 valve	8
Flow control valve	2
Sucker	2
Limit sensor	6
Magnetic sensor	1

Table 1: List of used components

4.1. Special pneumatic drives and sensors

4.1.1. Semi-rotary drive

The semi-rotary drives in our application are 180-degree actuators with two stop positions (0° and 180°). The rotary rod has an arm attached to it, and the end of the arm mounts a vacuum sucker. The purpose of this pneumatic actuator is to connect two stations transporting working pieces from point A to point B.

Although the rotating range is 180 degrees, it is difficult to make it stop at a certain position but only end positions (0 degrees and 180 degrees) by using a 5/2 valve. It is possible to use 5/3

valve, which allow the valve to stop somewhere between ending positions. However, this way of controlling is not very accurate. Therefore 5/3 double solenoid valve is not being used.

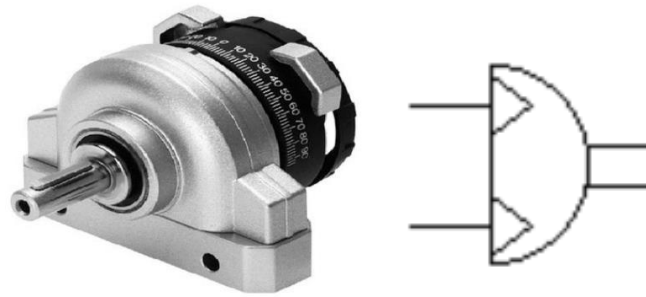


Figure 12: Picture and symbol of a semi-rotary drive

As shown in Figure 13, it presents how a semi-rotary drive is implemented in MPS. On the left hand side there is a semi-rotary drive with an arm and a vacuum sucker. It is taking the working piece from A to B.

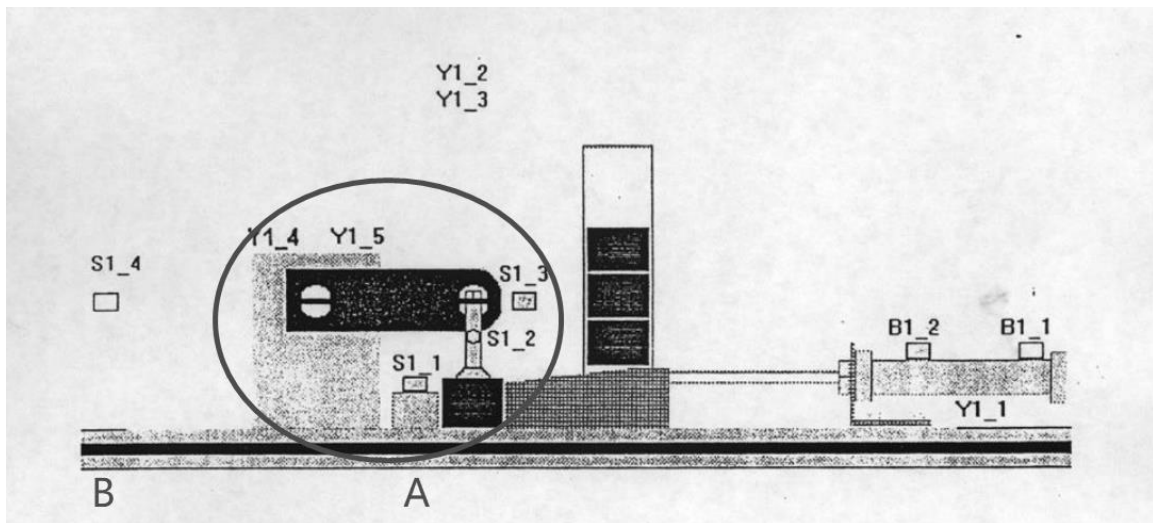


Figure 13: Rotary drive application in MPS



Figure 14: Rotary drive application in MPS

4.1.2. Linear drive

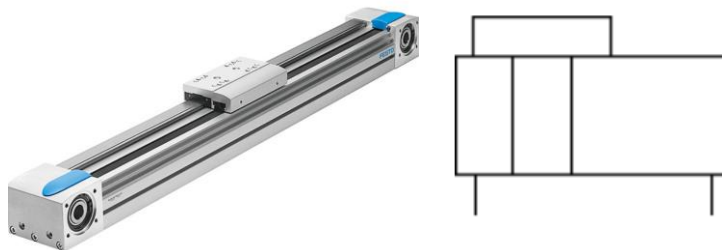


Figure 15: Picture and symbol of a semi-rotary drive

The other pneumatic special drive in the application is a linear drive. The inner structure of the linear drive is different from the double acting cylinder. A linear drive has a piston placed inside of its body and the movement of the piston is controlled by alternating pressured air. Instead of having a piston rod, there is an outer slide or a carriage connected with the piston mechanically. So the carriage would move along with the piston when the pressured air is supplied. A linear drive also equips sensor slots. A cylinder with a pneumatic clamp or a vacuum sucker is usually combined with a linear drive for transportation purpose.

In our design, the linear drive is designed to work with a double acting cylinder and a vacuum sucker in order to catch working pieces and transport it to its target position, like the semi-rotary drive. But the difference between them is its working route. Semi-rotary drives are used in shorter distance transportation, whereas linear drives can be used for long-distance transportation and they can cooperate with each other as a handling system.



Figure 16: Handling system

4.1.3. Magnetic sensor

Magnetic sensors are used for the semi-rotary drive and the linear drive. Each magnetic sensor has three wires, two of them are for power supply, and one is for the signal. A magnetic sensor will be activated when a solenoid is moved nearby. An indicator will be lit if the sensor is active. To fit a sensor with an actuator, a slot is needed in the frame.



Figure 17: Picture and symbol of a magnetic sensor

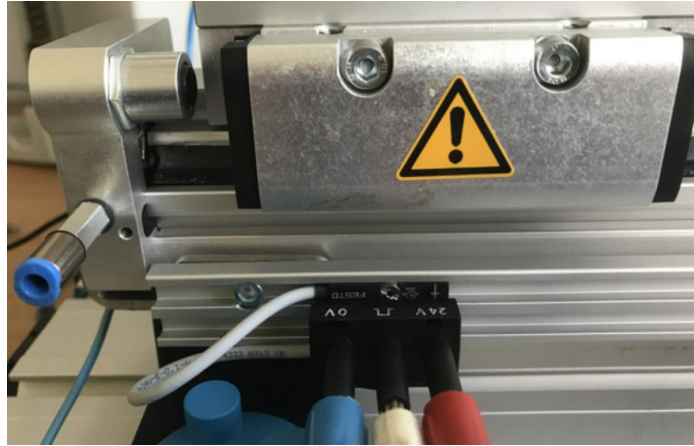


Figure 18: A magnetic sensor in the slot of the linear drive

To ensure the sensor stay still in the slot, it is necessary to tighten the small switch on the body of the sensor. The example is shown in Figure 18.

4.2. Normal pneumatic actuators

4.2.1. Double acting cylinder

Three double acting cylinders are used in the pneumatic manipulation system. The principle of a double-acting cylinder is that pressured air can flow into both sides of the cylinder and move the piston forward and backward. Many cases pneumatic cylinders are used for them when it is needed to produce a force in both directions. There are two ports at both ends on a double-acting pneumatic cylinder, where can connect to the air supply for both retraction movements and extension movements.

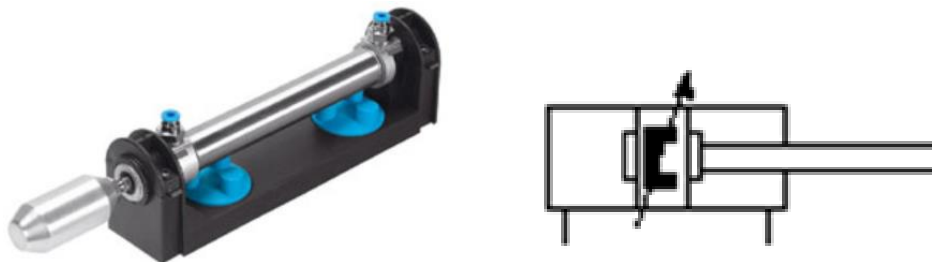


Figure 19: Picture and symbol of a double acting cylinder

In the pneumatic manipulation system, double acting cylinders mainly are for pushing working pieces from its feed magazine and from one position to next position where the next actuator can keep the process going.

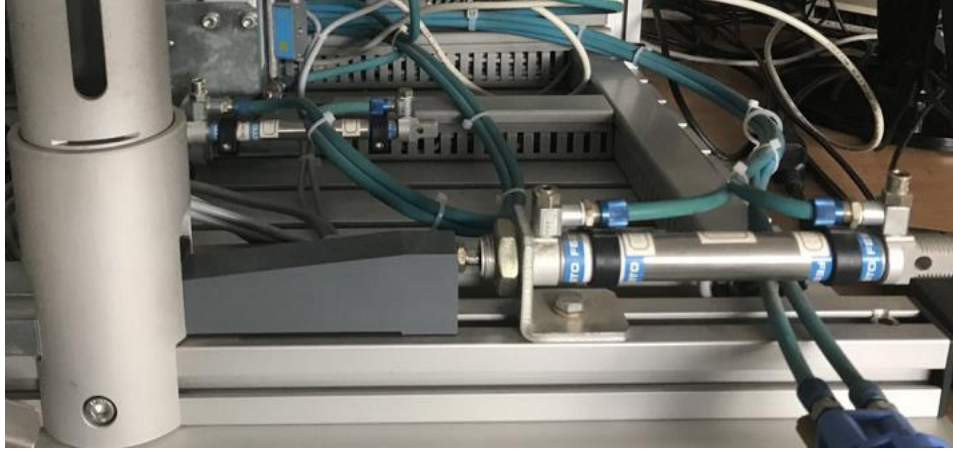


Figure 20: Double acting cylinder as a feeder in MPS

4.2.2. Double solenoid valve

Every actuator in our application is controlled by a double solenoid valve, also known as “bistable” valve. Some of the actuators are not connected directly with it but through a flow control valve when the speed of actuator needs to be adjusted. A double solenoid valve is an electromechanically operated valve. This type of valve is switching states when the corresponding solenoid is activated by an electrical signal. The left side and the right side should not have signal at the same time, as known as signal overlapping. This situation needs to be avoided while programming and operating, as it would cause unknown consequences and it is dangerous in the industrial application.

The valve has three pneumatic ports. One is for pressured air supply. The other two are for moving actuators. Each solenoid side has two ports, where should be connected to controller outputs and negative power supply respectively.

Two manual switches can be found on the top of a double solenoid valve. Operators can easily adjust the state of the valve by twisting the switches.

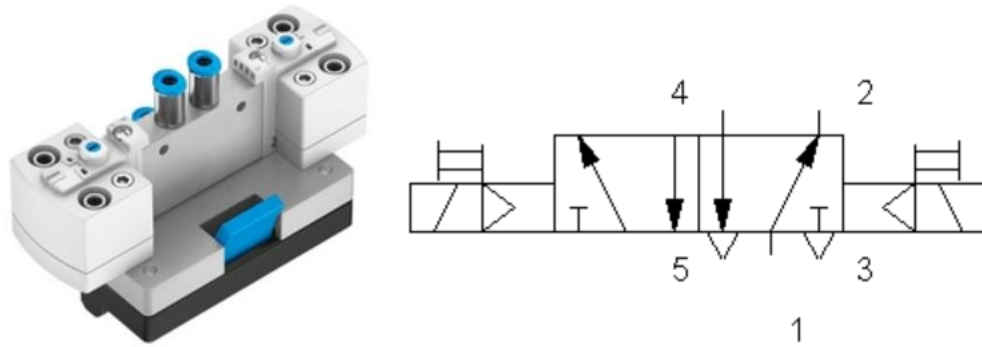


Figure 21: Picture and symbol for a double solenoid valve

4.2.3. One way flow control valve

In order to better visualize the sequential operations or slow down the moving speed of piston rod, flow control valves are introduced to these situations.

The working principle of a one-way flow control valve is that only one direction fluid is allowed to go through. The direction depends on the way how it is connected.

Flow control valves have two openings to plug tubes. Pressure air will flow through one port and leave from the other one. This type of valve works by itself, meaning that they are not controlled by operators or extra signals. There is a knob that is designed to manually adjust its opening level, which will change the actuator moving speed accordingly.

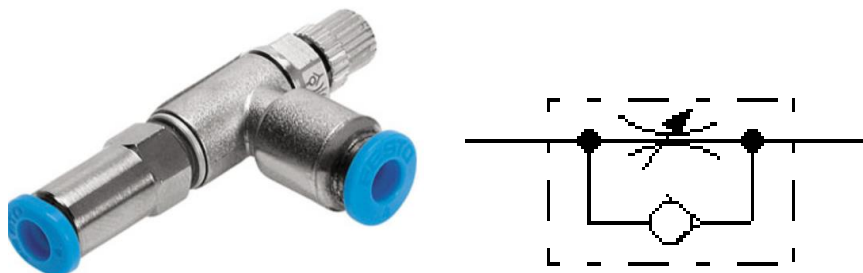


Figure 22: Picture and symbol of a one way flow control valve

4.2.4. Vacuum suction nozzle

The vacuum suction nozzle creates its vacuum based on the ejector principle. In this case,

compressed air from connection 1 to 3 would be creating a vacuum at connection 1v [6]. A vacuum sucker could be connected to a tube. The vacuum suction nozzle has only one opening for compressed air. Therefore only on-off control is applied to it.

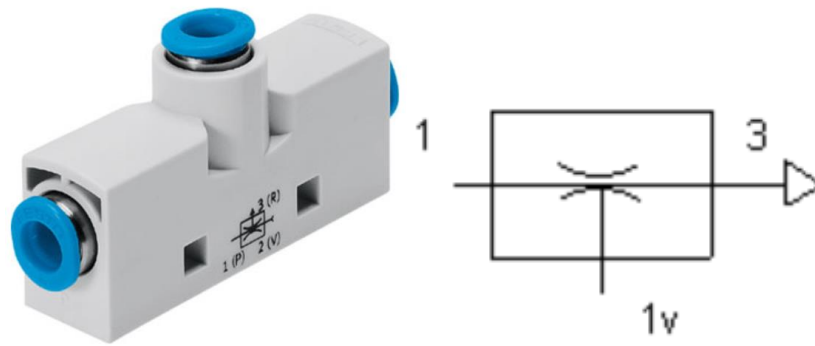


Figure 23: Picture and symbol of a vacuum suction nozzle

4.2.5. Sucker

A sucker is used to lift an object when it is connected to a vacuum suction nozzle. There are many different types of suckers with various shapes and suck forces. Sucker selection should consider what shape and weight of working pieces that needs to be transported. The one used in the pneumatic manipulation system is just a sucker for small, light objects with flat surface.



Figure 24: Picture and symbol of a sucker

4.2.6. Limit proximity sensor

Limit switches or sensors are used to detect the position of the pneumatic cylinders. The sensor will be closed when the cylinder rod reaches the roll lever attached to the limit switch. In

FluidSIM®, creating a limit switch needs to choose sensor's type in its properties by selecting switch with a roll.



Figure 25: Picture and symbol of a limit sensor

5 COMPUTER SIMULATION DESIGN AND PROGRAM

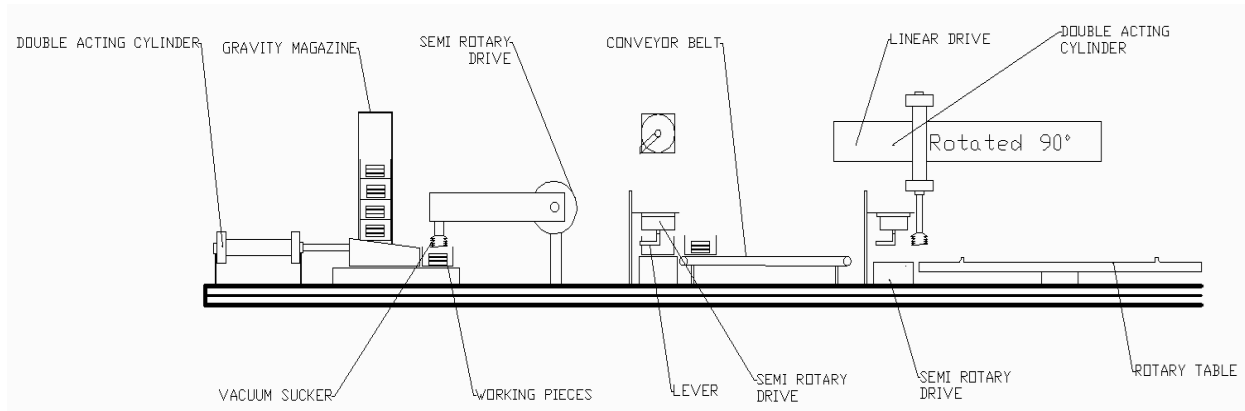


Figure 26: Scheme of the production line

The sequential operations were designed in this way:

One pot with filled goods will be pushed out by the first double acting cylinder from resources (Gravity magazine). The semi-rotary drive with an arm and a vacuum generator is carrying goods to next position (An empty container). The number of working times of the semi-rotary drive depends on how many goods in pots. Next step is that the filled container will be pushed to conveyer belt by another semi-rotary drive with a lever. When it reaches the place under the linear drive, the linear drive starts moving. The double acting cylinder and vacuum generator catches one cover and put it on top of the filled container. Then the semi-rotary drive with lever will push it to the next station.

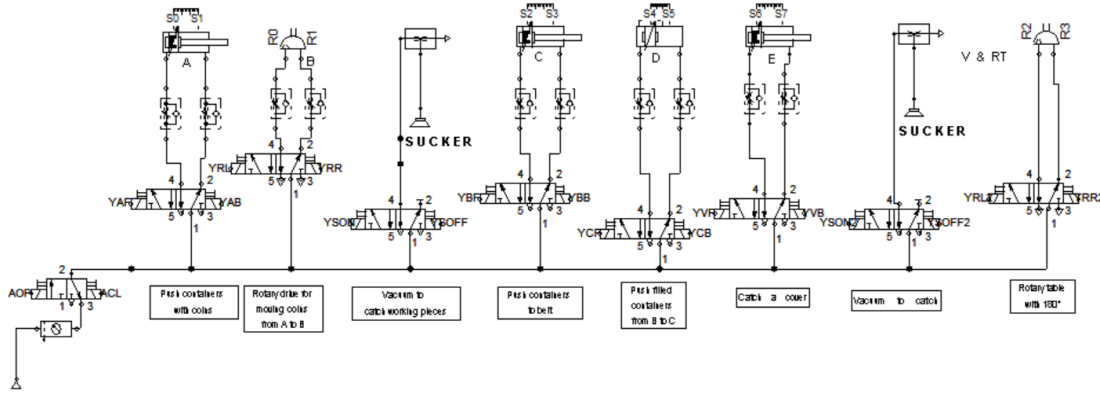


Figure 27: Manipulation system in FluidSIM®

In order to make the sequential operations look neat, a letter is assigned to each pneumatic actuator. And it could be simplified to this form:

A+ A- B- V1 B+ V0 C+ C- E+ V1 E- D- E+ V0 E- D+

Explanation of the abbreviation used:

- “A+ B+” ----- The pneumatic actuator is moving forward
- “A- B-” ----- The pneumatic actuator is moving backward
- “V1” ----- The vacuum generator is on
- “V0” ----- The vacuum generator is off

In order to illustrate the sequential operations better, the state diagram in Figure 28 shows the dynamic of the actuator. It also contains the moving length of the actuators, working time, the speed of the actuators, etc.

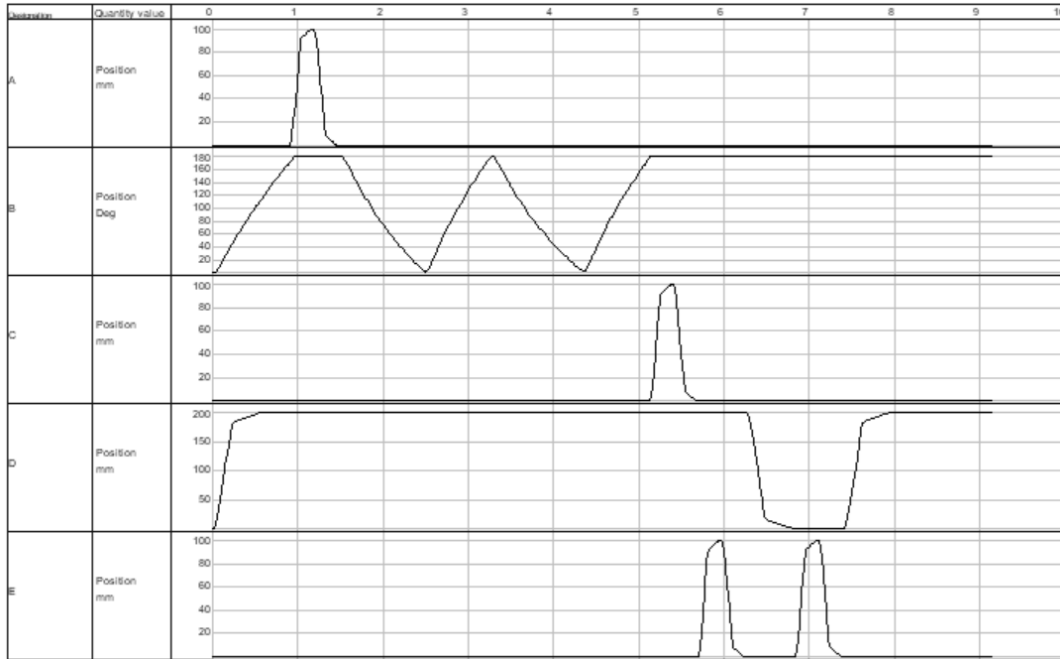


Figure 28: State diagram of the system

The initial position of each actuator has been listed in Table 2.

NAME	INITIAL POSITION
A: Double acting cylinder	RETRACT
B: Rotary drive with arm	EXTEND
Vacuum 1	OFF
C: Double acting cylinder	RETRACT
D: Linear drive	EXTEND
E: Double acting cylinder	RETRACT
Vacuum 2	OFF

Table 2: Initial position of each actuator

5.1. Modes description

Instead of having only an automatic mode, the manipulation system has several different functions of its control. The main control section (Mode selection), Manual mode, Reset mode, automatic mode, air supply cut-off and emergency stop.

a. Main control section:

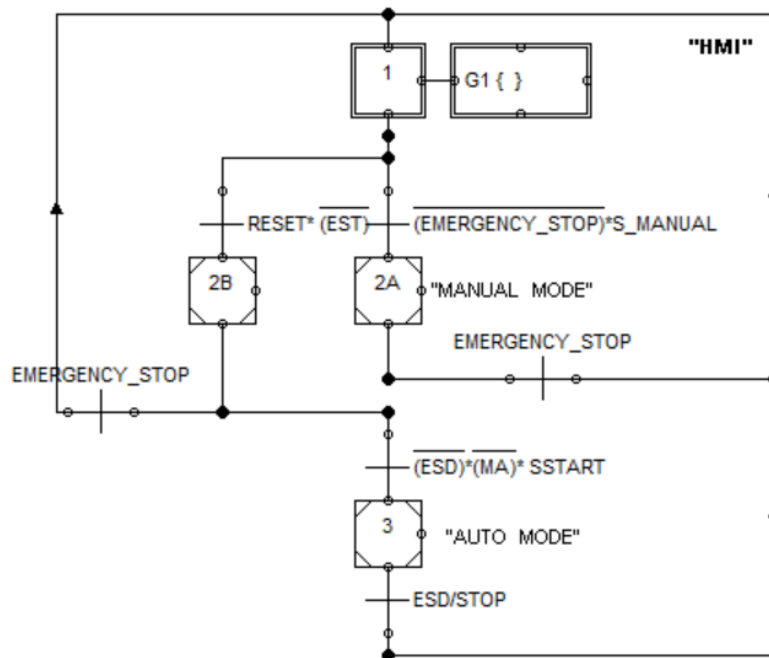


Figure 29: Main control part

The picture above is the mode selection part, which is also the essential part of this program. It contains 4 steps and 6 transitions. In order to make the whole process is under control by the emergency shutdown, each step is connected with the EMERGENCY STOP button. Step 1 is followed by a compulsory command that would freeze all of the GRAFCET running steps.

b. Manual mode:

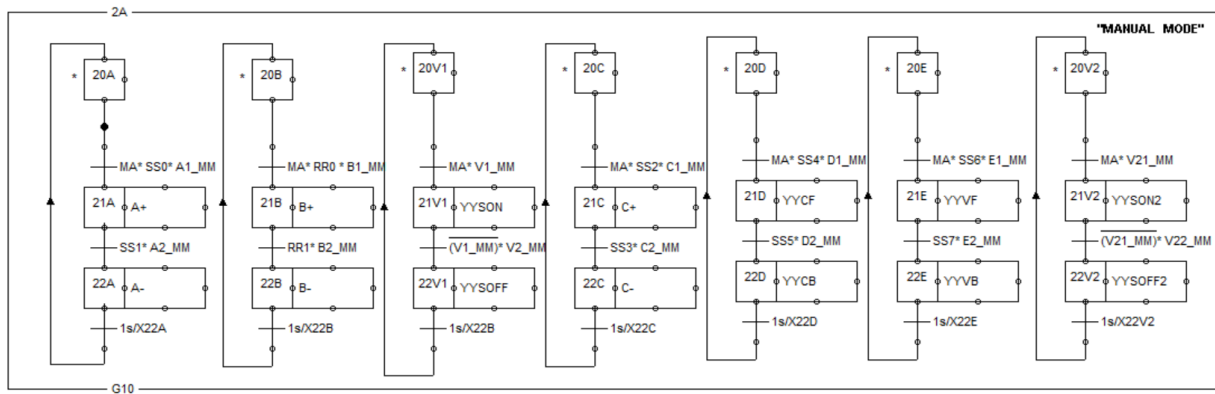


Figure 30: Manual mode

In the pneumatic system, Manual / Automatic mode can be switched by pushing and releasing the button. The logic of switching mode is “Manual” or “Not Manual(Automatic)”. This partial GRAFCET of Manual mode is able to control every single action of each actuator by pressing the corresponding pushbuttons.

c. Reset mode:

Reset mode is for initializing all of the pneumatic actuators before running automatic or after the Emergency stop is pressed.

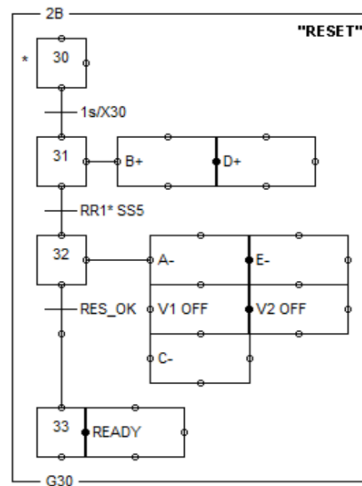


Figure 31: Reset mode

d. Automatic mode:

Automatic mode can be run when every actuator is at the initial position and the M/A mode is in “Not manual”. It has two pushbuttons, one is to start, and the other is to stop. The STOP function in the partial GRAFCET is to stop the loop after the last operation finishes.

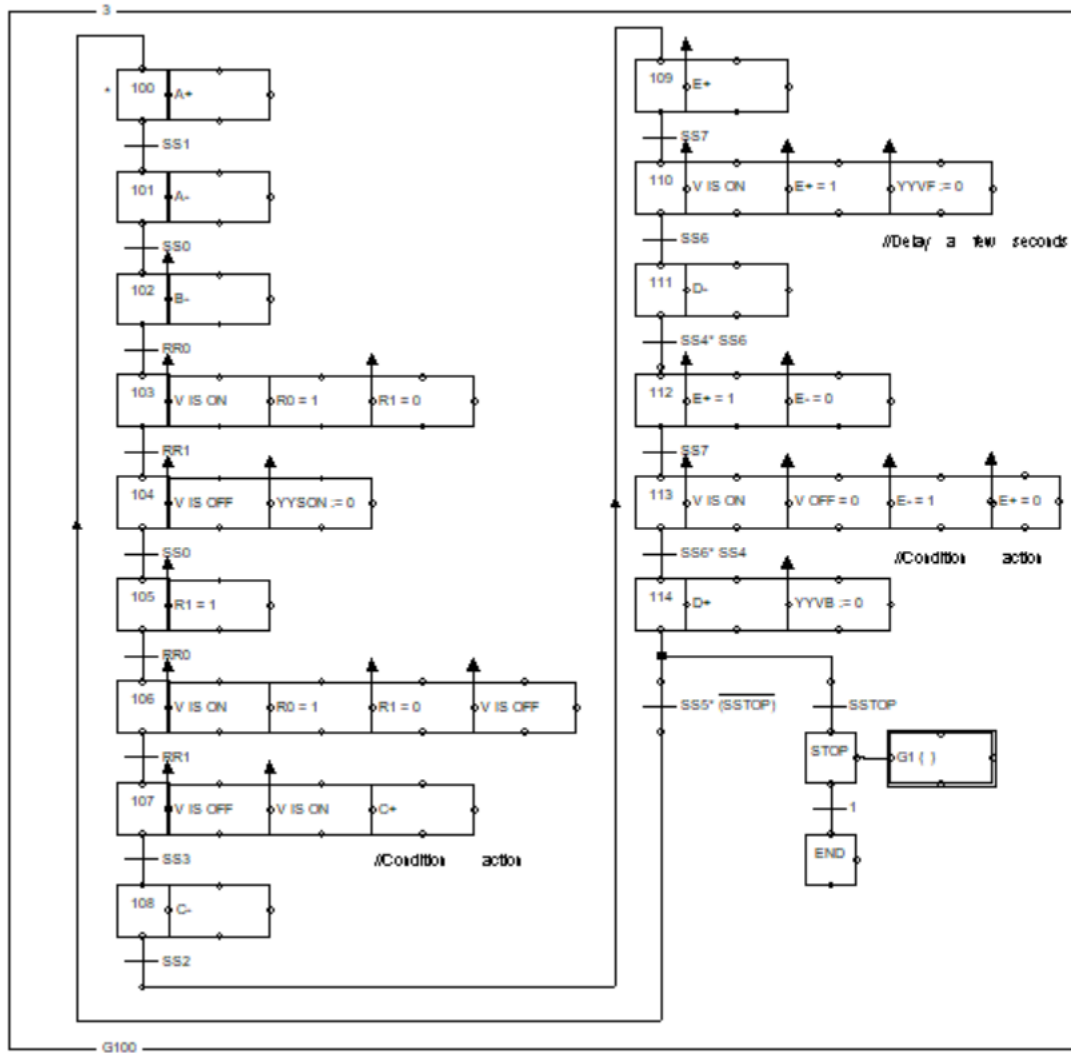


Figure 32: Automatic mode in GRAFCET

5.2. Hardware simulation connection

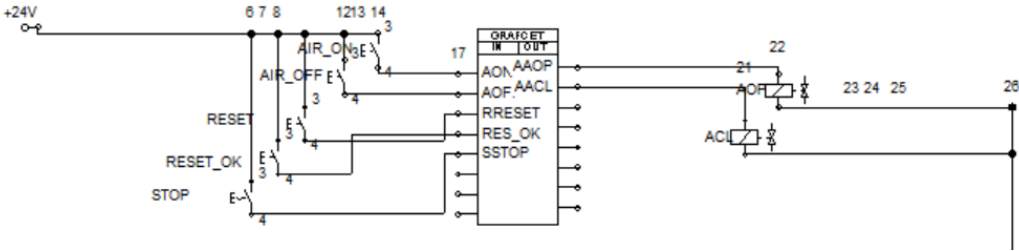


Figure 33: Hardware connection 1 in FluidSIM®

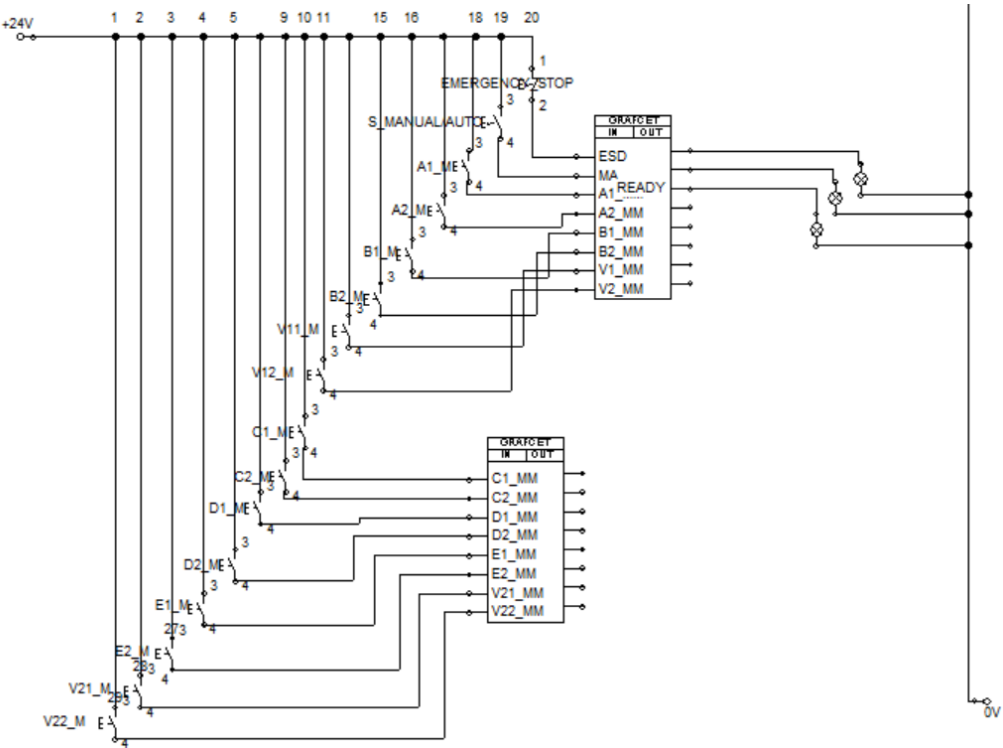


Figure 34: Hardware connection 2 in FluidSIM®

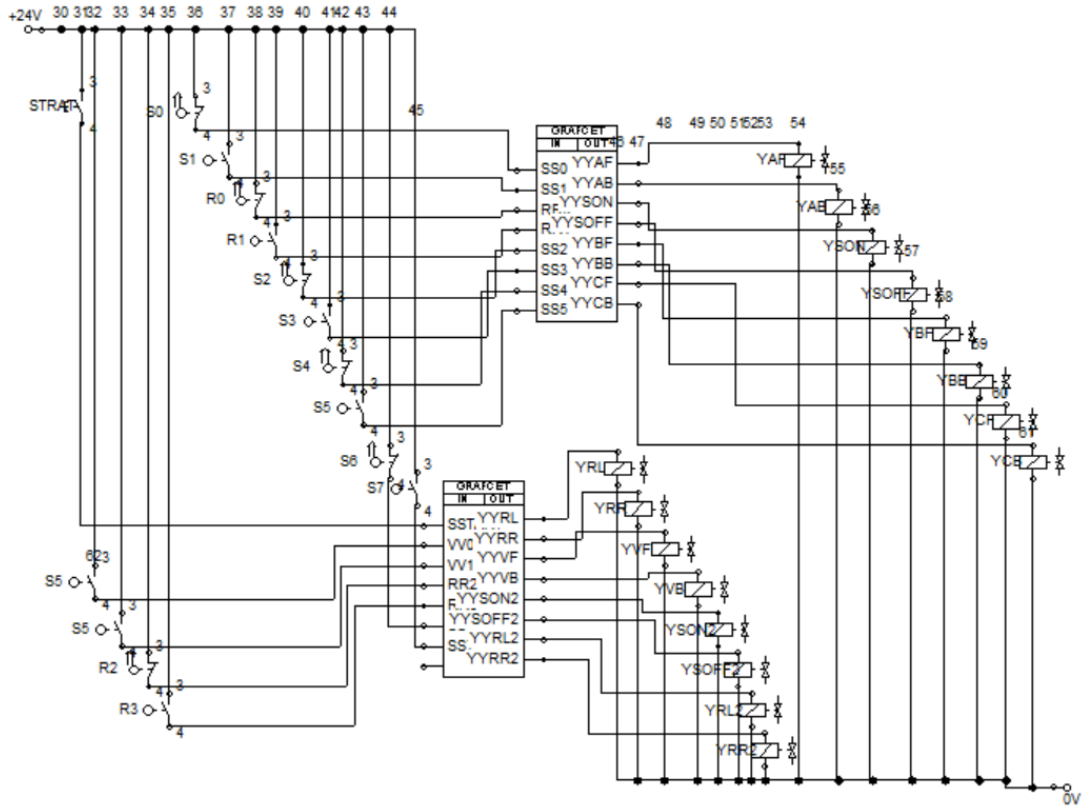


Figure 35: Hardware connection 3 in FluidSIM®

5.3. Ladder program simulation

This section is the preparation for real application in Lab 109. It is necessary to simulate the pneumatic system with FluidSIM® because building the whole manipulation line model is not possible in the real laboratory with a limited number of actuators and hardware restriction. Therefore, realizing all of the designed functions can be done by using FluidSIM® as there are infinite space and available actuators. In order to debug system better, the whole manipulation system turns to two sub-manipulation systems.

Part 1 : A+ A- B- V1 B+ V0

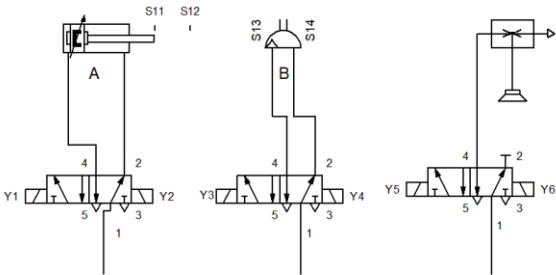


Figure 36: Station 1 in FluidSIM®

Part 2: C+ C- E+ V1 E- D- E+ VO E- D+

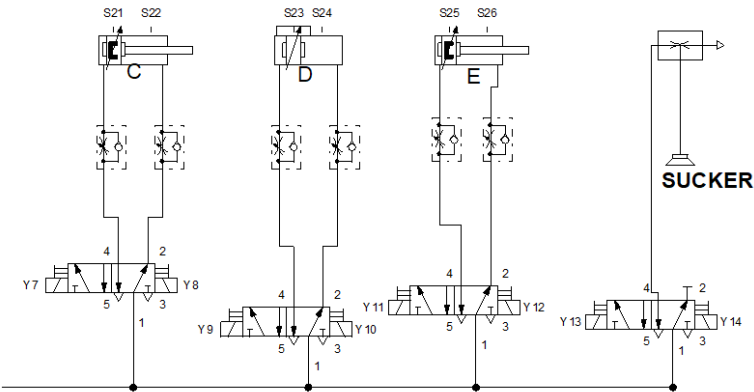


Figure 37: Station 2 in FluidSIM®

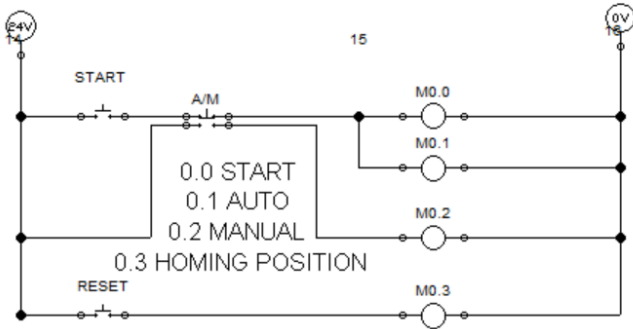


Figure 38: Main control in Ladder Logic

The picture above is mode selection logic in Ladder program. There are four internal bits M0.0 - M0.3 to control states of each mode.

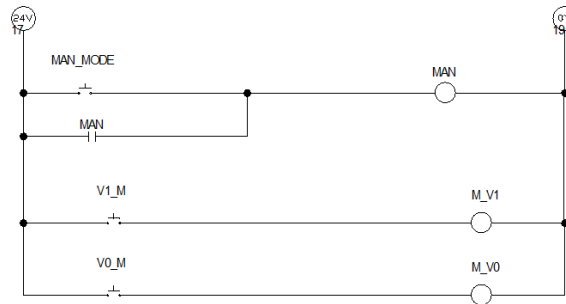


Figure 39: Manual mode in Ladder Logic

Figure 39 is the subprogram for manual mode. To clearly show the manual function simulation and reduce program structure repetition, only manual mode of one vacuum sucker is designed in this ladder program. It is able to apply the same logic to different actuators with manual manipulation function. But it requires more hardware, for instance, an extra PLC I/O module, pushbuttons and more space for placing them.

Variable name	Description
MAN_MODE	Manual mode
MAN	Manual mode internal bit
V1_M	Pushbutton to turn on the vacuum
V0_M	Pushbutton to turn off the vacuum

Table 3: Signals description

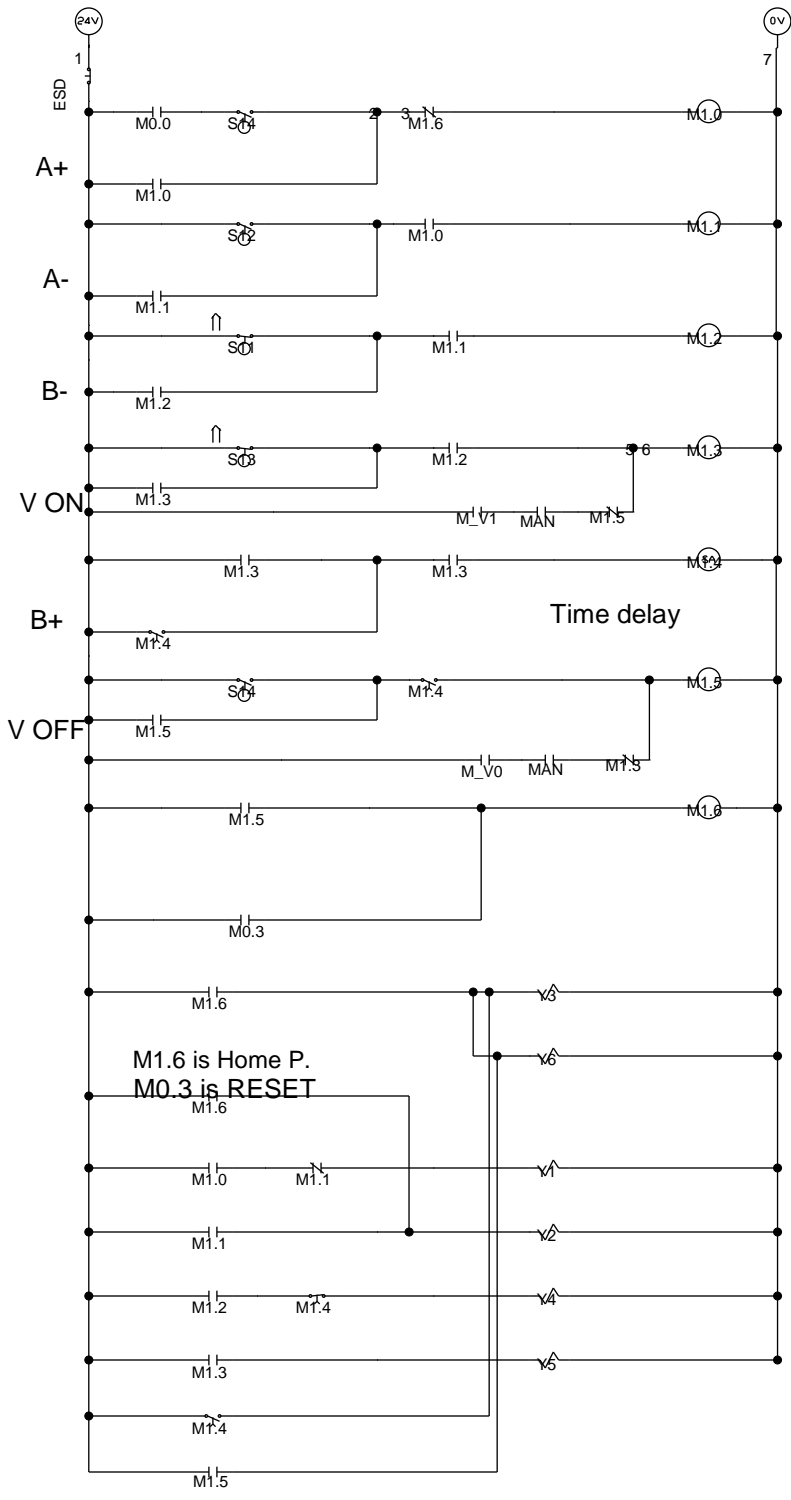


Figure 40: Station 1 main control in Ladder Logic
Home P. - Home position

6 LABORATORY 109 SIMULATION

The whole designed pneumatic manipulation system in the simulation requires loads of pushbuttons, sensors and I/O bits. However, the real application in Lab 109 is not exactly the same as how it is in FluidSIM® simulation, because the available PLCs do not have enough I/O bits. Lab 109 has only one pair of sensors for special actuators, like the rotary drive and the linear drive. Both of these special actuators are implemented in our design. Plus, the functionality of mode selection is not able to be realized in Lab 109. The reason is that the system needs more I/O bits and pushbuttons as mentioned earlier. But every function and mode are simulated successfully in FluidSIM® with both GRAFCET's language and LADDER diagram.

Therefore, due to the restriction of the laboratory space and facilities, the solution for this case is to divide the circuit into two parts. Station 1 and Station 2 both have three actuators, which are six in total. For the second part, the fourth actuator, vacuum control, that is removed because of the limited I/O. But the Station 1 proved that our station can work with vacuum technology.

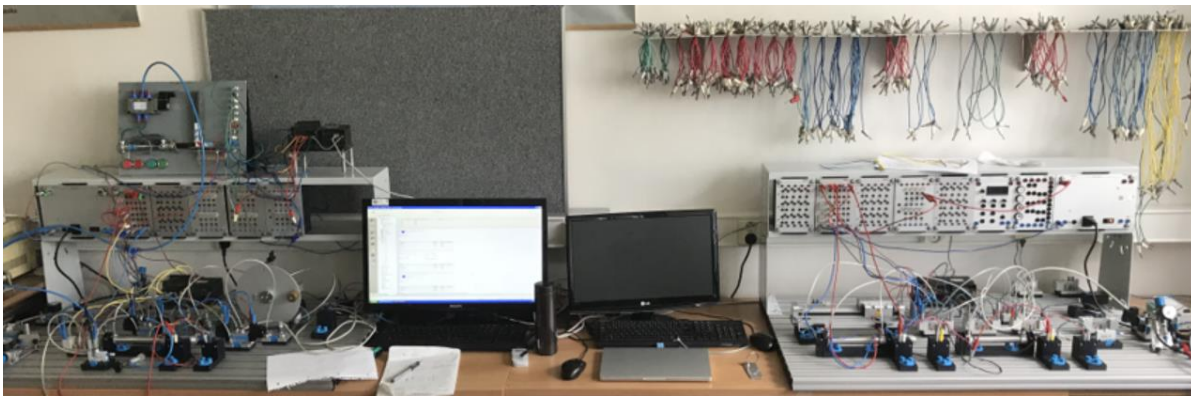


Figure 41: Station 1 (Left) and Station 2 (Right) in Lab 109

6.1. PLC selection

Different types of S7-200 series PLCs and modules available in the Lab 109.

Catalog	Feature	Order number	Description
Controller	CPU 222	212-1AB23-OXB0	2 nd generation 8 Inputs/6 Outputs
Controller	CPU 215	215-2AD00-OXB0	1 st generation Profibus DP
Extension module	EM 223	223-1PL22-0XA0	16 Inputs/16 Outputs
Communication module	CP 243-1	6GK7243-1EX01-0XE0	Industry Ethernet

Table 4: Data of PLCs and modules in the Lab



Figure 42: Extension module

6.1.1. S7-200 series

The S7-200 series of micro-programmable logic controllers (Micro PLCs) can control a wide variety of devices to support automation needs. The S7-200 monitors inputs and changes outputs

as controlled by the user program, which can include Boolean logic, counting, timing, complex math operations, and communications with other intelligent devices. The compact design, flexible configuration, and powerful instruction set combine to make the S7-200 a perfect solution for controlling a wide variety of applications [7].

With CPU 222, 8 inputs and 6 outputs are available. It can connect with two extension modules having maximum 78 digital bits or 10 analog bits. CPU 222 has four high-speed counters, PID controller, and an RS485 port [7].

6.1.2. CP 243-1 Communications processor for Industrial Ethernet



Figure 43: Ethernet module CP243-1

The CP 243-1 is a communications processor designed for operation in an S7-200 automation system. It is used for connecting an S7-200 system to Industrial Ethernet (IE). The CP 243-1 also facilitates communication via Ethernet for the S7 product family. As a result, S7-200 can be remotely configured, programmed, and diagnosed via Ethernet using STEP 7 Micro/WIN. Moreover, an S7-200 can communicate with another S7-200, S7-300, or S7-400 controller via Ethernet. It can also communicate with an OPC server.

In the open SIMATIC NET communications system, Industrial Ethernet is the network for both the coordination level and the cell level. Technically, the Industrial Ethernet is an electrical network built on the basis of a shielded coaxial cable, twisted pair cabling, or an optical network

using a fiber optic cable. Industrial Ethernet is defined by the international standard IEEE 802.3. Only one CP 243-1 may be connected to an S7-200 CPU. If additional CP 243-1 communications processors are connected, the S7-200 system may not function correctly [8].

6.2. I/O table

The following pictures are the I/O tables of two stations. It provides detailed information related to the controlled variables, including their symbols, addresses, data types, and comments. The Station 1 has 6 inputs, 6 outputs, and 11 internal bits. Station 2 has 8 inputs, 6 outputs, and 16 internal bits.

Symbol	Address	Data Type	Comment
FEEDER_EXT	%I0.4	BOOL	A+ EXTENDED SENSOR
FEEDER_RET	%I0.5	BOOL	A- RETRACTED SENSOR
ROTARY_LEFT	%I0.6	BOOL	B- SENSOR
ROTARY_RIGHT	%I0.7	BOOL	B+ SENSOR
EXT_FEEDER	%Q0.0	BOOL	EXTEND FEEDER
RET_FEEDER	%Q0.1	BOOL	RETRACT FEEDER
LEFT_ROTARY	%Q0.2	BOOL	SWING ROTARY ARM TO L
RIGHT_ROTARY	%Q0.3	BOOL	SWING ROTARY ARM TO R
ON_V1	%Q0.4	BOOL	TURN ON THE VACUUM
OFF_V1	%Q0.5	BOOL	TURN OFF THE VACUUM
A_PLUS_IB	%M1.0	BOOL	A+
A_MINUS_IB	%M1.1	BOOL	A-
B_MINUS_IB	%M1.2	BOOL	B-
V1_ON_IB	%M1.3	BOOL	V_PART1_ON
B_PLUS_IB	%M1.4	BOOL	B+
V1_OFF_IB	%M1.5	BOOL	V_PART1_OFF
HOMING_COIL1	%M1.6	BOOL	INITIALIZE ACTION
START_PB	%I0.0	BOOL	
RESET_PB	%I0.3	BOOL	
SWITCHING1_2	%M5.0	BOOL	
CUTOFFALL	%M5.1	BOOL	
START_P1	%M6.0	BOOL	
RESET_P1	%M6.1	BOOL	

Table 5: I/O table for Station 1

Symbol	Address	Data Type	Comment
C_PLUS_IB	%M1.7	BOOL	C+
C_MINUS_IB	%M2.0	BOOL	C-
E_PLUS_IB	%M2.1	BOOL	E+
V2_ON_IB	%M2.2	BOOL	V2 IS ON
E_MINUS_IB	%M2.3	BOOL	E-
D_MINUS_IB	%M2.4	BOOL	D-
E_2PLUS_IB	%M2.5	BOOL	E+ SECOND TIME
V2_OFF_IB	%M2.6	BOOL	V2 IS OFF
E_2MINUS_IB	%M2.7	BOOL	E- SECOND TIME
D_PLUS_IB	%M3.0	BOOL	D+
HOMING_COIL2	%M3.1	BOOL	HOMING POSITON
C_EXT	%I0.0	BOOL	C IS EXTENDED
C_RET	%I0.1	BOOL	
D_EXT	%I0.2	BOOL	
D_RET	%I0.3	BOOL	
E_EXT	%I0.4	BOOL	
E_RET	%I0.5	BOOL	
EXT_C	%Q0.0	BOOL	
RET_C	%Q0.1	BOOL	
EXT_D	%Q0.2	BOOL	
RET_D	%Q0.3	BOOL	
EXT_E	%Q0.4	BOOL	
RET_E	%Q0.5	BOOL	
ON_V2	%Q0.6	BOOL	
OFF_V2	%Q0.7	BOOL	
START_PB2	%I0.6	BOOL	
RESET_PB2	%I0.7	BOOL	
START_SCADA	%M4.0	BOOL	
RESET_SCADA	%M4.1	BOOL	
SWITCHING1_2	%M5.0	BOOL	
RESET_CT	%M5.1	BOOL	RESET COUNTER
NUM_CT	%MW0	INT	COUNTER TIMES

Table 6: I/O table for Station 2

6.3. Hardware application of Station 1

In Station 1, two pushbuttons are used in the hardware application. One normal pushbutton is to RESET/INITIALIZE the system. Another one is to START/STOP the system. The START/STOP pushbutton used would stay at its position when it is pushed. In order to turn it OFF, operators need to push it again when it is ON.

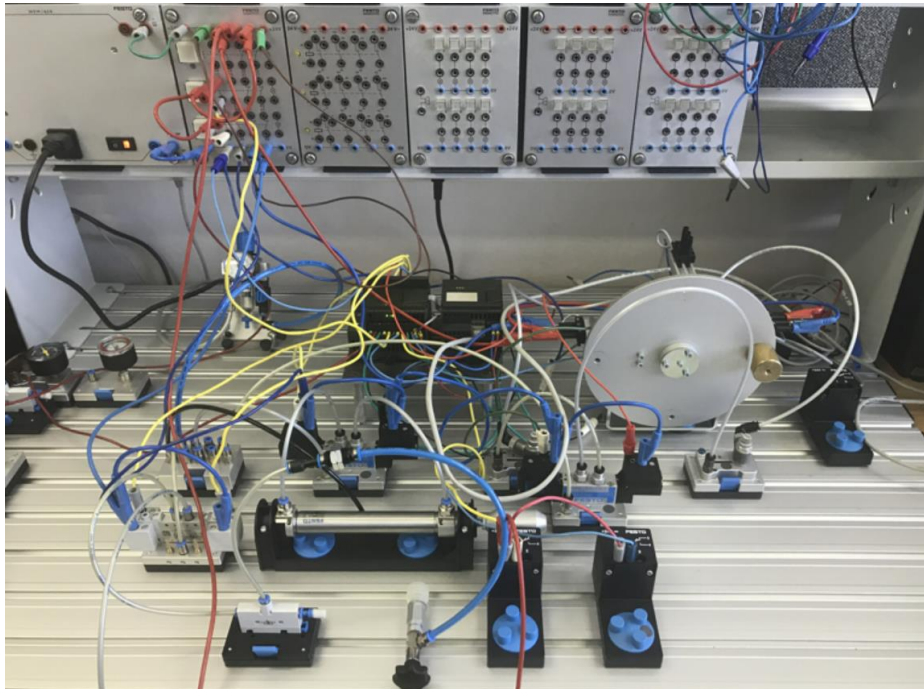


Figure 44: Station 1 in Lab 109

From the picture above we can see that there are three pneumatic actuators. The semi-rotary drive and the vacuum sucker are working together as a catch system. The double acting cylinder is the feeder pushing working pieces from its source. They are all controlled by bistable valves. PLC outputs are connected with three valves. Four sensors and two pushbuttons are inputs. Two flow control valves are used to slow down the speed of actuators in order to see the movements clearly.

6.4. Hardware application of Station 2

Station 2 has the same control logic panel as Station 1. One pushbutton is for RESET, the other one is to START/STOP. One permanent broken output O0.3 is not used in Station 2 so that the retraction action of the linear drive needs to be carried out manually.

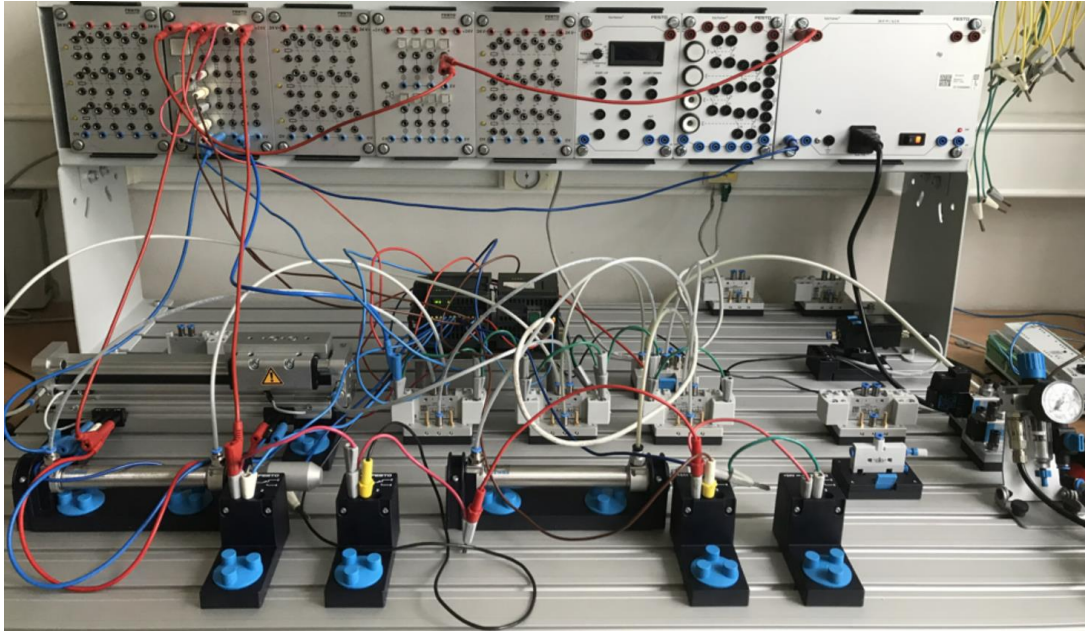


Figure 45: Station 2 in Lab 109

Due to the shortage of PLC output pins, the vacuum sucker is not used in Station 2. Like Station 1, three pneumatic actuators are presented. The double acting cylinder on the right side should be fixed on the carriage of the linear drive. If the vacuum sucker was able to apply, these three actuators will be seen as a whole working together to catch and transport working pieces. The other double acting cylinder is to move the piece from Station 1 to Station 2.

In this Lab 109, it is not capable of constructing these actuators in its ideal condition because of all kinds of limitations. Therefore, the application only achieved its pre-defined moving sequences.

7 VISUALIZATION OF THE SYSTEM

7.1. OPC server and clients

OPC stands for Object linking and embedding for process control based on clients and server technology. The Client allows the user to request read/write command to an OPC server, and then from OPC server, the request is translated to device protocol that the machine would understand.

The purpose of OPC is to standardize a real-time communication between different technologies. The data from OPC are completely universal because a plant or a control system in the real industry will involve a variety of hardware and software. In order to translate those different languages, OPC is introduced. Basically, the function of OPC is to transfer data block into tags. Once the data is converted to OPC, every application can start speaking with other applications, for example, SCADA system reads data from PLC.

7.2. Ethernet wizard and PC Access

Ethernet wizard in STEP 7- Micro/WIN helps users to connect the PLC with Ethernet. The communication protocol is based on ISO and TCP/IP. Ethernet wizard has to be set correctly because it is the bridge between PLC and PC Access (OPC server) communication.

When the setting of Ethernet wizard is done, a subroutine called “ETHO_CTRL” will be generated automatically. This “ETHO_CTRL” is a subroutine to initialize and control the program, which would check Ethernet module before starting it. One program should have only one “ETHO_CTRL” subroutine.

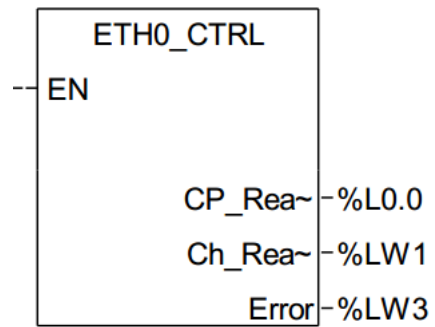


Figure 46: Ethernet control block

PC Access is the OPC server software that we used in the design. It serves, especially for S7-200 series PLCs. PC Access offers a few ways of establishing communication between PC and the PLC, including PC/PPI, Ethernet, and Modem.

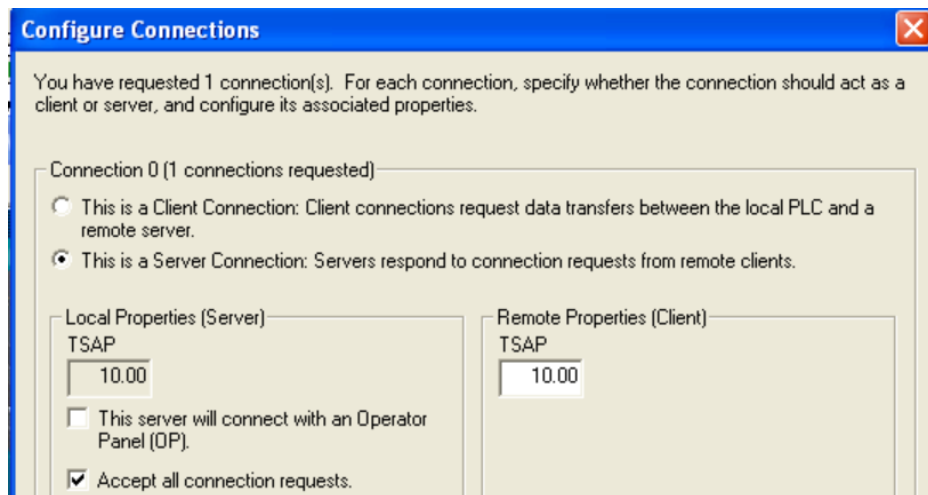


Figure 47: Ethernet wizard in Micro/Win

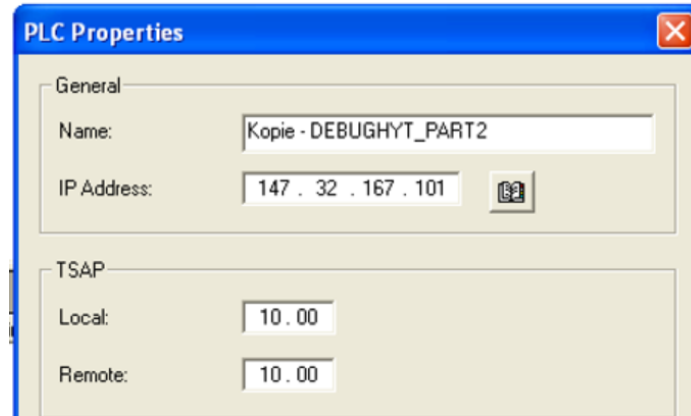


Figure 48: Setting in PC ACCESS

It is important to set the IP address of the PLC and its TSAP (Transport Service Access Point), otherwise, PC Access cannot communicate between PC and the PLC. TSAP is a part of a network address which identifies which application or module is sending or receiving data. The TSAP must be identical with TSAP in STEP 7- Micro/WIN Ethernet wizard. TSAP has the form xx.xx where every x should be substituted by a number. The first two numbers represent the communication number, and the last two numbers tell the location of the module. The default value of TSAP is 10.00.

The picture below showed the screen of PC Access for the second part manipulation system. It has information about the name of PLC in PC Access, variable folders and imported tags from the real PLC program.

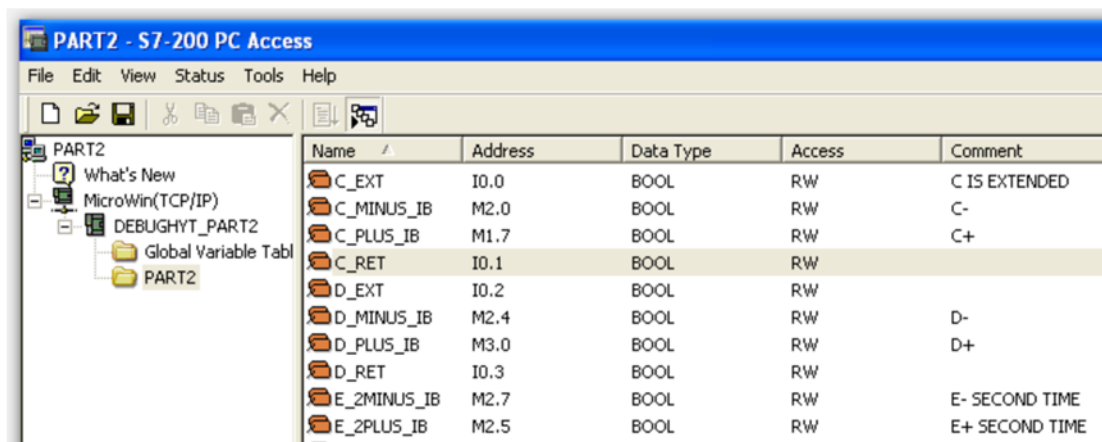


Figure 49: Variables in PC ACCESS

By dragging variables from the list, if the “quality” is good, the value of the tags can be monitored after running test client. In this Figure 50: Test client in PC ACCESS, real-time states are displaying on the screen.

Test Client			
Item ID	Value	Time Stamp	Quality
MicroWin.DEBUGHYT_PART2.PART2.C_EXT	0	15:53:11:546	Good
MicroWin.DEBUGHYT_PART2.PART2.C_RET	1	15:53:11:546	Good

Figure 50: Test client in PC ACCESS

7.3. Communication with Reliance SCADA

After connecting PLC to PC Access, the next step is to select the established server in Reliance SCADA system. In the system, users can add new devices in the device manager. Once the S7-200 OPC server is added, it allows project in Reliance SCADA system to import tags from the server as it is shown in Figure 51.

OPC Device		OPC Server					
Name	Type	Name	Value	Type	R/W	ID/Comment	
OPC1		S7200.OPCServer		OPC ...			
Tags		MicroWin					
Group1	500	DEBUGHYT_PART2					
		Global Variable Table					
		PART2					
		C_EXT	False	Bool	RW	C IS EXTENDED ...	
		C_MINUS_IB	False	Bool	RW	C-	
		C_PLUS_IB	False	Bool	RW	C+	
		C_RET	True	Bool	RW		
		D_EXT	True	Bool	RW		
		D_MINUS_IB	False	Bool	RW	D-	
		D_PLUS_IB	False	Bool	RW	D+	
		D_RET	False	Bool	RW		

Figure 51: List of tags from the real PLC

The key point of establishing the communication is to link these real-time tags with elements in Reliance visualization project. The state of each tag is equivalent to its corresponding variable in STEP7-Micro/WIN, and tags from PC Access will be refreshed at a certain rate. Reading and writing tags through Reliance SCADA is possible, which means users can alter the states of tags in runtime mode. But this function can be achieved only when the PLC program has provided the

control logic, controlled by real push buttons and by PLC internal bits that are linked with Reliance SCADA system.

7.4. Data exchange possibilities between controllers

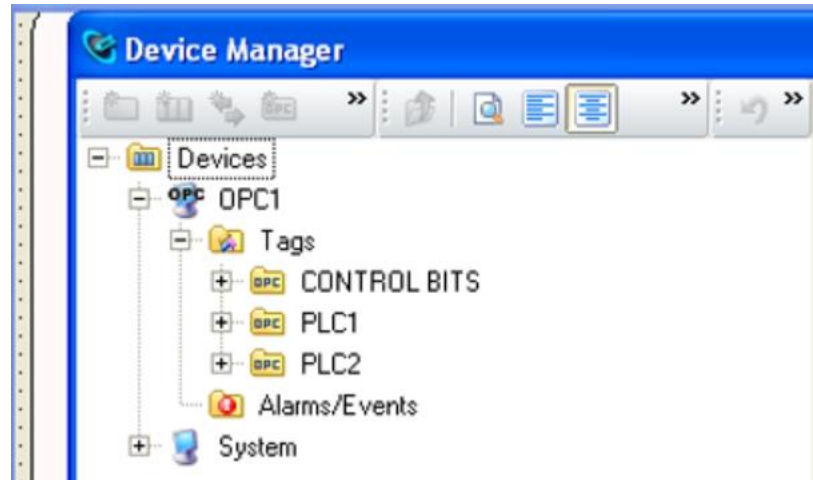


Figure 52: Device manager in Reliance 4

The screenshot above shows that there are three groups of tags in the folder tree. PLC1 and PLC2 groups have tags related to inputs and outputs variables, and CONTROL BITS group contains internal bits mostly that involved with both programs from two PLCs, for example, the memory bit for switching stations. Since the two used PLCs do not know each other, the way of sending signals to each other is through a common platform - Reliance SCADA system. There are hardware and software solutions for our application.

7.4.1. Script method

In the script manager of Reliance, reading and writing value to PLC can be done with a script. The logic of exchanging state is that when Station 1 is complete, an ending internal bit will be set to 1. The state of the bit will be read by SCADA with help from OPC server. The next step is to set starting signal of Station 2 by writing a script.

Where Station 1SWTICHING1_2 is the ending signal of the Station 1, part2SWITCHING1_2 is the starting signal of the Station 2.

7.4.2. Event method

The other method to realize the goal is to create an Event. In the condition of the event, there are different options to choose how the event will be activated, including any value change, leading edge, etc. The interface for creating events is shown in the following figure.

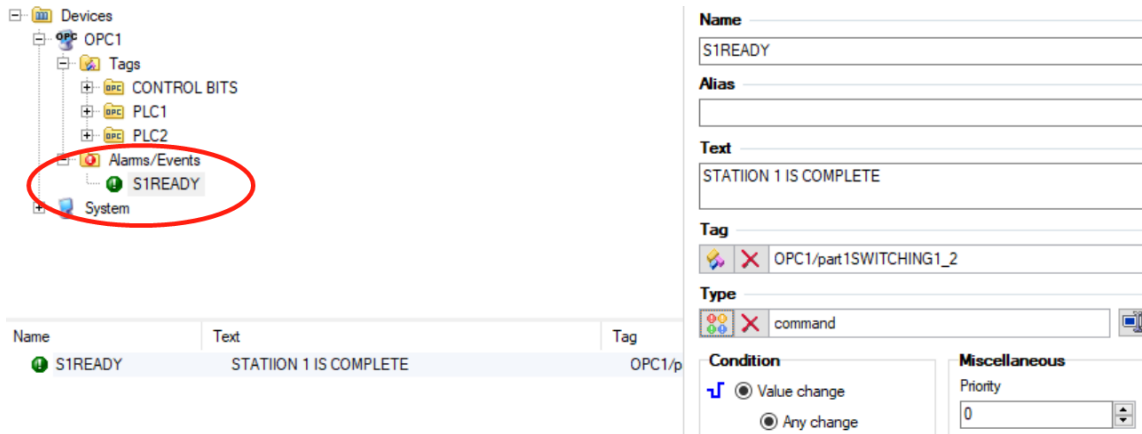


Figure 53: Creating events in Reliance 4

In the Advanced tab, another tag for exchanging data needs to be linked. Therefore, the event will be turned on when the first tag value changes in its pre-defined way. And then the related tag will also be turned on.

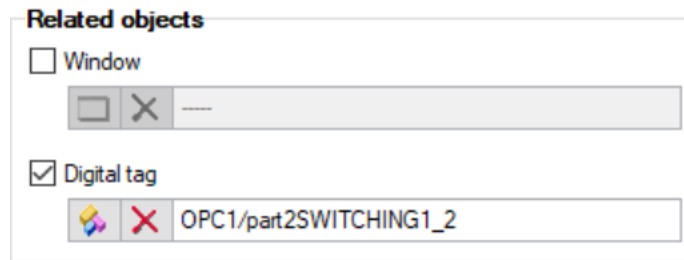


Figure 54: Event-related tag

7.4.3. Hardware method

The previous two solutions are based on software, which means the OPC server and the Reliance SCADA system must be working when we want to achieve the data exchange. However, if the manipulation system does not have an available OPC server or a SCADA system, there is an

alternative way to realize it. Interconnecting two PLCs would be able to solve the same problem, which needs to use an output pin of the first PLC can be connected to an input pin of the second PLC by the electrical wire.

7.5. Visualization design in Reliance

Two projects in Reliance SCADA systems are designed for the application. One is linked to the real tags from PLCs, the other one is running with scripts.

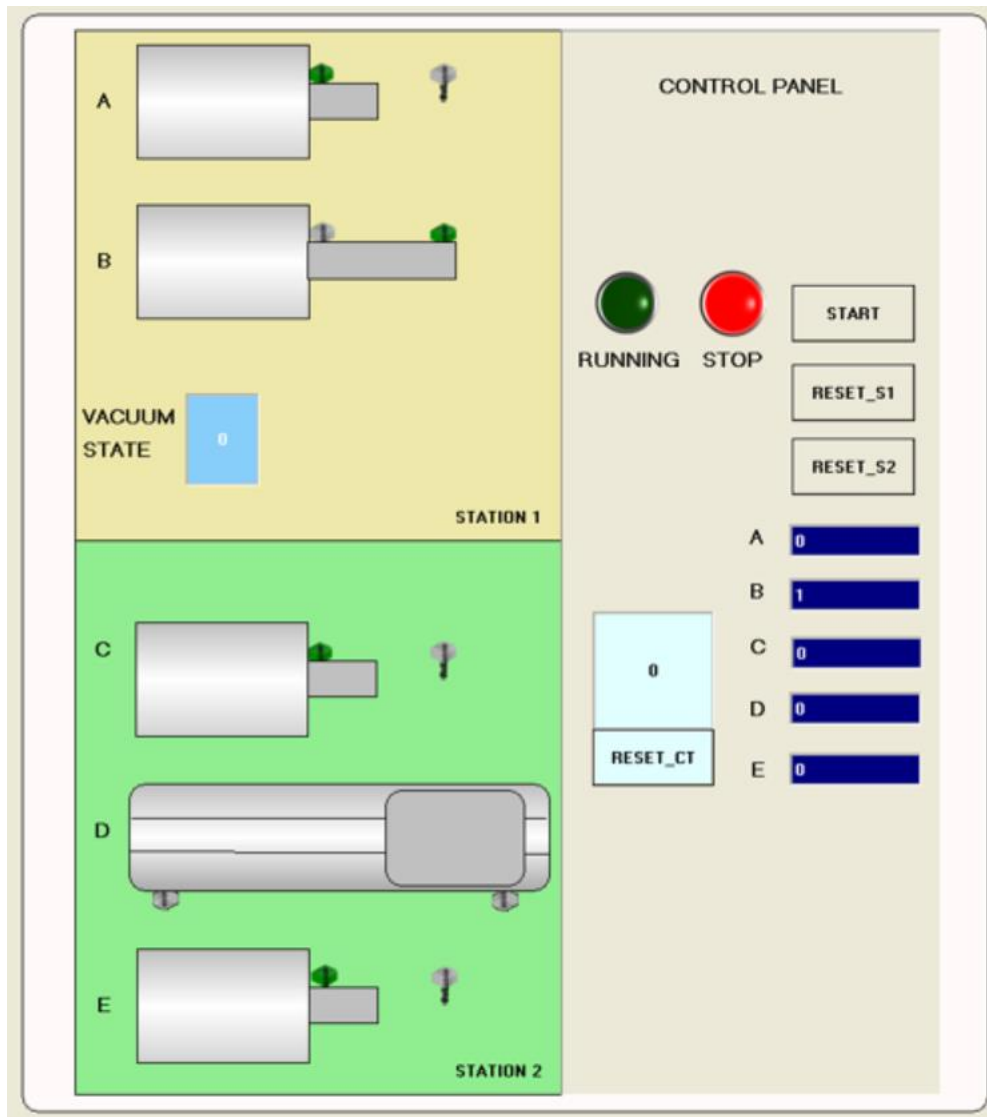


Figure 55: Reliance SCADA system of the process with top view

The project in Reliance SCADA system in Figure 55 contains a control panel and two stations that are represented in the different background color. Each station has three pneumatic actuators with sensors that would indicate the current position of the corresponding actuator. The state of those graphical actuators would change along with the value of linked tags. The state of vacuum generator is identified with a square display. If the vacuum generator is on, the number being displayed will be set to one.

In the control panel, two indicators are placed next to the start button showing if the station is running. Three push buttons stand for START, RESET STATION 1 and RESET STATION 2 respectively. In case of the graphical actuators malfunction, five rectangle displays are able to show the state of the actuator's state numerically.

The Station 1 has another SCADA design with front view, which looks closer to the real application. However, it is rather time-consuming to design such visualization and it is not capable of displaying the whole application with many actuators within one monitor. Instead of connecting with PC Access, the Reliance SCADA system in Figure 56 is running based on the scripts, where the value of each tag is assigned accordingly.

The models of actuators are drawn by using AutoCAD so that more elements and shapes can be used. The graphical models would have better looking because they are imported pictures. It is possible to create fancy graphical models and apply it to Reliance 4. By contrast, the system shown in Figure 56 has used the only combination of vectors (Rectangles and cycles).

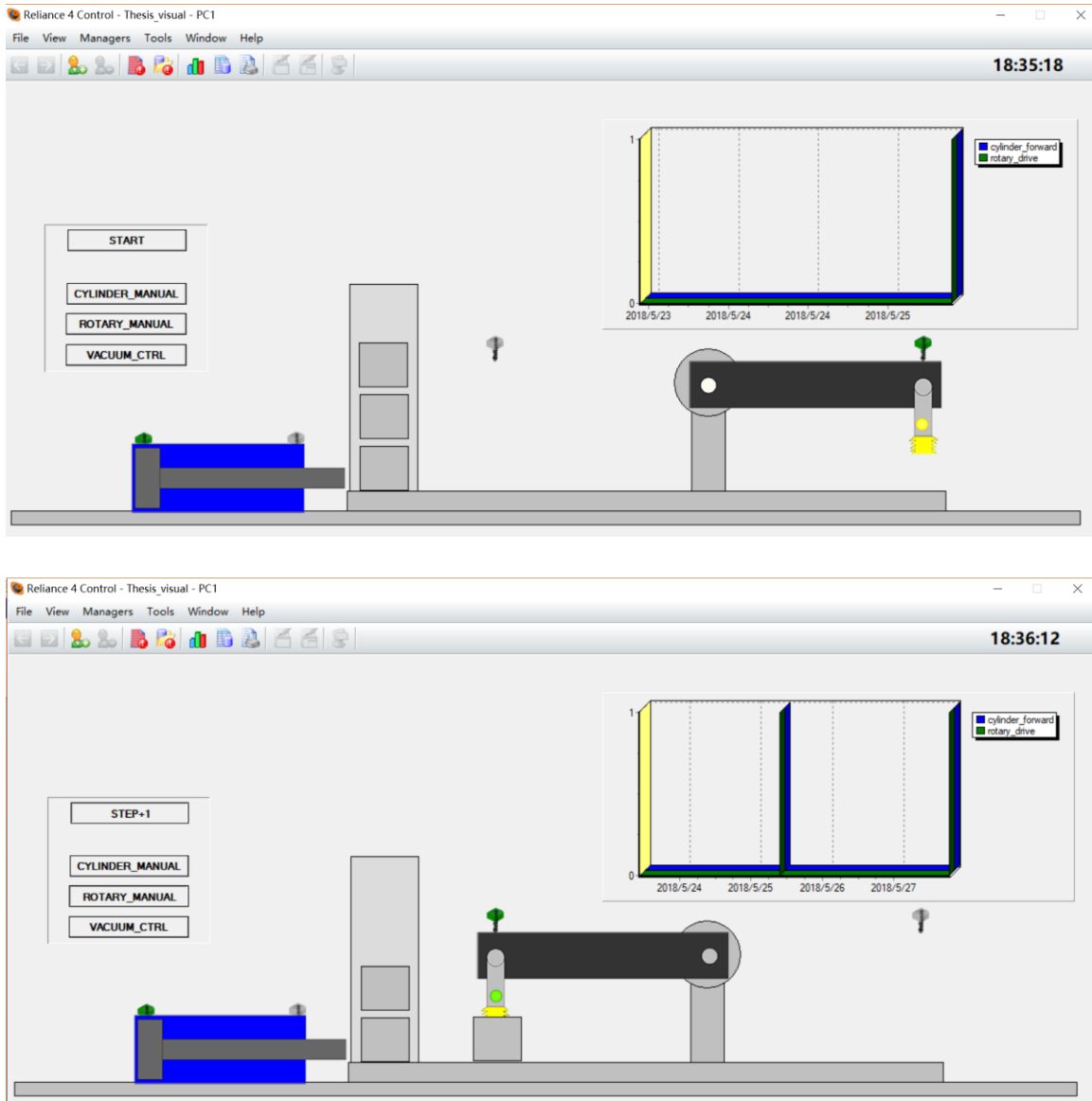


Figure 56: Runtime window for the Station 1 from the SCADA Reliance project

7.6. Runtime window of the project

Runtime mode is a window or windows where users can interact with the visualization project. When project design is complete, runtime mode is able to test the system real-time response. In runtime mode, settings and size of each element are no longer changeable.

When the project is in runtime mode, the real-time window enables the following functions:

- Movements observation of actuators
- State updates of actuators
- Interacting with pushbuttons and switches
- Alarms/Events check (Current & historical)
- View trends and reports
- User login

It is dangerous that a Reliance SCADA system is accessible to everyone. Therefore, assigning rights to different level users is necessary. Rights assignment can be done in user manager when it is in design mode. Commanding, acknowledging alarms are able to be secured by not assigning rights to operators. In this case, issuing commands requires users to login in runtime mode.

8 CONCLUSION

According to the guidelines of Master's thesis, the sequential operations have been designed and pneumatic actuators with different functions are chosen properly, including a semi-rotary drive and a linear drive. Vacuum technology is integrated with both types of special drives. The idea was inspired by FESTO MPS in Lab 109 and a FESTO production line from the Internet [9]. The designed sequential operations have been fulfilled successfully. The introduction of special drives and other components can be found in Chapter 4.

The second task is to simulate the system in FluidSIM®. The whole process is simulated with both GRAFCET and LAD programming languages. The simulation contains reset function, automatic mode, manual mode, etc. Mode selection is done by using enclosing steps and partial GRAFCETs. Emergency shutdown is linked to each mode making sure that the stations can be stopped whenever needed. The control algorithm of the simulation is introduced in Chapter 5.1.

The third task requires implementation of the design. Therefore it is necessary to transfer the LAD simulation in Micro/win and run the program in real PLCs. Implementation of the process into real PLCs differs from how it is in the simulation. The LAD programs have been improved while debugging the whole process in Lab 109. The detailed programs of both stations are shown in Appendix A.

The fourth task is to design the visualization by using Reliance 4 and test it in runtime mode. To do that, establishing communications is a crucial step. The visualization of the process would not be observable without a good communication between devices and OPC server (PC Access). Since two stations are not controlled by the same PLC, SCADA system (Reliance 4) provided the possibility of exchanging data. Data exchanging is described in Chapter 7.4.

Visualization design has been done in two ways. One method can display the actual movements by linking real tags from PC Access. The other way of visualizing is that the scripts with controlling algorithm control the states of actuators. The result of visualization in runtime mode is superb with dynamic updates about states of pneumatic actuators, control panel, working

pieces counting block and so on. Assigning users with different rights has been done too, as it is important to the industrial safety.

The fifth task is to debug the whole system in the real time. It is fulfilled without any problem since the PLC program is running very well. The designed SCADA system is able to visualize all the components and issue commands. The interface is introduced in Chapter 7.5.

There are extra works or bonus in the thesis and experiments. For example, to control all of the Inputs/Outputs, it is easier to use only one PLC with an extension module. However, establishing communication between two PLCs is a challenge. I have learned many things and hands-on skills while working on the thesis.

In the future, the topic of the possible next thesis can be extended because more functions can be added to the Reliance SCADA system, including data collecting, operation rights assignment, real-time trends, etc. The pictures and actuator models used in Reliance SCADA system could be improved.

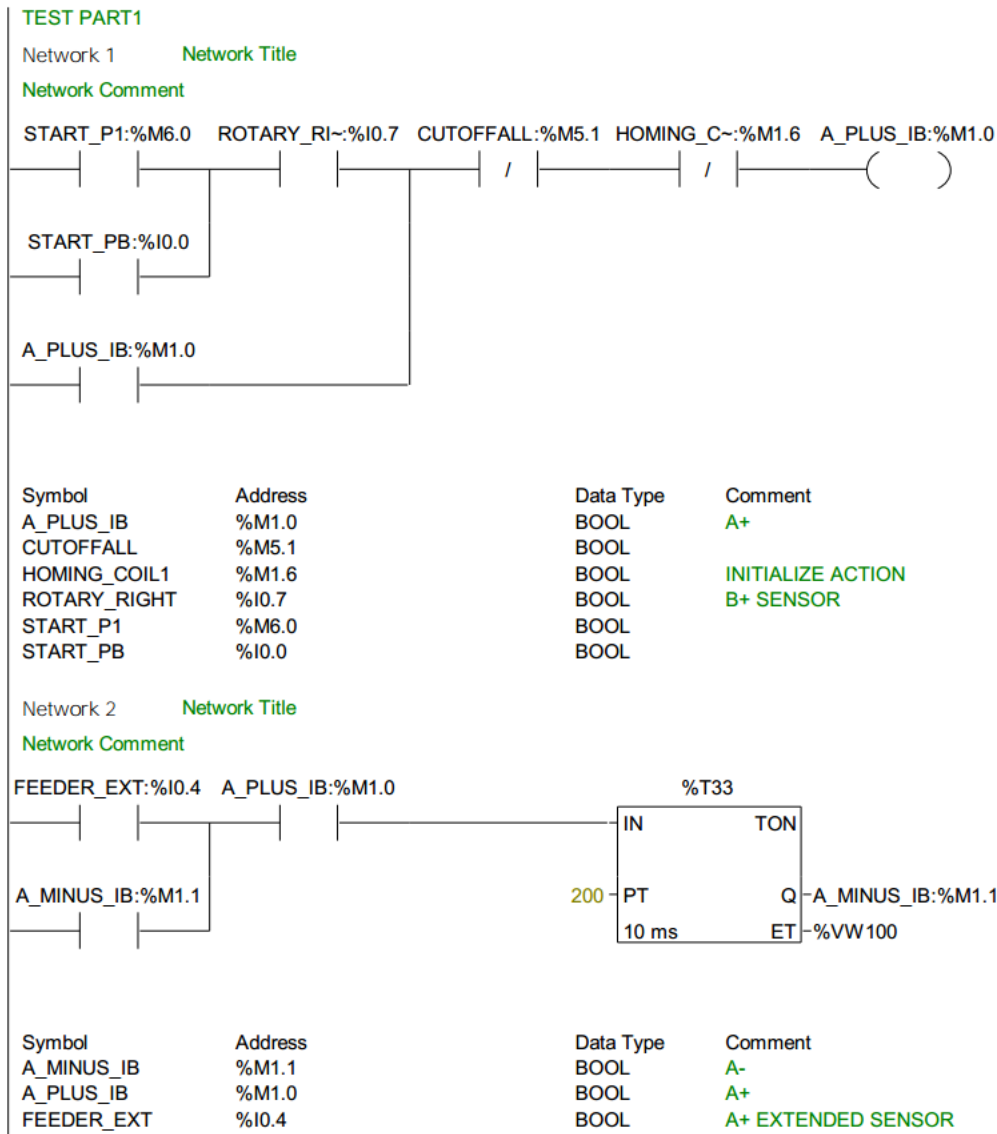
REFERENCES

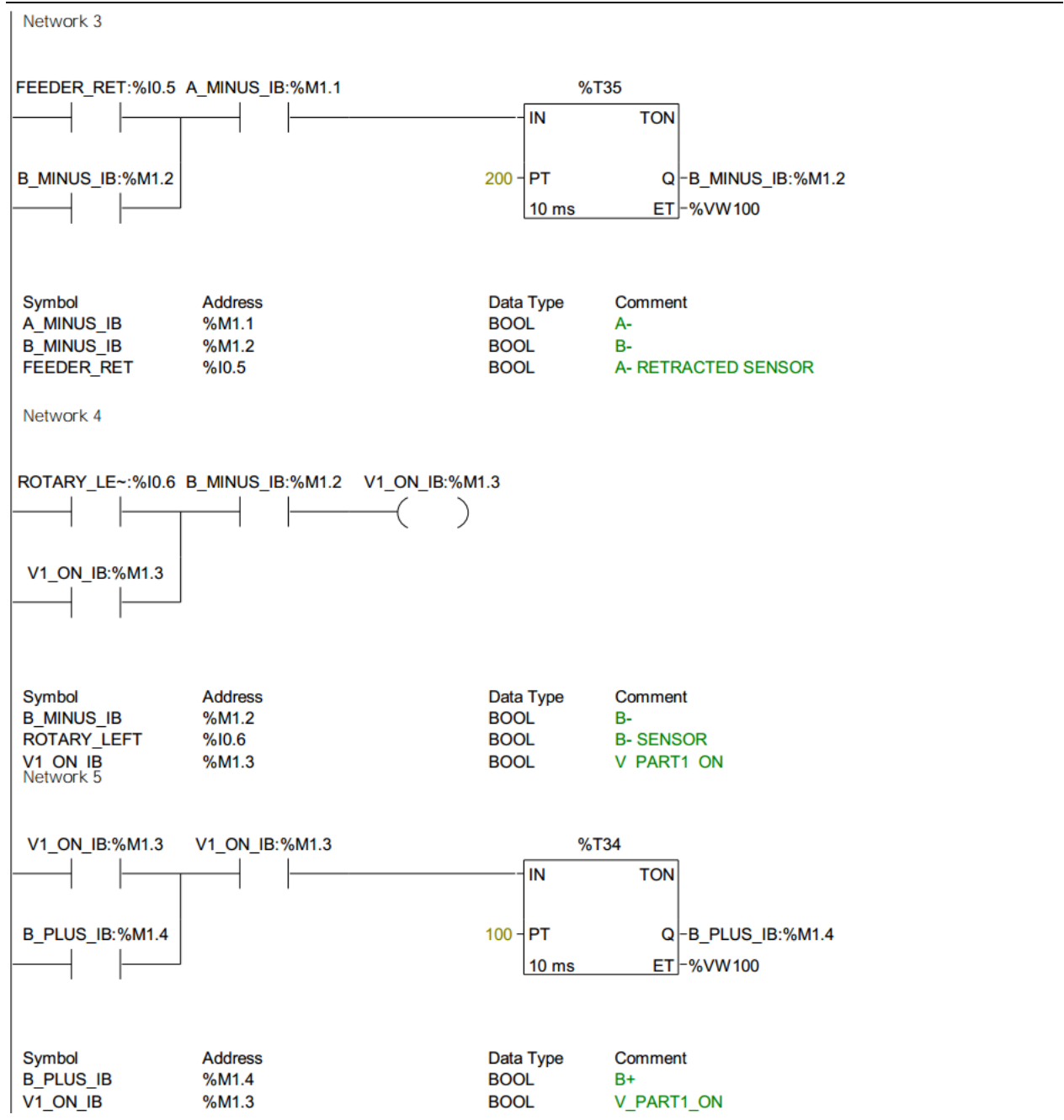
- [1] BEATER, Peter. Pneumatic drives: system design, modeling and control. Berlin: Springer, ©2010. xiv, 323 s.
- [2] Bailey, David, and E. Wright. Practical SCADA for Industry. 2003.
- [3] Reliance 4 SCADA [Online]. Available at: <https://www.reliance-scada.com/en/products/reliance4-scada-hmi-system#page=structure>
- [4] SIMATIC Programming with STEP 7- Micro/WIN manual, SIEMENS [Online]. Available at: https://cache.industry.siemens.com/dl/files/056/18652056/att_70829/v1/S7prv54_e.pdf
- [5] Baker A D, Johnson T L, Kerpelman D I, et al. GRAFCET and SFC as Factory Automation Standards Advantages and Limitations[C]// American Control Conference. IEEE, 1987:1725-1730.
- [6] FluidSIM@ 4 Pneumatics user's guide, Festo [Online]. Available at: <http://www.festo-didactic.com/ov3/media/customers/1100/00422240001150875008.pdf>
- [7] S7-200 programmable controller system manual, SIEMENS [Online]. Available at: http://www1.siemens.cz/ad/current/content/data_files/automatizacni_systemy/mikrosystemy/simatic_s7200/manual_s7_200_2005_en.pdf
- [8] CP 243-1 communications processor for Industrial Ethernet and information technology Operating Instructions, SIEMENS [Online]. Available at: https://cache.industry.siemens.com/dl/files/427/42122427/att_57098/v1/BA_CP-243-1_76.pdf
- [9] Padami. FESTO production line. 2014 [Online]. Available at: <https://www.youtube.com/watch?v=Y7T993UIM90&list=LL9vp7CHxzhINB3SapFuwNHw&index=108&t=49s>

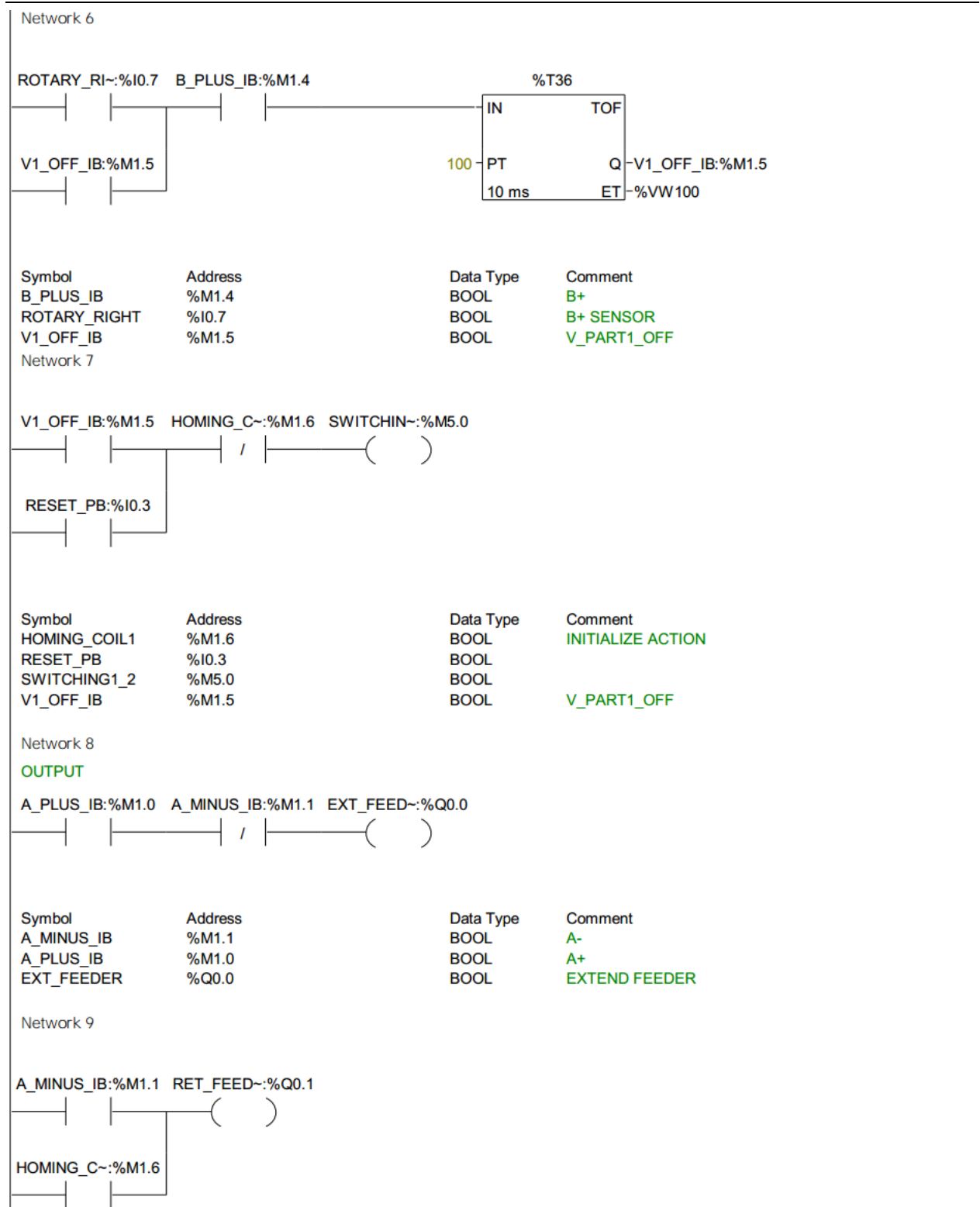
APPENDIX A – PLC CODES

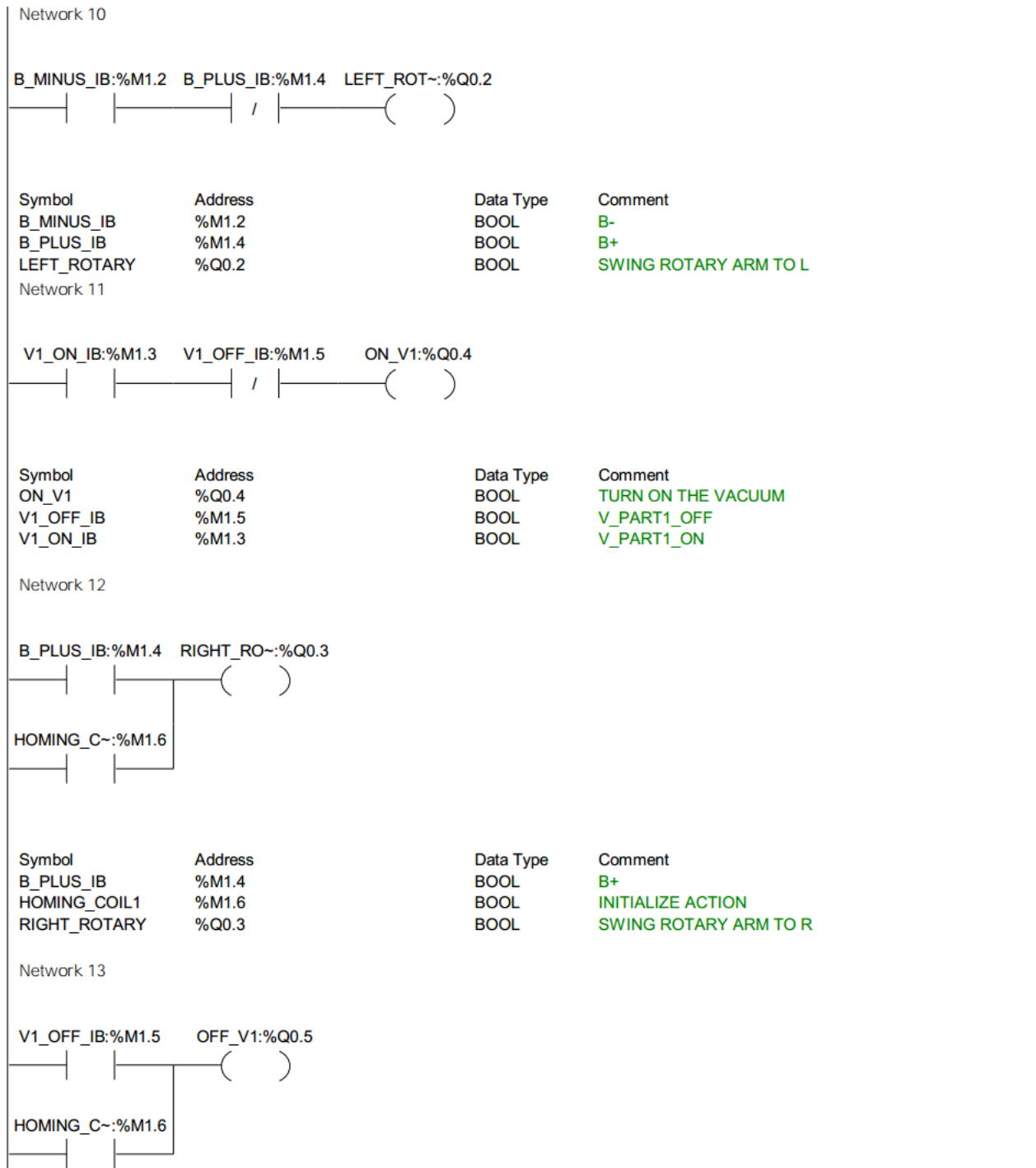
Appendix A contains the STEP 7- Micro/WIN program codes of two stations.

A.1 Program code of Station 1



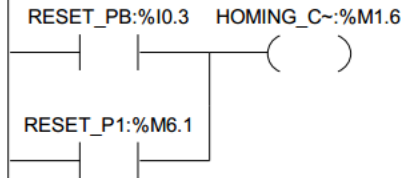






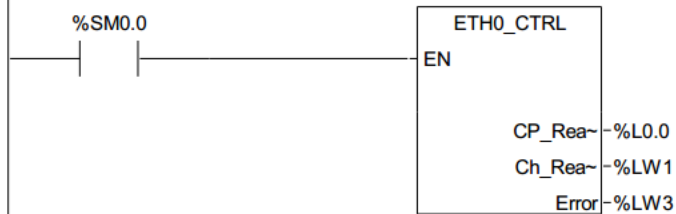
Symbol	Address	Data Type	Comment
HOMING_COIL1	%M1.6	BOOL	INITIALIZE ACTION
OFF_V1	%Q0.5	BOOL	TURN OFF THE VACUUM
V1_OFF_IB	%M1.5	BOOL	V_PART1_OFF

Network 14

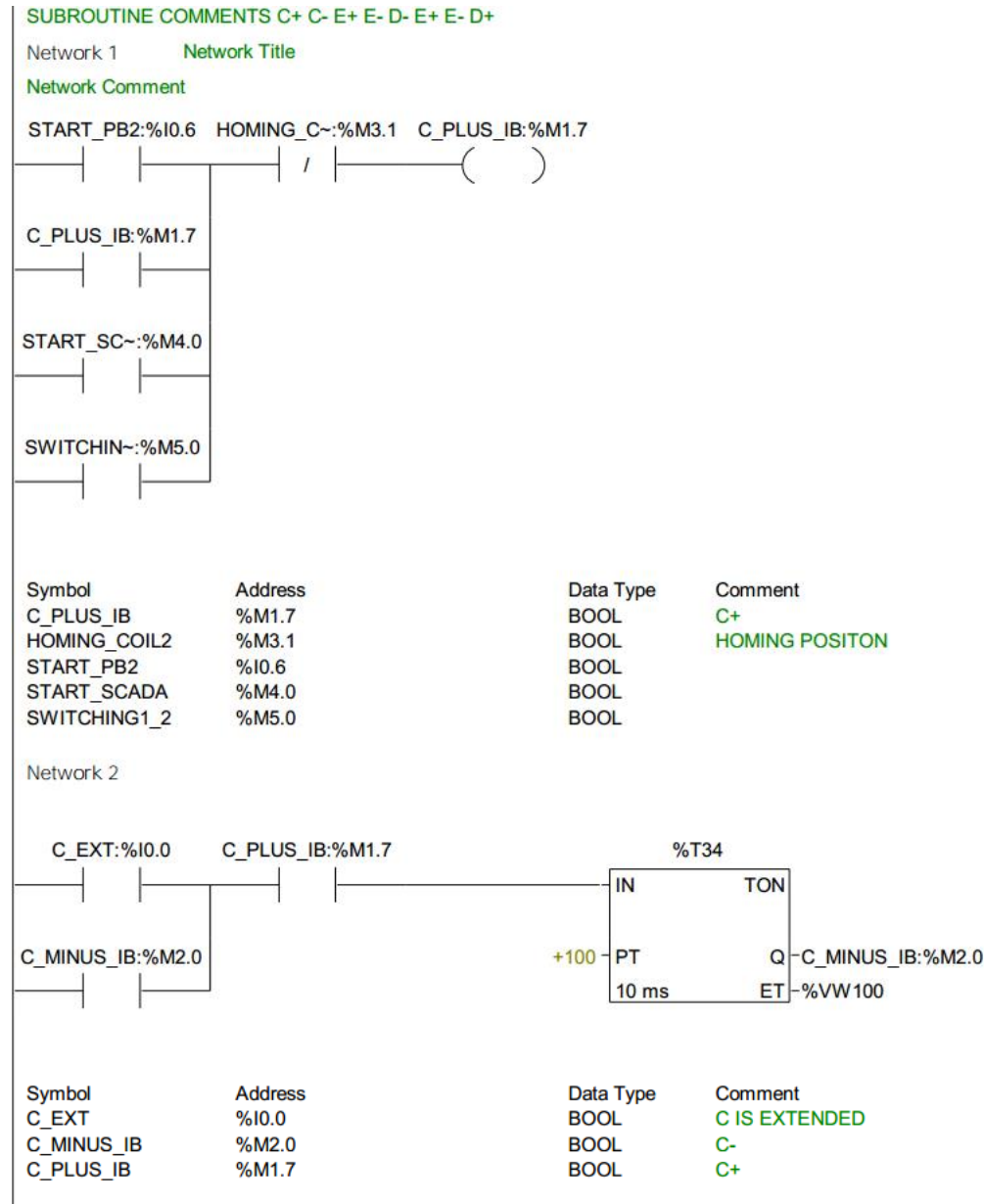


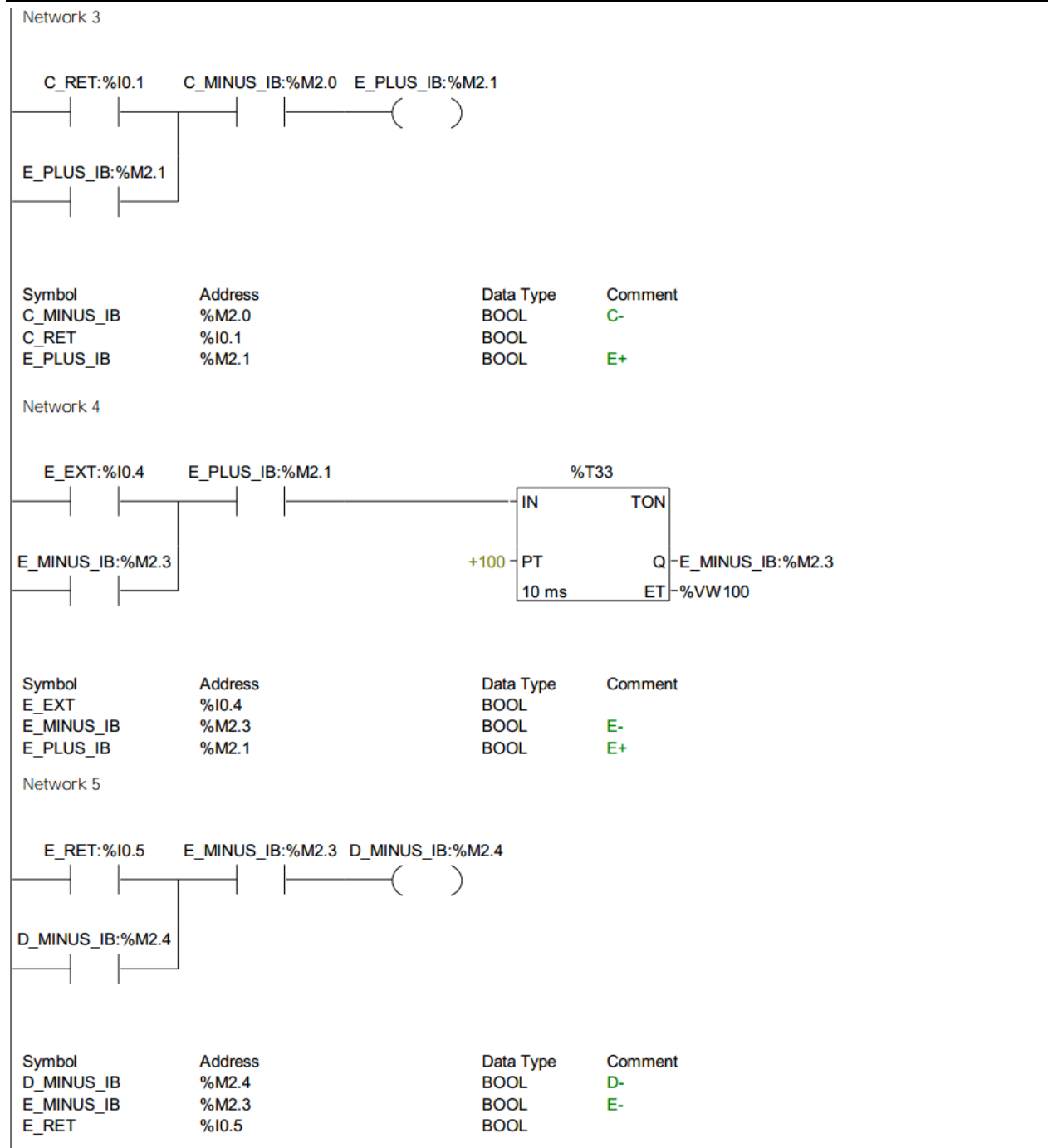
Symbol	Address	Data Type	Comment
HOMING_COIL1	%M1.6	BOOL	INITIALIZE ACTION
RESET_P1	%M6.1	BOOL	
RESET_PB	%I0.3	BOOL	

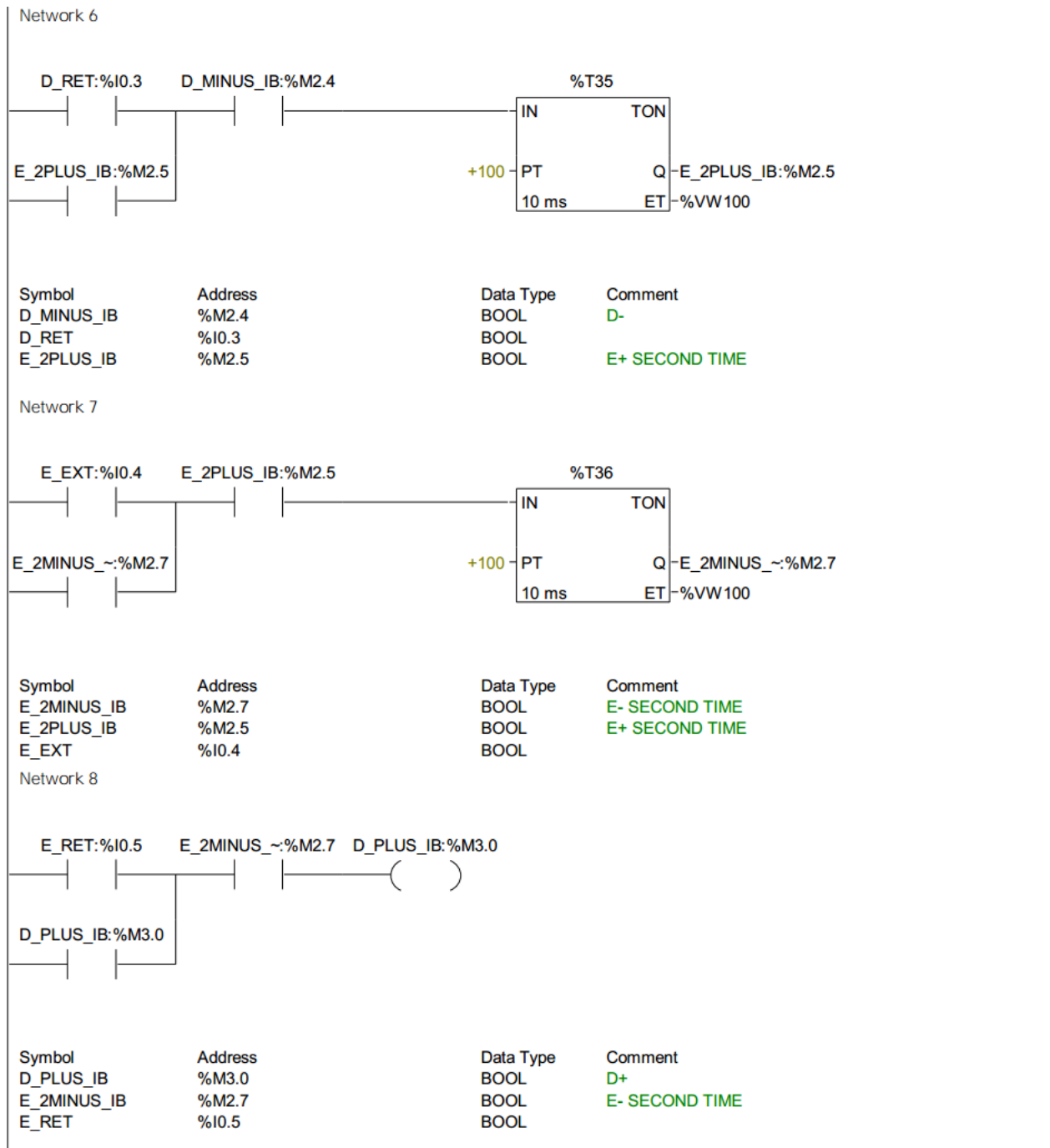
Network 15

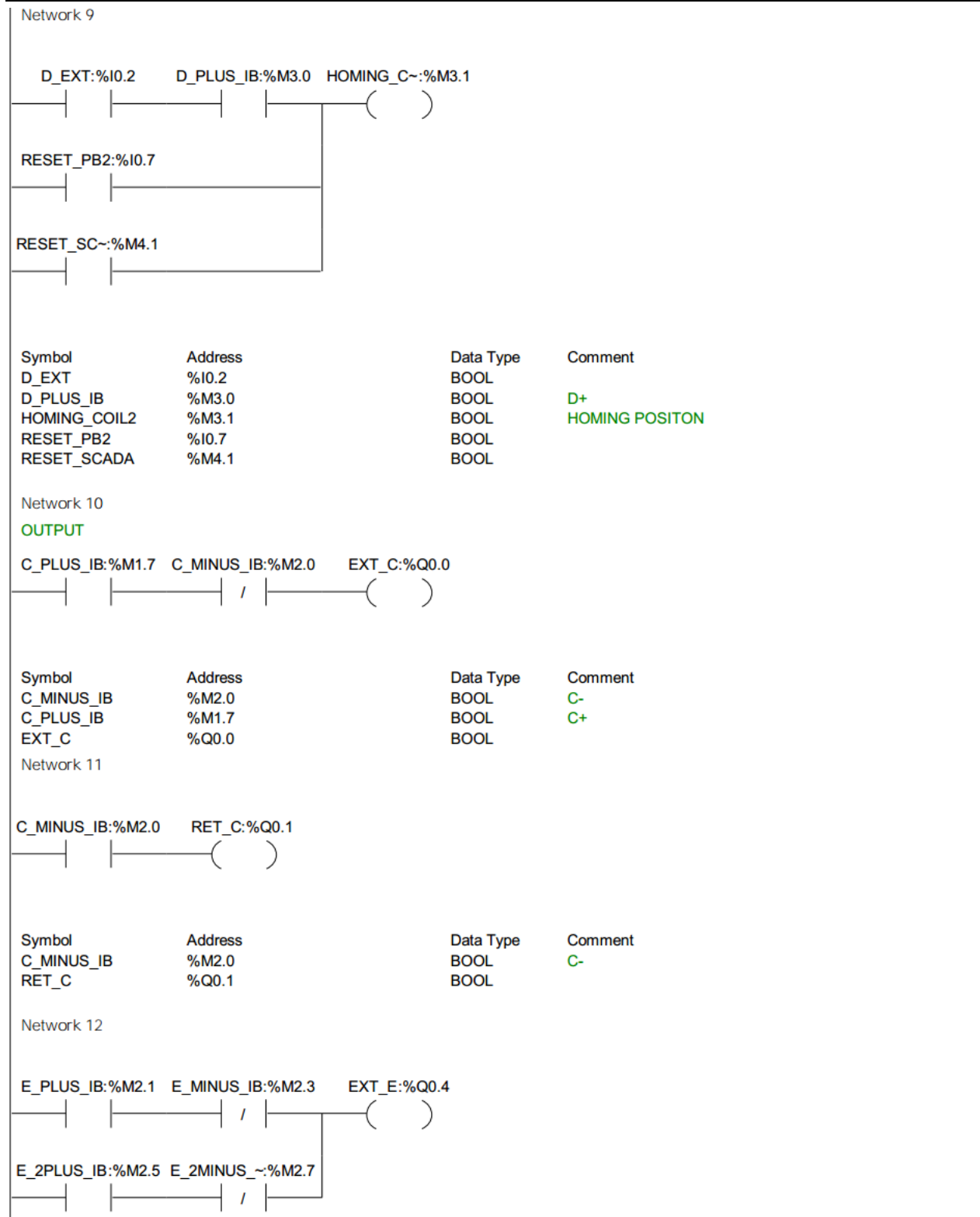


A.1 Program code of Station 2

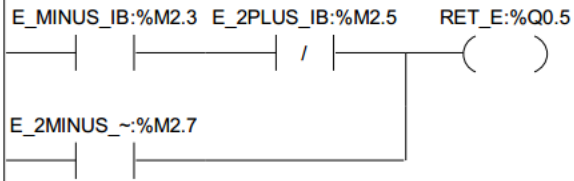






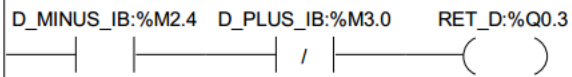


Network 13



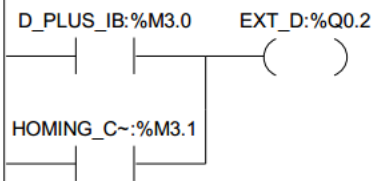
Symbol	Address	Data Type	Comment
E_2MINUS_IB	%M2.7	BOOL	E- SECOND TIME
E_2PLUS_IB	%M2.5	BOOL	E+ SECOND TIME
E_MINUS_IB	%M2.3	BOOL	E-
RET_E	%Q0.5	BOOL	

Network 14



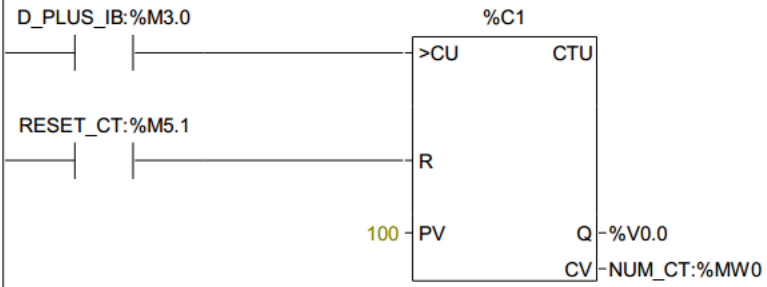
Symbol	Address	Data Type	Comment
D_MINUS_IB	%M2.4	BOOL	D-
D_PLUS_IB	%M3.0	BOOL	D+
RET_D	%Q0.3	BOOL	

Network 15



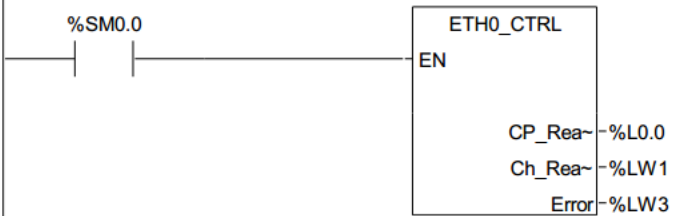
Symbol	Address	Data Type	Comment
D_PLUS_IB	%M3.0	BOOL	D+
EXT_D	%Q0.2	BOOL	
HOMING_COIL2	%M3.1	BOOL	HOMING POSITON

Network 16



Symbol	Address	Data Type	Comment
D_PLUS_IB	%M3.0	BOOL	D+
NUM_CT	%MW0	INT	COUNTER TIMES
RESET_CT	%M5.1	BOOL	RESET COUNTER

Network 17



APPENDIX B – RELIANCE 4 SCRIPT CODES

Appendix B contains codes of scripts in Reliance 4 for two visualization designs

B.1 Script code of exchanging data

```

rem *****
rem Reliance 4
rem Project: XXPART2
rem User: superstudent
rem Date: 21.5.2018
rem Time: 15:33:25
rem *****

DIM part1SWITCHING1_2, part2SWITCHING1_2

part1SWITCHING1_2 = RTag.GetTagValue("OPC1", "part1SWITCHING1_2")
part2SWITCHING1_2 = RTag.GetTagValue("OPC1", "part2SWITCHING1_2")
CUTOFFALL = RTag.GetTagValue("OPC1", "CUTOFFALL")
HOMING_COIL2 = RTag.GetTagValue("OPC1", "HOMING_COIL2")

IF part1SWITCHING1_2 = TRUE THEN
    part2SWITCHING1_2 = TRUE
END IF

IF part1SWITCHING1_2 = FALSE THEN
    part2SWITCHING1_2 = FALSE
END IF

IF HOMING_COIL = TRUE THEN
    CUTOFFALL = TRUE
END IF

IF HOMING_COIL = FALSE THEN
    CUTOFFALL = FALSE
END IF

RTag.SetTagValue "OPC1", "part1SWITCHING1_2", part1SWITCHING1_2
RTag.SetTagValue "OPC1", "part2SWITCHING1_2", part2SWITCHING1_2
RTag.SetTagValue "OPC1", "CUTOFFALL", CUTOFFALL
RTag.SetTagValue "OPC1", "HOMING_COIL2", HOMING_COIL2

```

B.2 Script code of virtual Station 1

```

rem *****
rem Reliance 4
rem Project: Thesis_visual
rem User: yutia
rem Date: 2018/3/25
rem Time: 19:41:00
rem *****

Option Explicit
Dim cylinder_forward, piston_X, rotary_drive, vacuum_signal1,
    vacuum_signal2
Dim bottom_visible, top_visible, piece_pushed, endposition_show
Dim vacuum1_visible, vacuum2_visible
Dim step_increment, step_num

rotary_drive = RTag.GetTagValue("Virtual1", "rotary_drive")
cylinder_forward = RTag.GetTagValue("Virtual1",
"cylinder_forward")
piston_X = RTag.GetTagValue("Virtual1", "piston_X")
piece_pushed = RTag.GetTagValue("Virtual1", "piece_pushed")
bottom_visible = RTag.GetTagValue("Virtual1", "bottom_visible")
top_visible = RTag.GetTagValue("Virtual1", "top_visible")
step_increment = RTag.GetTagValue("Virtual1", "step_increment")
step_num = RTag.GetTagValue("Virtual1", "step_num")
endposition_show = RTag.GetTagValue("Virtual1",
"endposition_show")
vacuum_signal1 = RTag.GetTagValue("Virtual1", "vacuum_signal1")
vacuum_signal2 = RTag.GetTagValue("Virtual1", "vacuum_signal2")
vacuum1_visible = RTag.GetTagValue("Virtual1", "vacuum1_visible")
vacuum2_visible = RTag.GetTagValue("Virtual1", "vacuum2_visible")

If step_num = 1 Then
    cylinder_forward = True
    piece_pushed = True
    bottom_visible = True
    vacuum1_visible = False
End if

If step_num = 2 Then
    top_visible = True
    bottom_visible = False
    cylinder_forward = False
    vacuum1_visible = False
End if

If step_num = 3 Then
    rotary_drive = True
    vacuum_signal2 = True

```

```
vacuum1_visible = True
vacuum2_visible = True
End if
```

```
If step_num = 4 Then
    rotary_drive = False
    endposition_show = True
```

```
    piece_pushed = False
    vacuum_signal1 = True
    vacuum2_visible = False
    vacuum1_visible = False
End if
```

```
RTag.SetTagValue "Virtual1", "cylinder_forward",
cylinder_forward
RTag.SetTagValue "Virtual1", "rotary_drive", rotary_drive
RTag.SetTagValue "Virtual1", "piece_pushed", piece_pushed
RTag.SetTagValue "Virtual1", "top_visible", top_visible
RTag.SetTagValue "Virtual1", "bottom_visible", bottom_visible
RTag.SetTagValue "Virtual1", "step_num", step_num
RTag.SetTagValue "Virtual1", "endposition_show",
endposition_show
RTag.SetTagValue "Virtual1", "vacuum_signal1", vacuum_signal1
RTag.SetTagValue "Virtual1", "vacuum_signal2", vacuum_signal2
RTag.SetTagValue "Virtual1", "vacuum1_visible", vacuum1_visible
RTag.SetTagValue "Virtual1", "vacuum2_visible", vacuum2_visible
```