

2221112

by William Utting

Submission date: 30-Jan-2023 11:09AM (UTC+0000)

Submission ID: 196412087

File name: 18983_William_Utting_2221112_180905_252019659.pdf (11.54M)

Word count: 4141

Character count: 20376

Course: HNC Electrical and Electronic Engineering
Module: SGEE4004 - Programmable Logic Controllers
Assignment Title: 1. Pick and Place Robot
Name: William Utting
University number: 2221112
Date Submitted: 30/01/2023
Word Count: 1917 words

Table of Contents

1. Work Plan for Lifelong Learning	3
2. Research: Schneider Electric	4
3. Hardware Investigation	6
Input and Output Addresses	9
4. Program Design and Explanation	10
5. Simulation and 6. Practical Testing	17
7. Discussion, Conclusions and Recommendations	18
8. Team Working	19
9. References	20
10. Appendices	
Appendix A: Tables	21
Appendix B: Supplementary Information	30
Appendix C: Figures	31
Appendix D: Final Code	35
Appendix E: Testing Evidence	36

1. Work Plan for Lifelong Learning

- Research into programmable logic controller (PLC) types and display the data gathered in a concise format.
- Investigate the hardware that is used in the robot and determine an input/output (I/O) list from the findings.
- Create a basic template for the human-machine interface (HMI) screens.
- Write pseudo code or a flow chart for the simulation, automatic and manual controls.
- Convert the pseudo code or flow chart into structured control language (SCL) code and implement it into the program, using comments to explain how the program functions as it is written.
- Test the program in the simulation mode first to ensure that the hardware is not damaged by any errors.
- Fix any errors and remove any code that does not have a function.
- Once it is operating successfully in simulation, test the program in a practical application.
- Assess the program's functionality and construct a conclusion.
- Assess the method of approaching the assignment.
- Include an appendix and a citation list for the research.

Sounds like a good plan

2. Research: Schneider Electric

Schneider produces both PLCs and PACs (see Appendix B, Item B1 for definitions); however, the program to be designed for the low-I/O system in this assignment would be more suited to a PLC, and therefore research must be conducted on the types of PLC they offer.

There are a few Schneider legacy systems, called Quantum, Momentum and Premium. (Nagda 2022) Quantum PLCs are rack mounted and large, making them too large for this application. The Premium PLC has the same drawbacks as the Quantum. Moreover, both Quantum and Premium are discontinued; therefore, acquiring the parts needed may be costly and difficult. The Momentum PLCs are designed for large, industrial-scale operations where distributed I/O systems are most effective (Process Industry Forum 2022). Since the robot we are choosing the PLC for is standalone and not integrated into a large system, it is unnecessarily powerful, overly complex and too costly to warrant Momentum's use.

Schneider also offers PLCs under the Modicon name. The most relevant to the application is the standard Modicon range. Out of that range, three likely candidates have been selected: Zelio, Modicon M221 and the Modicon M241. This is due to the low number of I/Os needed, the small-scale application and the lack of expandability needed. (Schneider Electric 2022)
this citation should contain a comma

Using data gathered from Schneider's website, it can be determined which PLC is the best option (see Appendix A, Table A1). The Zelio SR3B101B + SR3XT141B I/O extension combination is the first to be eliminated. While operation of the robot arm is likely possible using this device, it is inconvenient to program using only ladder and function blocks. An I/O extender is also needed, which increases the overall size, and as it is being used on such a small scale it makes it the worst option for a compact solution. The TM221C24T would work, but is not the best option out of the three. It is probably the best value for money, the price is close to the Zelio combination and it offers a lot of the same or similar functionality as the TM241CEC24T. The TM241CEC24T is the most desirable option, mainly because it allows programming in structured text (ST) languages. This allows for simple control and complex operations to be conducted.

Overall, the final decision would be based mainly on the price of the PLCs. If there is a small budget, the best option would be the TM221C24T. It can make the robot functional; however, the code behind it would probably have to be written in an instruction list. Since the PLC cannot be programmed in an ST, the college would have to find a different method to teach

students the ST. If the budget is large, the TM241CEC24T would be the best option: it allows code in a variety of languages, so students can learn more by using it. The EcoStruxure Machine Expert software used by the PLC is used for most of Schneider's PLCs, unlike the Zelio software which is used less.

Therefore, the recommendation is to use TM241CEC24T if the correct funds are available or TM221C24T if the budget is restricted and the Zelio SR3B101B for simple applications.

Good conclusions

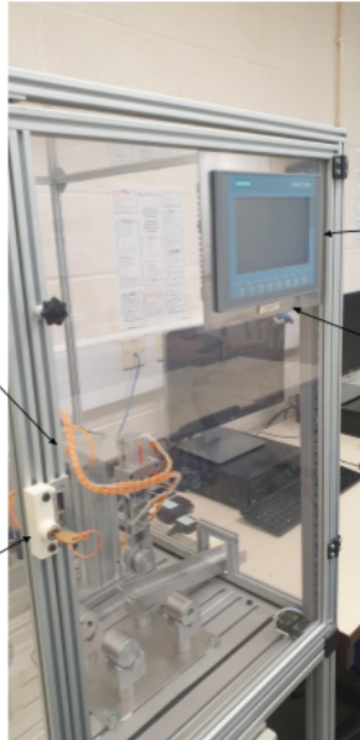
You have presented a reasoned argument

blank page ?

3. Hardware Investigation

The solenoids are supplied with an electrical current through the white wire. The air used by the solenoids is transported by the orange pneumatic tubes. These tubes seem to be connected by quick connect fittings. The components dealing with the high pressure are substantial enough to handle the pressures. Wires and tubes are managed well and kept out of the way of the robot's range of movement.

The system is fully enclosed with a locked door for safety.



This system uses a Siemens Simatic HMI as a user interface for operators and engineers. It can be used to display data and control the system. Wires for the HMI are kept out of the way of the robot and securely attached to the frame.

There is a label beneath the HMI with the IP of 10.13.90.125.

There are four stations that the robot can be navigated to; they are designed to hold cylindrical objects that are entered into the system through the in chute into the first input station. There are two storage stations. After the storage stations there is one output station/chute. Both chutes are angled so the objects cannot roll back into, or out of, the stations.

There are three red push buttons labelled X10 to X12. These could be used as physical start switches.

The PLC's IP is displayed on a label: 10.13.90.124



There is a locked box that the PLC of the robot is stored in. This decreases the risk of the PLC being tampered with.

There are two switches with the labels X13 and X14 above them.

There is a switch that turns the air compressor on and off. It does not seem to be connected to the PLC, due to the lack of labelling.

There is an emergency stop button that is wired to cut the power to the system. This is useful due to there being no reliance on the PLC to shut the system down in an emergency.

The PLC passed a safety test in November of 2022. At the date of writing, the test was conducted very recently. This indicates that the system is likely in a good operating condition.

The wires of the system are all routed to the same place before being routed to the components. Each wire is clearly labelled, which will make fault finding and repairs easy.

The compressed air's pressure can be measured through a physical dial. The location of the dial is not ideal; it could be made more accessible and visible for the engineers and operators. It appears that it can be moved easily due to the aluminium extrusion being used as a base.



It seems like the air from the compressor is split through this block into different tubes for each solenoid. The red tube leads to the air reservoir.

The robot arm is mounted on a steel bar and bearings. This gives a smooth motion and minimal resistance. However, the bearings may wear out over time.

The structural components of the robot are made out of aluminium, which is lighter than some alternatives while retaining strength. The fixings holding it together are made of steel; this reduces the chance of materials cold welding together, making it much easier to disassemble for repairs and upgrades.



The robot is connected into the network using ethernet cables.

The cage surrounding the robot is constructed out of aluminium extrusion and some form of clear plastic. The aluminium extrusion allows for new components to be added and existing components to be moved easily. Aluminium extrusion is much more modular than other options. The clear plastic keeps the operator or engineer safe from the robot while being able to clearly see the robot. This is much better than a fence which would partially obstruct your vision.

3.Input and Output Addresses

The addresses for the inputs of the system are as follows:

Sensors

- %I0.0 - up sensor
- %I0.1 - down sensor
- %I0.2 - open gripper sensor
- %I0.3 - closed gripper sensor
- %I0.4 - in sensor
- %I0.5 - out sensor
- %I0.6 - right sensor
- %I0.7 - left sensor
- %I1.5 - object sensor

Buttons and Switches

- %I1.0 - button X10
- %I1.1 - button X11
- %I1.2 - button X12
- %I1.3 - switch X13
- %I1.4 - switch X14

The addresses for the outputs are:

Solenoids

- %Q0.0 - down solenoid
- %Q0.1 - up solenoid
- %Q0.2 - close gripper solenoid
- %Q0.3 - open gripper solenoid
- %Q0.4 - out solenoid
- %Q0.5 - in solenoid
- %Q0.6 - left solenoid
- %Q0.7 - right solenoid

4. Program Design and Explanation

Good explanation given

Safety Interlock Design

Safety interlocks reduce the risk of the machine damaging itself or injuring the operator. In the case of this robot, we want the arm to move left, right, in or out only when the arm is fully up and the grippers are closed. The code for this is simple, so it can be written in ladder in the MAIN function of the PLC so that it is always active.

The UP solenoid will activate IF ((Auto Toggle = off AND Up HMI request = on) OR (Auto Toggle = on AND Up Auto = on) OR (LEFT and RIGHT sensors = off) OR (IN and OUT sensors = off)) AND UP sensor = off (see Appendix C, Figure 1). This allows manual control or auto control and stops the arm dropping while in motion.

The system shown in Appendix C, Figure C2, allows the DOWN solenoid to activate in manual or automatic modes but only when it is above a station.

For every other movement except open and close, the UP sensor must be on for the arm to move (see Appendix C, Figure C3).

Graphical User Interface (GUI)

A user interface (UI) must be simple to navigate and discern data from. Any controls must be clearly labelled and large enough to accurately use from an HMI. To achieve the objectives of the assignment, the following features are needed:

Operator screen

- On/Off switch
- Manual/Auto switch
- Visualisation of objects in the system and where the arm is
- A button to go to the engineer's screen
- What step number the arm is on

Engineer's screen

- On/Off switch
- Manual/Auto switch
- Manual controls
- Step number

- Object sensor light
- Object counter
- Object sensor simulated input
- Object counter simulator
- Operator screen button
- System visualisation

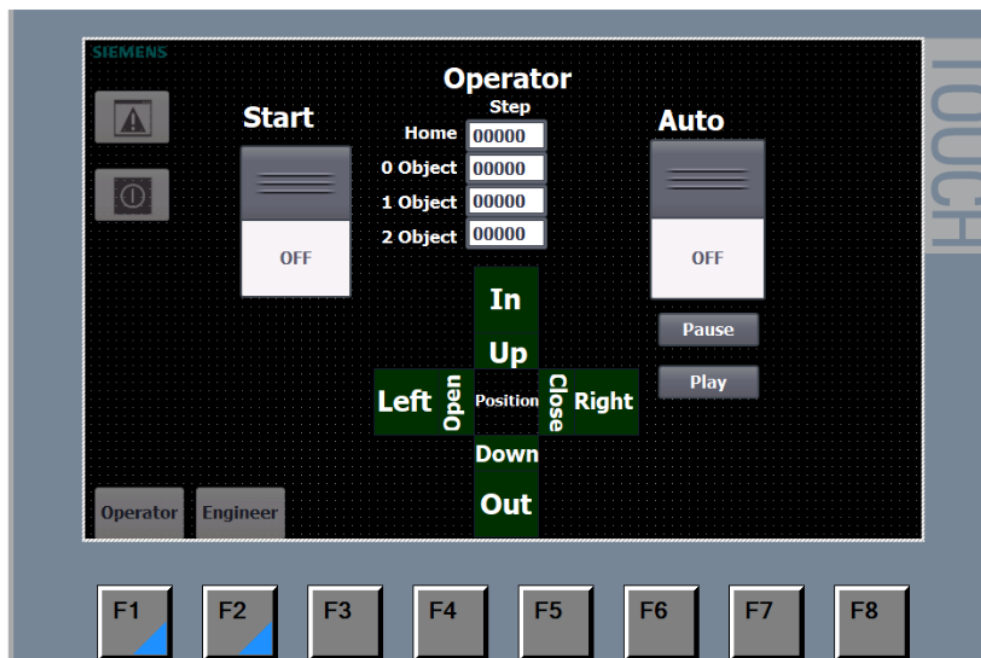
To help with the final GUI design, the screens can be designed roughly first (see Appendix C, Figures C4 & C5). The simplistic design of the screens should help the user to learn how to use the robot and quickly discern data. The large buttons make interactions simple.

Good explanation given

Final HMI Design

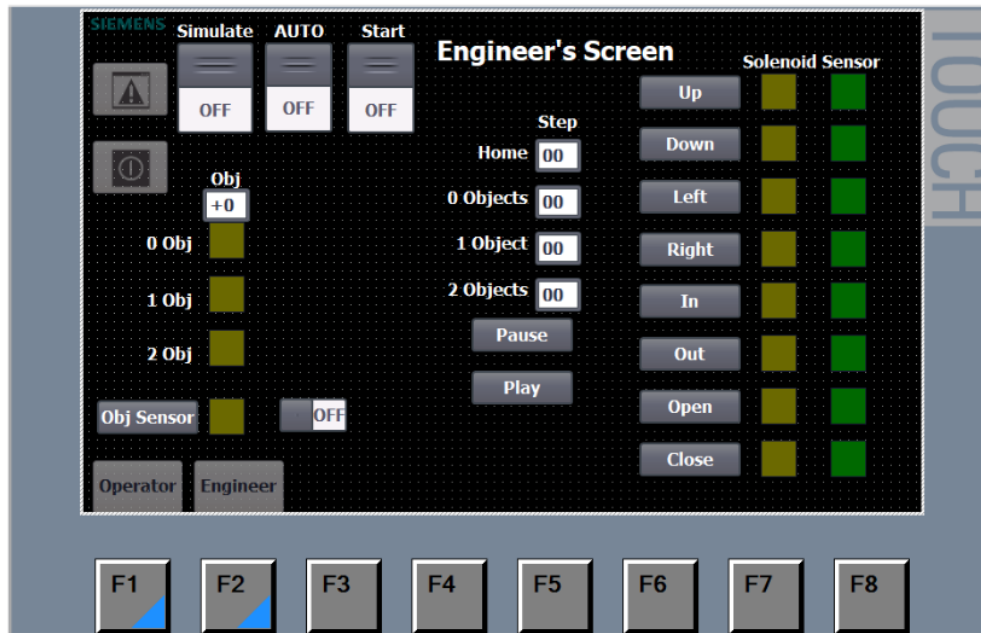
The final UI is closely based on the rough design. The addition of the Pause and Play buttons allows the operator to pause the automatic program and resume it from the same point. The auto switch sets every step back to 0 to prevent failure if moved in manual mode then switched back to auto.

Good explanation given



The engineer's screen is also the same as the rough design but also with added Pause and Play buttons. The resetting functionality of the auto switch is present on this screen as well.

start should be a push button not a switch



There are controls on this screen which have not been asked for, you must learn to stick to the design specification

Automatic Mode Design

To help with writing a functional program, a flow chart with the main steps of the program can be created. The original flow chart is shown in Appendix C, Figure C6.

The principle behind this design is to split the code into four main sections: the home, 0 object, 1 object and 2 objects sections. For example, if the machine is off and there is currently one object in the system, the order of sections executed would be: Power on -> run home code -> wait for object sensor -> move object in system to station 3 -> move new object to station 2 -> return to home. To achieve this, CASE OF loops can be nested within each other. The nonlinear design allows the program to specifically run certain sections depending on variables (see Appendix B, Item B2 for pseudocode).

Ladder programming in MAIN

Appendix C, Figure C7 shows how the object counter is set up and how the object sensor's output is made suitable for the program.

Final Automatic Program

See Appendix D for the final program.

5.Simulation and 6.Practical Testing

Simulation Testing

A structured testing table is used to fully test the program (see Appendix A, Table A2). All tests were eventually passed, even if a fix was required to have a feature fully operating. With all of the simulation tests concluded and passed, practical testing could now be conducted.

Practical Testing

The tests were similar to simulation testing, with a few added (see Appendix A, Table A3). From testing the program in a practical scenario any bugs will be shown. All issues will be noted in the testing table and further discussed in the conclusion. It was decided by the group that the program developed in this paper would be put forward to the practical testing.

7. Discussion, Conclusions and Recommendations

The aim of this assignment was to produce a program to control a pick and place robot. It had to include safety interlocks, simulation, manual controls, automatic mode, an engineer's screen and an operator's screen. Reading the brief after finishing the assignment has shown that the majority of the objectives have been completed and that the program developed covered all of the criteria. The development of the program was documented and the thought process behind the design explained. The UI design featured everything necessary in a tidy and user-friendly way. It displays all of the data possible and has buttons to simulate or change certain values. The object simulation feature had to be removed as it was causing too many issues. The suggestion of using the TM241CEC24T PLC from Schneider Electric in place of the current Siemens PLC would have worked. It had the correct amount of I/O and the programming capability to conduct the same operations.

The program shown in this document could have been optimised more. There are a few pieces of code that could be removed, but this would also cause a few issues that would need to be fixed to make the program functional again. The biggest optimisation would be an entire rework of the structure of the program. Instead of having four different, separate sections of code to handle different scenarios, the code could have been compiled into a linear instruction set. This would use less CASEs and decrease the overall line count and complexity. Sensors could also be installed at the stations to determine whether an object is there. This would eliminate the possibility of an operator manually placing an object in the system and the system not recognising the object is there. Another recommendation would be to install a turnstile at the input chute to avoid multiple objects entering station 1 at the same time. In addition, to help with fixing or expanding the code, the variables could be renamed to be more logical and relevant. Moreover, a password could be set on the engineer's screen to ensure only engineers have access to it. From the practical testing it was clear that the program needed more work to function safely. Operators could change the value of the steps through the interface which made the robot execute the wrong steps at the wrong times. This resulted in a couple of major errors: the arm trying to pick up an object from the output chute and the arm moving an object from station 2 to 3 before clearing station 3. If the text boxes displaying the values were locked these errors could not occur. Another issue that arose was the failure of the simulation mode, it did not work. On top of that the state of the switches were not the same between screens. Overall, the program would need more work before being used on a machine on a permanent basis.

8. Team Working

At the beginning, the class as a whole did the hardware investigation and identified I/O. This did provide teamwork and discussion. While the class members were programming, sometimes a few people would have similar problems and discuss them to find a solution, which also provided some form of teamwork. However, the majority of the program development was individual. This was good as it gave a range of different ideas to choose from for a final program. While testing the final program any issues were discussed, this provided ideas into how they would be fixed.

9. References

A reasonable range of sources listed correctly

- Nagda, V. (2022). *Different Types of Schneider PLC*. [online] Inst Tools. Available at: <https://instrumentationtools.com/types-of-schneider-plc/> [Accessed 28 Dec. 2022].
- Payne, J. (2013). *PLC vs. PAC*. [online] Available at: https://www.idc-online.com/technical_references/pdfs/electronic_engineering/PLC_vs_pac.pdf [Accessed 18 Dec. 2022].
- Process Industry Forum. (2022). *What is distributed I/O and what are the benefits?* [online] Available at: <https://www.processindustryforum.com/article/distributed-io-benefits> [Accessed 3 Jan. 2023].
- Schneider Electric. (2022). [online] Available at: <https://innovatronica.com.ec/wp-content/uploads/2020/04/PLC-Industria-Schneider.pdf> [Accessed 28 Dec. 2022].
- Schneider Electric. (2022). *Product Comparator | Schneider Electric*. [online] Available at: <https://www.se.com/uk/en/product-comparator/2y5km/TM241CEC24T%7CSR3B101B%7CTM221C24T> [Accessed 3 Jan. 2023].
- Schneider Electric. (2022). *PLC (Programmable Logic Controllers), PAC and Dedicated Controllers | Schneider Electric UK*. [online] Available at: <https://www.se.com/uk/en/product-category/3900-plc-pac-and-dedicated-controllers/?filter=business-1-industrial-automation-and-control> [Accessed 28 Dec. 2022]

10.Appendix A: Tables

Table A1

	Zelio (SR3B101B + SR3XT141B I/O extension)	Modicon M221 (TM221C24T)	Modicon M241 (TM241CEC24T)
Price	£171.02 + £94.94 (SR3XT141B I/O extension)	£284.75	£475.67
I/O ports	10 + 14	24	24
Languages	LDR, FBD	LDR, GRAFCET, SFC, List, IL	LDR, FBD, SFC, IL, ST
Execution time	90ms cycle time 50ms response LDR 50–255ms response FBD	0.3ms for event & periodic task	0.3ms for event & periodic task
Memory		256kB, 256kB for internal variables	8MB system, 64MB system
Redundancy	Needs multiple processors	Needs multiple processors	Needs multiple processors
Scalability	I/O expansion	I/O expansion	I/O expansion
Compatibility	None for software	None for software	None for software
Connection	COM port Bluetooth USB Modem	USB 2.0 RS232/RS485	USB 2.0 RS232/RS485 RJ45 J1939
Ethernet	No	No	Yes
Software	Zelio Soft 2	EcoStruxure Machine Expert	EcoStruxure Machine Expert

Size	90mm × 71.2mm × 59.5mm + 90mm × 72mm × 59.5mm	90mm × 70mm × 110mm	90mm × 95mm × 150mm
-------------	---	---------------------	---------------------

(Automation IT 2023; Schneider Electric 2020)

Table A2

Test	Expected Result	Result	Notes/Fix	Pass/Fail	Evidence
Simulation switch	Turns on/off simulation mode	Switches simulation on/off			Appendix E: Figure E1
Auto switch	Turns on/off auto mode	Switches in and out of auto mode			Appendix E: Figure E2
Start switch	Starts/stops auto program	Allows the program to run			Appendix E: Figure E3
Manual controls	Using the controls should move the arm but not surpass safety interlocks	Manual controls work, arm cannot move across the horizontal plane unless the arm is in the up position			Appendix E: Figure E4
Safety interlocks	Works in manual mode and automatic mode	Works in manual Works in			Appendix E: Figure E5

		auto			
Object sensor button	Simulates an object entering the system	Simulates an object entering the system	Sensor turns off halfway through the first step of 1obj and 2obj. Fix:  An extra timer in the sensor circuit to delay the reset of the sensor by 4s		Appendix E: Figure E6
Run with 0 objects	Moves object to station 2	Functions as expected			Appendix E: Figure E7
Run with 1 object	Moves 1 object to station 3 and 1 to station 2	Functions as expected			Appendix E: Figure E8
Run with 2 objects	Shuffles the objects through the system and output 1 object	Functions as expected			Appendix E: Figure E9
Turn off simulation halfway through running	Stops auto program and resumes when put back into simulation	Pauses and resumes the program running			Appendix E: Figure E10

Pause/Play buttons	Pauses and resumes the program	Pauses and resumes			Appendix E: Figure E11 & E12
Pause program, move arm in manual controls, turn auto back on, play	Wipes the program's step and starts with the send home steps	Wipes the program's step when auto is turned off, runs the home code when turned back on			Appendix E: Figure E13
Check that object count lights work	Respective object lights turn on when the object is in the system	Lights reflect the object count in the system correctly			Appendix E: Figure E14
Set arm to down, closed position and start auto	Program returns home	Returns the arm home			Appendix E: Figure E15
Try to use manual controls in auto mode	No effect on auto mode	Manual controls do not work in auto			Appendix E: Figure E16
Press the object sensor button throughout	No effect on operations	Does not affect the program			

program running					
Switch between screens while auto is running	Program continues to run	As expected			Appendix E: Figure E17
Start auto from operator screen	Program starts	Program starts			Appendix E: Figure E18
Start auto from engineer's screen, stop from operator's screen, vice versa	Program starts and stops	As expected			Appendix E: Figure E19
Check the position of arm is correct on operator screen	Active sensors match between screens	Position is correct			Appendix E: Figure E20
Check current step is correct on operator screen	Step numbers match between screens	Matches between screens			Appendix E: Figure E20

Table A3

Test	Expected Result	Result	Notes/Fix	Pass/Fail	Evidence
Simulation switch	Turns on/off simulation mode	Does not work	Slows down the operating speed of the arm but does not simulate it.		Appendix E: Figure E35
Auto switch	Turns on/off auto mode	Turns auto on or off			Appendix E: Figure E24
Start switch	Starts/stops auto program	Starts/stops program	Needs to be in an OR state with the auto switch instead of AND		Appendix E: Figure E24
Manual controls	Using the controls should move the arm but not surpass safety interlocks	Works correctly			Appendix E: Figure E30
Safety interlocks	Works in manual mode and automatic mode	Correctly stops the arm moving when in the down position			Appendix E: Figure E23
Object sensor	Simulates an object entering	Works correctly	Object count in the		Appendix E: Figure E25

button	the system		text box can be altered which messes up the current step		& E28
Run with 0 objects	Moves object to station 2	Runs correctly when not tampered with			Appendix E: Figure E26
Run with 1 object	Moves 1 object to station 3 and 1 to station 2	Runs correctly			Appendix E: Figure E28 & E29
Run with 2 objects	Shuffles the objects through the system and output 1 object	Runs correctly			Appendix E: Figure E36
Turn on simulation halfway through running	Stops auto program and resumes when simulation is turned off	Simulation does not work			Appendix E: Figure E35
Pause/Play buttons	Pauses and resumes the program	Works correctly			Appendix E: Figure E26
Pause program, move arm in manual controls, turn auto	Wipes the program's step and starts with the send home steps	Wipes the step count but it still causes issues			Appendix E: Figure E30

back on, play					
Check that object count lights work	Respective object lights turn on when the object is in the system	They work correctly when the object count is not tampered with			Appendix E: Figure E34
Try to use manual controls in auto mode	No effect on auto mode	No effect			
Press the object sensor button throughout program running	No effect on operations	No effect			
Switch between screens while auto is running	Program continues to run	Switches do not display the correct state			Appendix E: Figure E33
Start auto from operator screen	Program starts	Program starts			
Start auto from engineer's	Program starts and stops	Program does start and stop,			

screen, stop from operator's screen, vice versa		however, the switches do not indicate the correct state			
Check the position of arm is correct on operator screen	Active sensors match between screens	Displays the correct position			Appendix E: Figure E33
Check current step is correct on operator screen	Step numbers match between screens	Step numbers match between screens			Appendix E: Figure E33
Load multiple objects into the input area	Arm cannot pick up an object from the input station	Misaligns the objects in the chute	Turnstile at the beginning to allow 1 object through at a time		Appendix E: Figure E26
Put an object in a station manually	The program runs as if the object is not there	Arm attempts to put an object in the occupied station	Fixed with sensors on each station to detect objects		
Put an object in the unloading	Object rolls out of system	Object rolls out of the system			

station					
Insert an irregular shaped object	Gripper cannot grip object	Object cannot roll into system	An unlikely scenario as objects must roll into the system		
Open cage while robot is running	Robot is shut off	Robot shuts off, it may also cause errors in the program			

Appendix B: Supplementary Information

Item B1: Definition of a PLC and PAC

There is little difference between PLCs and programmable automation controllers (PACs); with the advancements of technology, the difference between the two has decreased. PLCs are much simpler than PACs: the ladder programming language used by PLCs ensures that electrical engineers can easily create simple logical functions. PACs use an open architecture, which requires operations to be coded in C, BASIC and SCL languages. This increases the skill required to use a PAC but also allows a wide range of operations to be conducted. PACs can also handle a lot more data than PLCs can. PLCs' simplicity has the advantage of decreasing the processing time, which increases the overall speed. PACs are used in an environment where complex automation and large amounts of I/O are present. This means they will likely be present in automated assembly lines, for example in a car manufacturing plant. PLCs are used for smaller automation or logical operations. They are more likely to be used to control one machine. (Payne 2013)

Item B2

Decide between sending the arm home or moving objects around the stations

1: Send the arm home

Steps for sending the arm home

2: Move objects

Decide which operation to run

1: Move object from station 1 to 2

Steps

2: Move object from station 2 to 3

Steps

3: Move object from station 3 to 4

Steps

Appendix C: Figures

your ladder code lacks network titles and comments

Figure C1

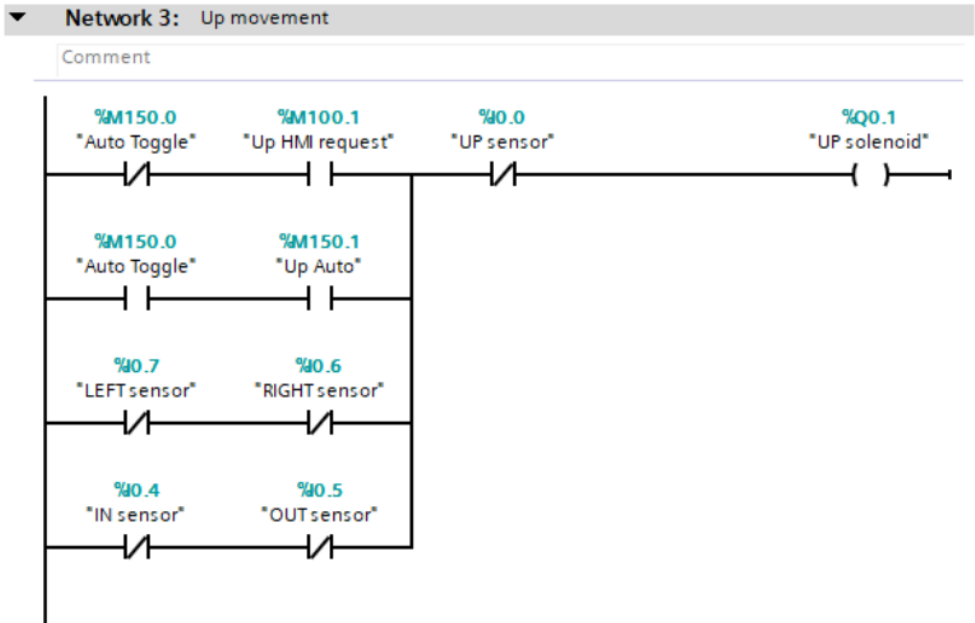


Figure C2

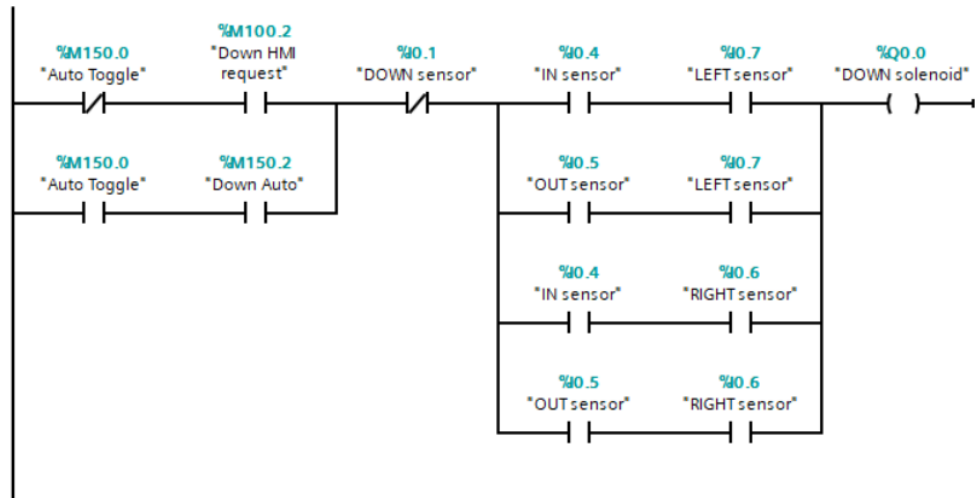


Figure C3

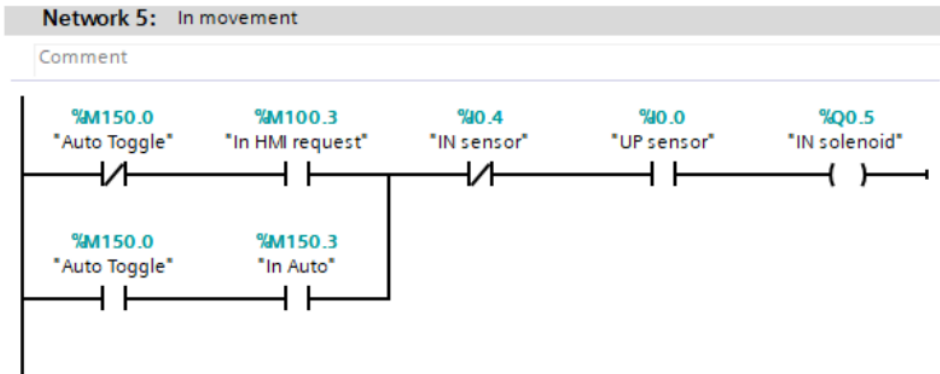


Figure C4

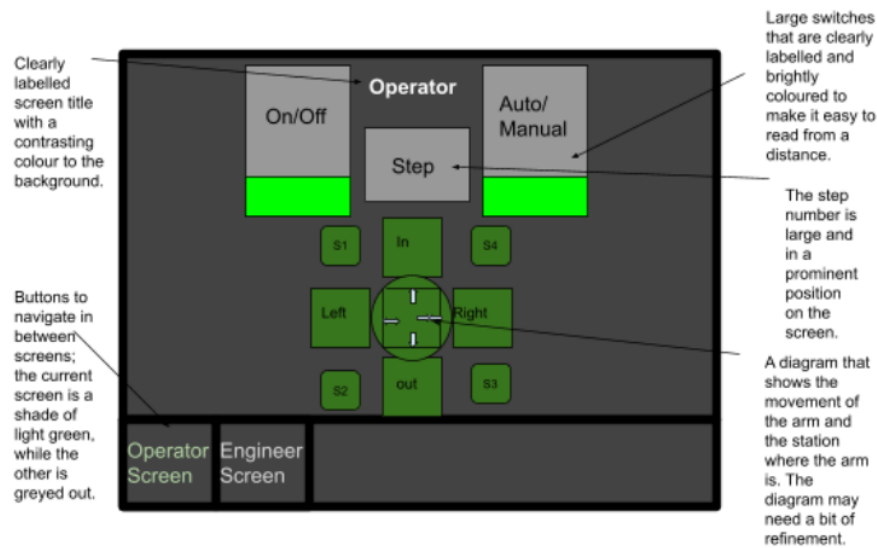


Figure C5

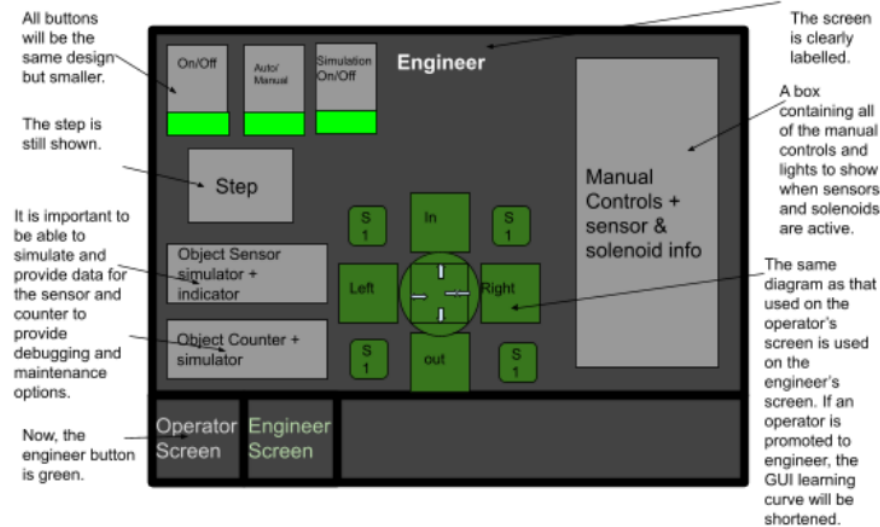


Figure C6

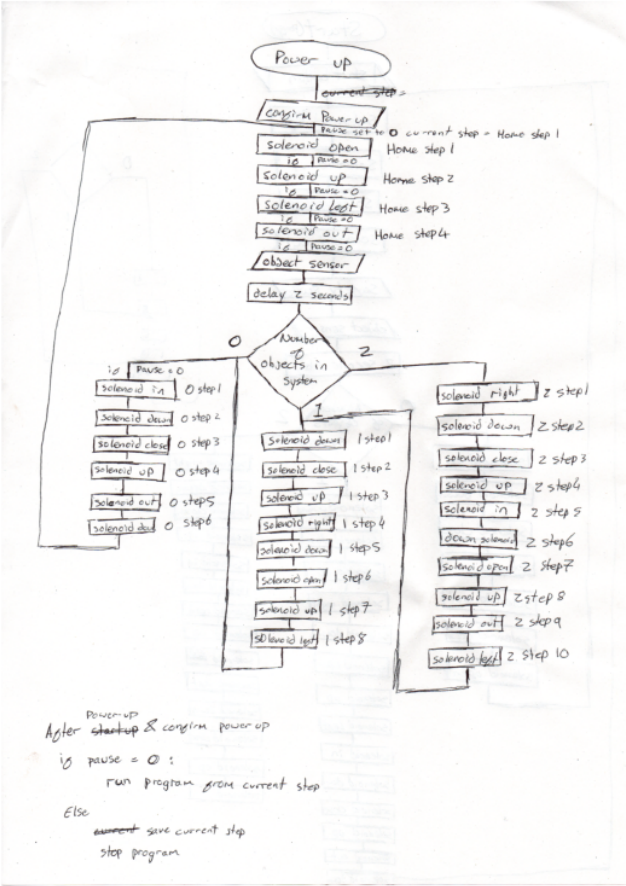
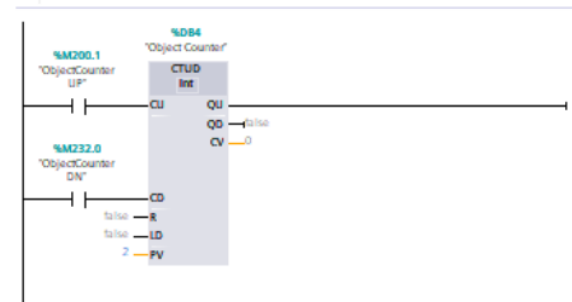


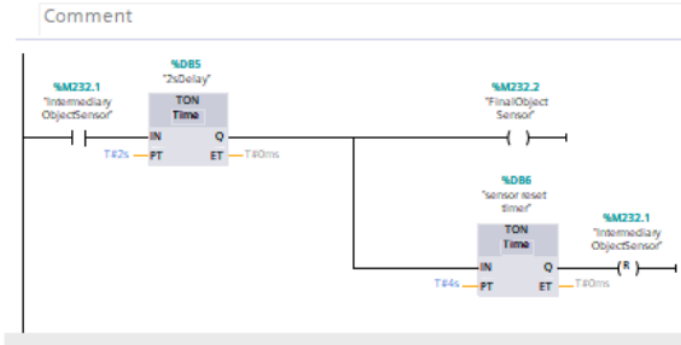
Figure C7



Network 13:



Network 14:



Appendix D: Final Code

```

1
2 IF "Powerup" THEN //On the powerup pulse all of the step and next step variables are set to 0
3     "SendHomeStep" := 0;
4     "NextSendHomeStep" := 0;
5     "HomeStep" := 0;
6     "NextHomeStep" := 0;
7     "NumObjectStep" := 0;
8     "NextNumObjectStep" := 0;
9     "0ObjectStep" := 0;
10    "Next0ObjectStep" := 0;
11    "1ObjectStep" := 0;
12    "Next1ObjectStep" := 0;
13    "2ObjectStep" := 0;
14    "Next2ObjectStep" := 0;
15 END_IF;
16
19 IF "Auto Toggle" = 1 AND "StartSwitch" = 1 THEN //Auto program will only run when the toggle is on and the start switch is on
20
21
22 CASE "SendHomeStep" OF //Decides If the Arm is sent home
23     0: //Arm is sent Home
24     CASE "HomeStep" OF ... END_CASE;
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58

```

```

65         1: //Arm is not sent home (only used after arm is sent home)
66         CASE "NumObjectStep" OF //Decides which code to run depending on the number of objects in the system
67             0: //0 objects
68                 CASE "0ObjectStep" OF ... END_CASE;
111
112             1: //1 object
113                 CASE "1ObjectStep" OF ... END_CASE;
168
169             2: //2 objects
170                 CASE "2ObjectStep" OF ... END_CASE;
253
254
255         END_CASE;
256
257
258
259     END_CASE;
260
261 END_IF;

07
08     U: //U objects
09     CASE "0ObjectStep" OF //Steps for 0 objects in system
10         0: //moves arm in
11             IF "FinalObjectSensor" = 1 OR "1ObjectStep" = 7 THEN //only runs when the object sensor is on or the steps for
12                 "In Auto" := 1; //moving an object from station 2 to 3 is complete
13             IF "IN sensor" = 1 THEN
14                 "In Auto" := 0;
15                 "Next0ObjectStep" := 1;
16             END_IF;
17         END_IF;
18
19         1:
20             "Down Auto" := 1; //moves arm down
21             IF "DOWN sensor" = 1 THEN
22                 "Down Auto" := 0;
23                 "Next0ObjectStep" := 2;
24             END_IF;
25
26         2:
27             "Close Auto" := 1; //closes grippers
28             IF "CLOSED sensor" = 1 THEN
29                 "Close Auto" := 0;
30                 "Next0ObjectStep" := 3;
31             END_IF;
32
33         3:
34             "Up Auto" := 1; //moves arm up
35             IF "UP sensor" = 1 THEN
36                 "Up Auto" := 0;
37                 "Next0ObjectStep" := 4;
38             END_IF;
39
40         4:
41             "Out Auto" := 1; //moves arm out
42             IF "OUT sensor" = 1 THEN
43                 "Out Auto" := 0;
44                 "Next0ObjectStep" := 5;
45             END_IF;
46
47         5:
48             "Down Auto" := 1; //moves arm down, increases object count, continues to the home steps
49             IF "DOWN sensor" = 1 THEN
50                 "Down Auto" := 0;
51                 "ObjectCounterUP" := 1;
52                 "0ObjectStep" := 0;
53                 "NextHomeStep" := 0;
54                 "NextSendHomeStep" := 0;
55             END_IF;
56     END_CASE;
57 END_IF;

```

```

112 1: //1 object
113 CASE "1ObjectStep" OF //Steps for 1 object in system
114 0:
115 IF "FinalObjectSensor" = 1 OR "2ObjectStep" = 11 THEN //runs if the object sensor is on or
116 "Down Auto" := 1; //the steps to move an object from station 3 to 4 is complete
117 IF "DOWN sensor" = 1 THEN //moves arm down
118 "Down Auto" := 0;
119 "Next1ObjectStep" := 1;
120
121 END_IF;
122
123 1:
124 "Close Auto" := 1; //closes grippers
125 IF "CLOSED sensor" = 1 THEN
126 "Close Auto" := 0;
127 "Next1ObjectStep" := 2;
128 END_IF;
129
130 2:
131 "Up Auto" := 1; //moves arm up
132 IF "UP sensor" = 1 THEN
133 "Up Auto" := 0;
134 "Next1ObjectStep" := 3;
135 END_IF;
136
137 3:
138 "Right Auto" := 1; //moves arm right
139 IF "RIGHT sensor" = 1 THEN
140 "Right Auto" := 0;
141 "Next1ObjectStep" := 4;
142 END_IF;
143
144 4:
145 "Down Auto" := 1; //moves arm down
146 IF "DOWN sensor" = 1 THEN
147 "Down Auto" := 0;
148 "Next1ObjectStep" := 5;
149 END_IF;
150
151 5:
152 "Open Auto" := 1; //open grippers
153 IF "OPEN sensor" = 1 THEN
154 "Open Auto" := 0;
155 "Next1ObjectStep" := 6;
156 END_IF;
157
158 6:
159 "Up Auto" := 1; //moves arm up
160 IF "UP sensor" = 1 THEN
161 "Up Auto" := 0;
162 "Next1ObjectStep" := 7;
163 "0ObjectStep" := 5;
164 END_IF;
165
166 7:
167 "Left Auto" := 1; // moves arm left and continues to steps for moving an object from station 1 to 2
168 IF "LEFT sensor" = 1 THEN
169 "Left Auto" := 0;
170 "0ObjectStep" := 0;
171 "NextNumObjectStep" := 0;
172 END_IF;
173 END_CASE;
174
175 2: //2 objects
176 CASE "2ObjectStep" OF ... END_CASE;
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256

```

```

170 CASE "2ObjectStep" OF //Steps for 2 objects in system
171 0:
172     IF "FinalObjectSensor" = 1 THEN//runs when the object sensor is active
173         "Right Auto" := 1;//moves arm right
174     IF "RIGHT sensor" = 1 THEN
175         "Right Auto" := 0;
176         "Next2ObjectStep" := 1;
177     END_IF;
178 END_IF;
179
180 1:
181     "Down Auto" := 1;//moves arm down
182     IF "DOWN sensor" = 1 THEN
183         "Down Auto" := 0;
184         "Next2ObjectStep" := 2;
185     END_IF;
186
187 2:
188     "Close Auto" := 1;//closes grippers
189     IF "CLOSED sensor" = 1 THEN
190         "Close Auto" := 0;
191         "Next2ObjectStep" := 3;
192     END_IF;
193
194 3:
195     "Up Auto" := 1;//moves arm up
196     IF "UP sensor" = 1 THEN
197         "Up Auto" := 0;
198         "Next2ObjectStep" := 4;
199     END_IF;
200
201 4:
202     "In Auto" := 1;//moves arm in
203     IF "IN sensor" = 1 THEN
204         "In Auto" := 0;
205         "Next2ObjectStep" := 5;
206     END_IF;
207
208 5:
209     "Down Auto" := 1;//moves arm down
210     IF "DOWN sensor" = 1 THEN
211         "Down Auto" := 0;
212         "Next2ObjectStep" := 6;
213     END_IF;

```

```

209         6:
210         "Open Auto" := 1;//open grippers
211         IF "OPEN sensor" = 1 THEN
212             "Open Auto" := 0;
213             "Next2ObjectStep" := 7;
214         END_IF;
215
216         7:
217         "ObjectCounterDN" := 0;//decreases object count as an object has been put in the exit chute
218         "Up Auto" := 1;//moves arm up
219         IF "UP sensor" = 1 THEN
220             "Up Auto" := 0;
221             "Next2ObjectStep" := 8;
222         END_IF;
223
224         8:
225         "Out Auto" := 1;//moves arm out
226         IF "OUT sensor" = 1 THEN
227             "Out Auto" := 0;
228             "Next2ObjectStep" := 9;
229         END_IF;
230
231         9:
232         "Left Auto" := 1;//moves arm left
233         IF "LEFT sensor" = 1 THEN
234             "Left Auto" := 0;
235             "Next2ObjectStep" := 10;
236         END_IF;
237
238         10:
239         "Down Auto" := 1;//the next two steps are unnecessary, however, if i remove them step 3 of moving an
240         IF "DOWN sensor" = 1 THEN//object from station 2 to 3 doesn't work so I'm keeping these in
241             "Down Auto" := 0;
242             "Next2ObjectStep" := 11;
243         END_IF;
244
245         11:
246         "Open Auto" := 1;
247         IF "OPEN sensor" = 1 THEN
248             "Open Auto" := 0;
249             "1ObjectStep" := 0;
250             "ObjectCounterDN" := 1;
251             "NextNumObjectStep" := 1;//allows the code to move an object from station 2 to 3 to run
252         END_IF;
253     END_CASE;
254
255
256     "SendHomeStep" := "NextSendHomeStep"; //sets the current step to the next step
257     "HomeStep" := "NextHomeStep"; //sets the current step to the next step
258     IF "Object Counter".CV = 2 THEN//when there are 2 objects in the system it switches to the code to move an object
259         "NumObjectStep" := 2; //from station 3 to 4
260     END_IF;
261
262     IF "Object Counter".CV = 1 OR "2ObjectStep" = 11 THEN//when there is one object in the system or the code for moving an object
263         "NumObjectStep" := 1;//from station 3 to 4 has run it runs the code to move an object from station 2 to 3
264     END_IF;
265
266     IF "Object Counter".CV = 0 OR "1ObjectStep" = 7 THEN//if there are no objects in the system or the code for moving an object
267         "NumObjectStep" := 0;//from station 2 to 3 has been run it runs the code to move an object from station 1 to 2
268     END_IF;
269
270
271
272
273
274
275
276
277
278
279
280
281
282     "0ObjectStep" := "Next0ObjectStep";//sets current steps to next steps
283     "1ObjectStep" := "Next1ObjectStep";
284     "2ObjectStep" := "Next2ObjectStep";
285

```

Appendix E: Testing Evidence

Figure E1

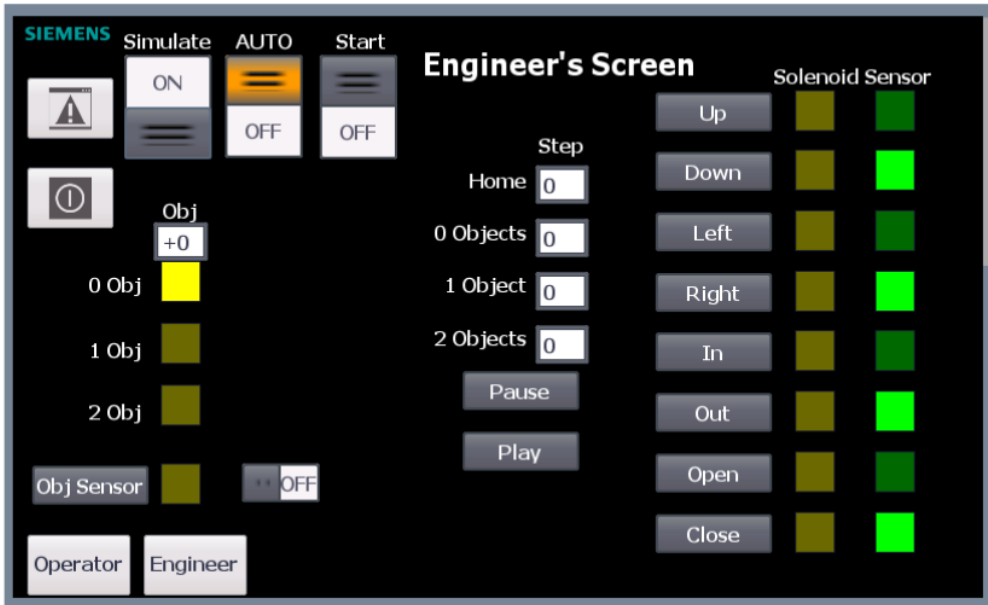


Figure E2

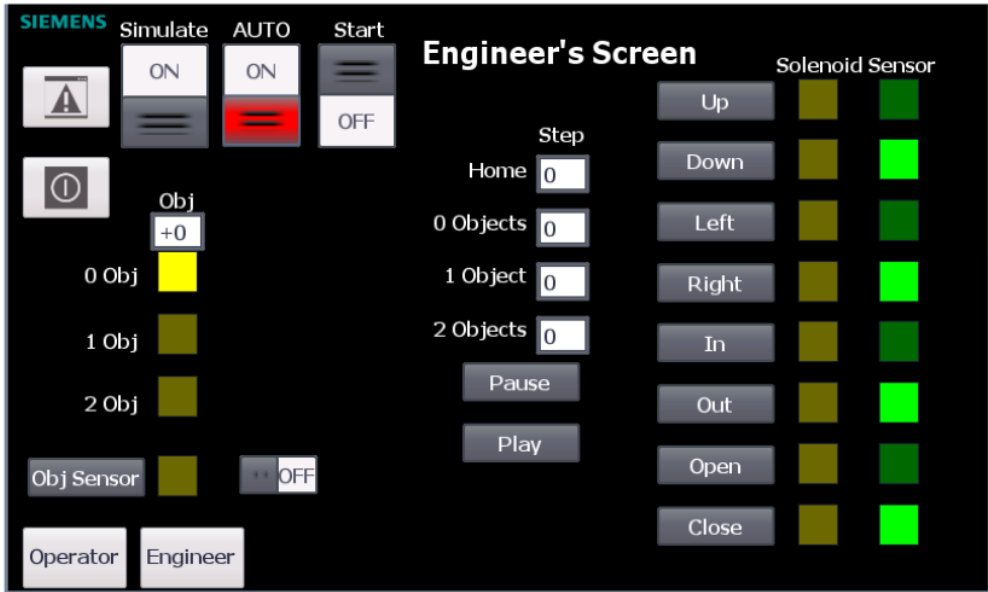


Figure E3

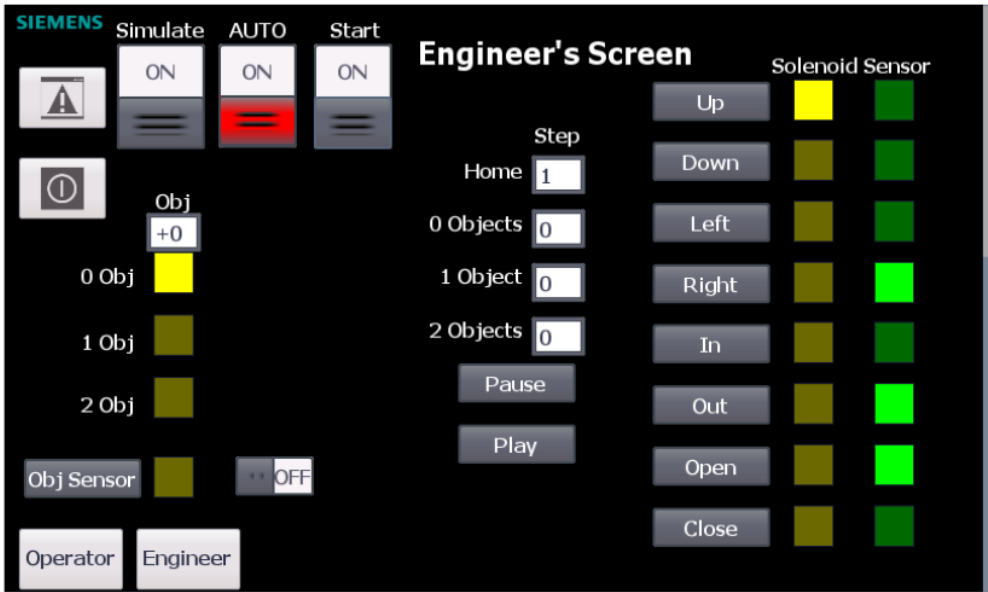


Figure E4

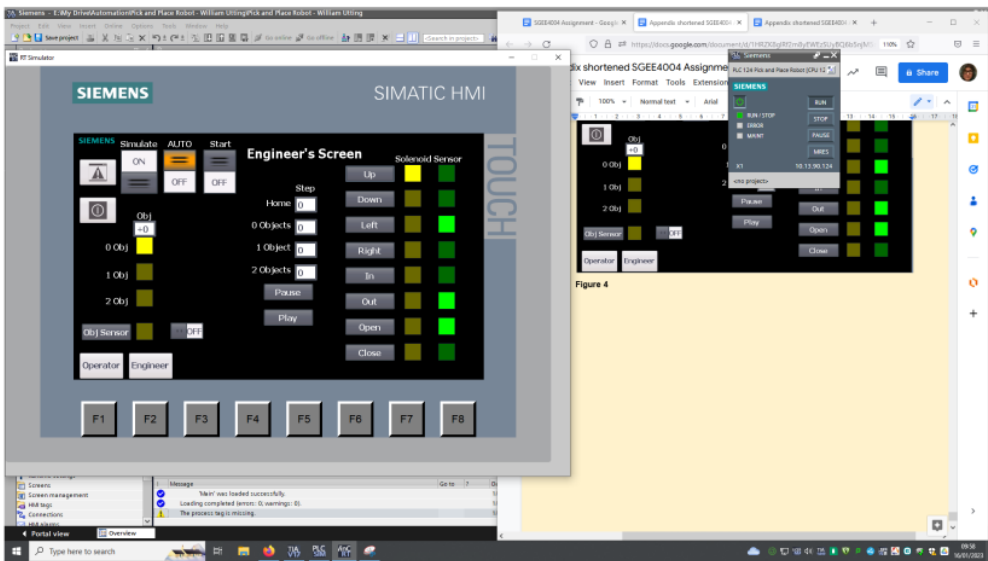


Figure E7

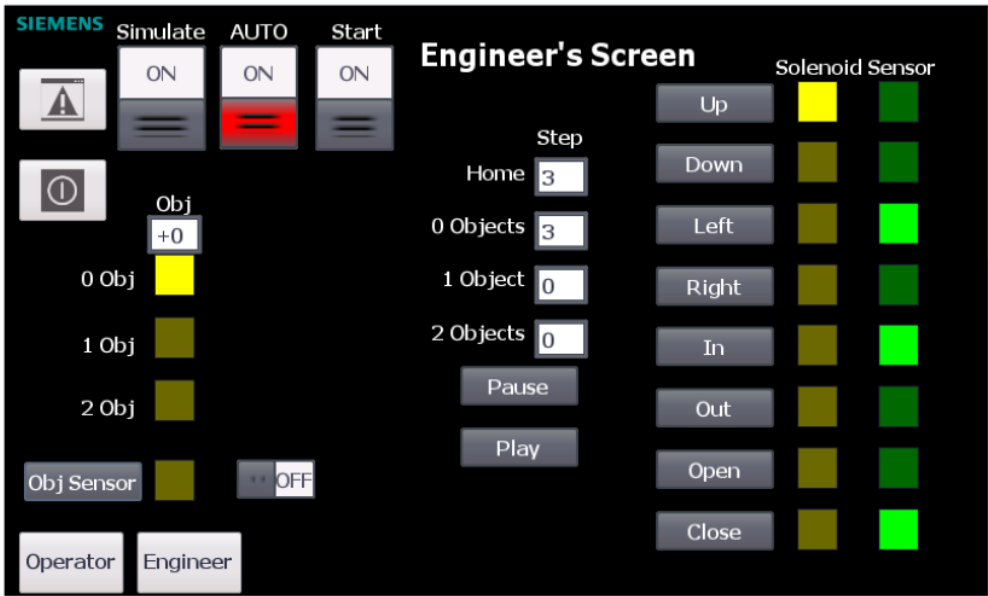


Figure E8

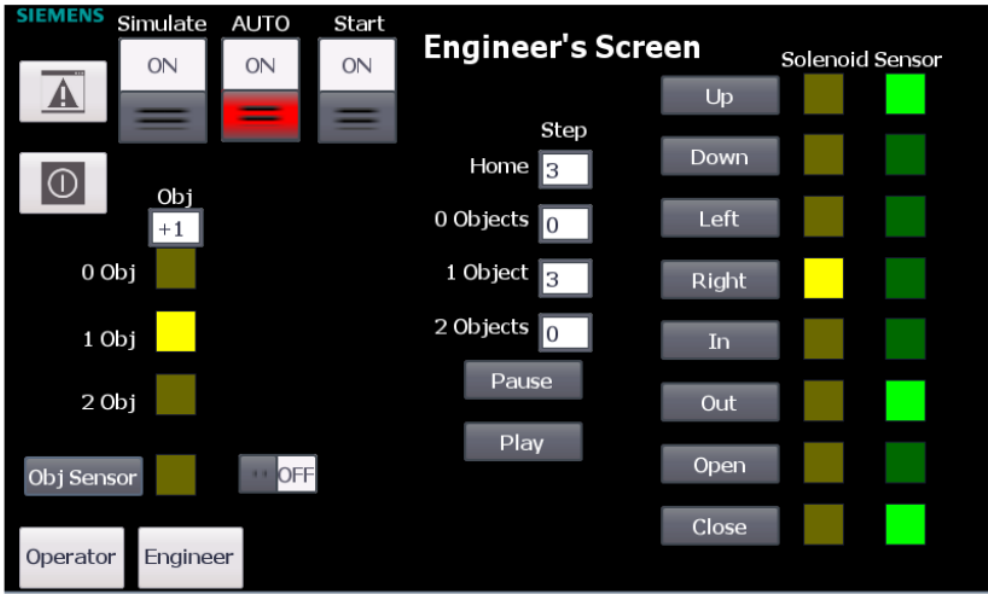


Figure E9

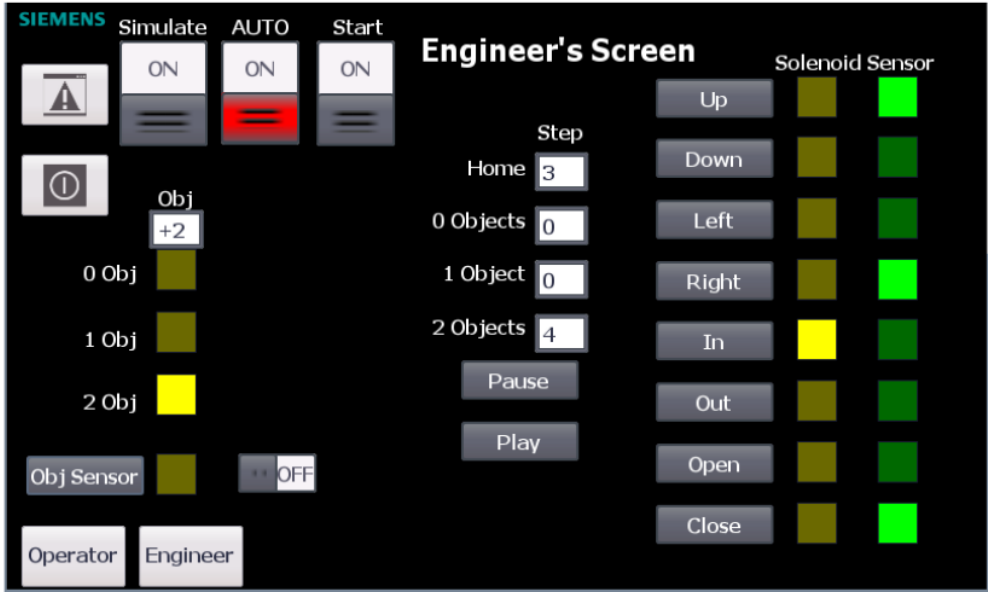


Figure E10

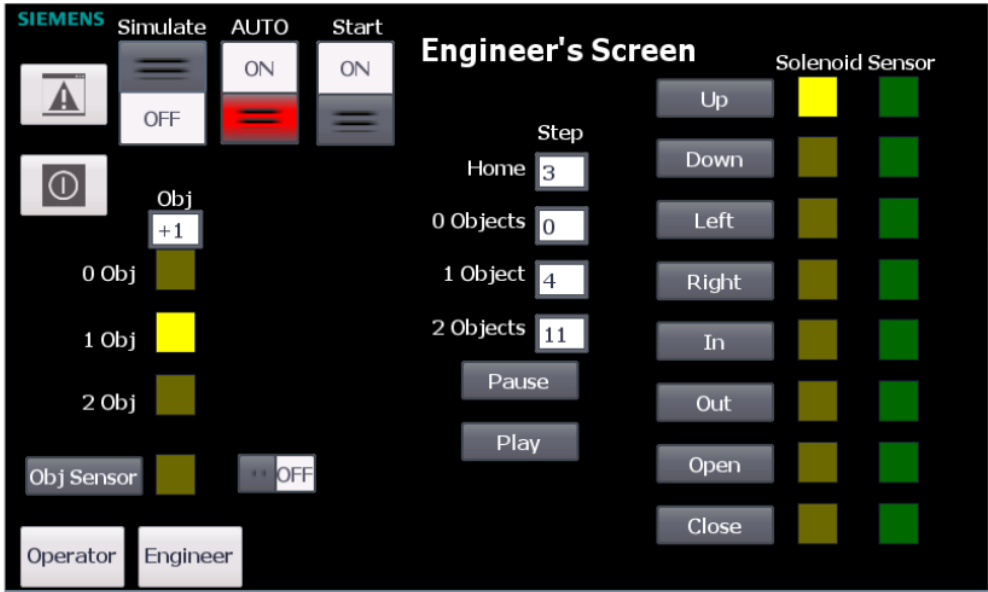


Figure E11

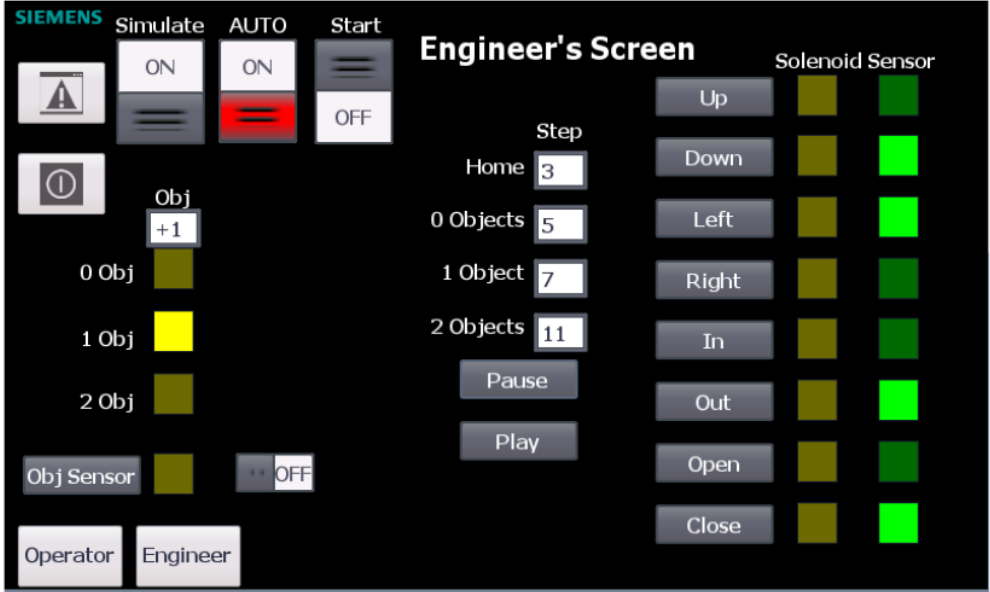


Figure E12

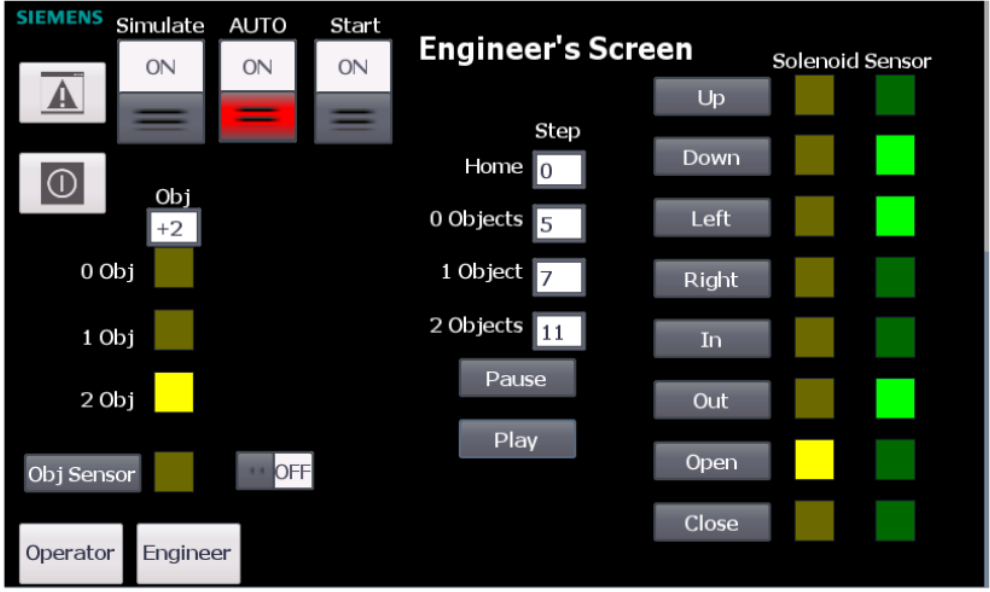


Figure E13

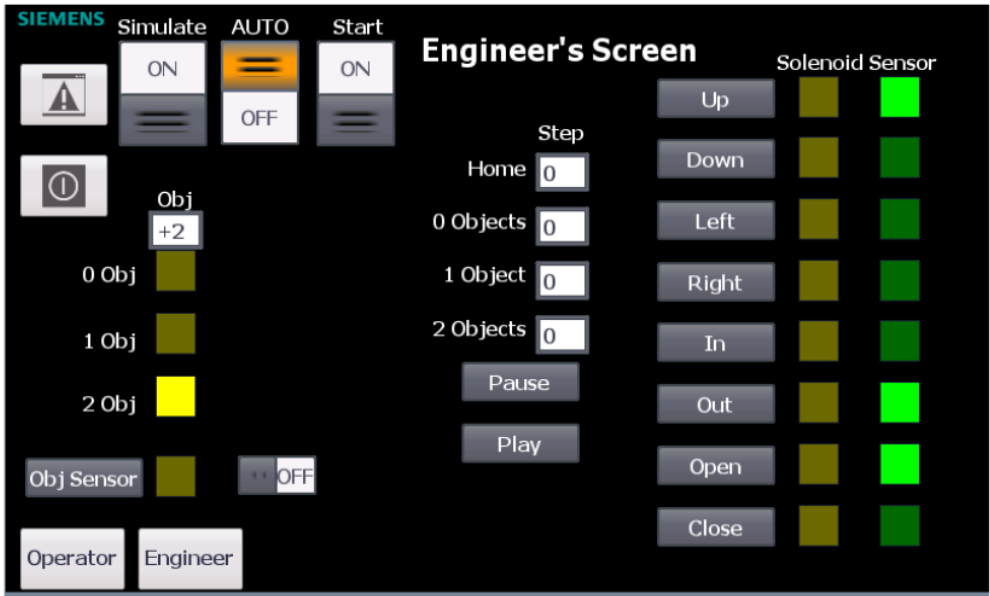


Figure E14

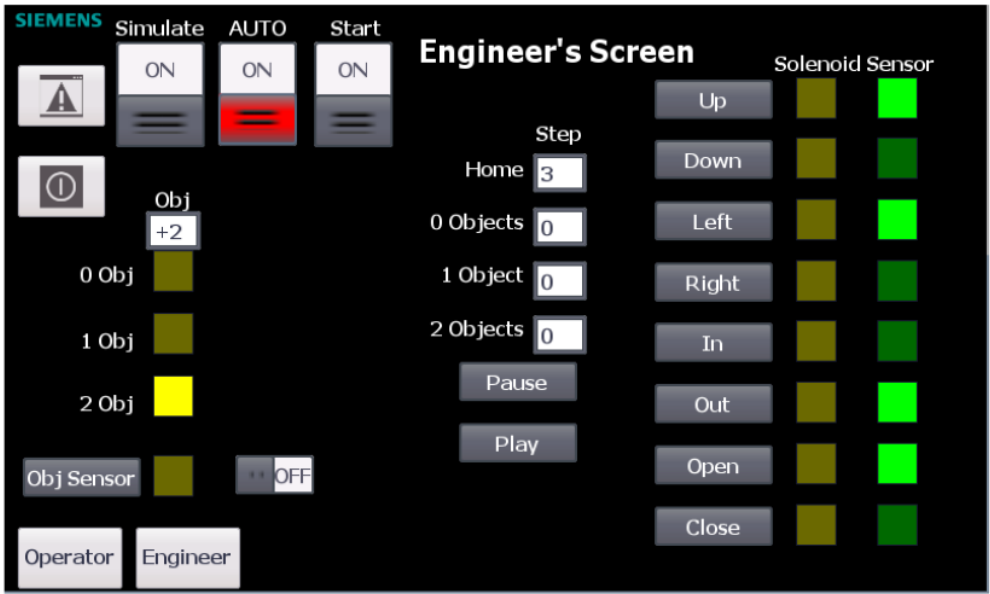


Figure E15

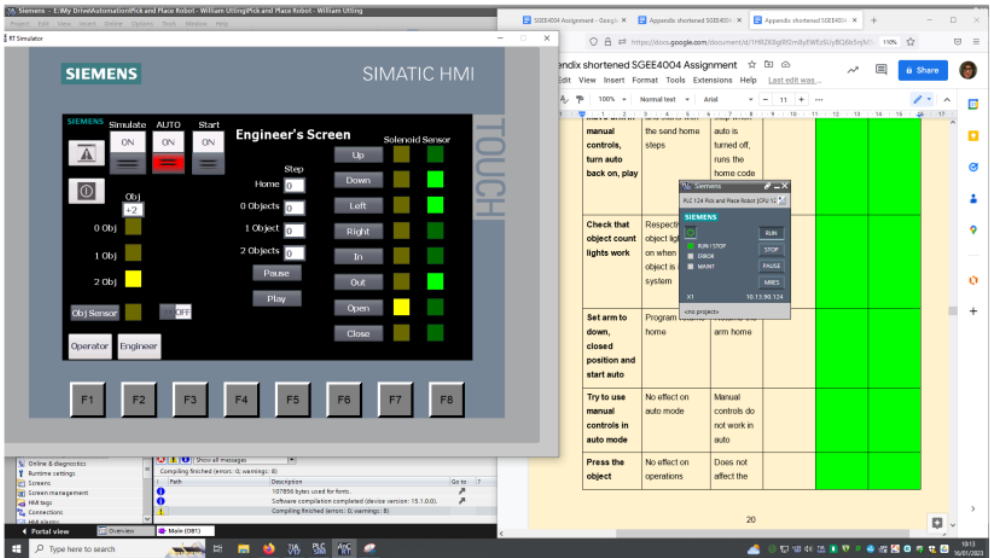


Figure E16 - pressing left button

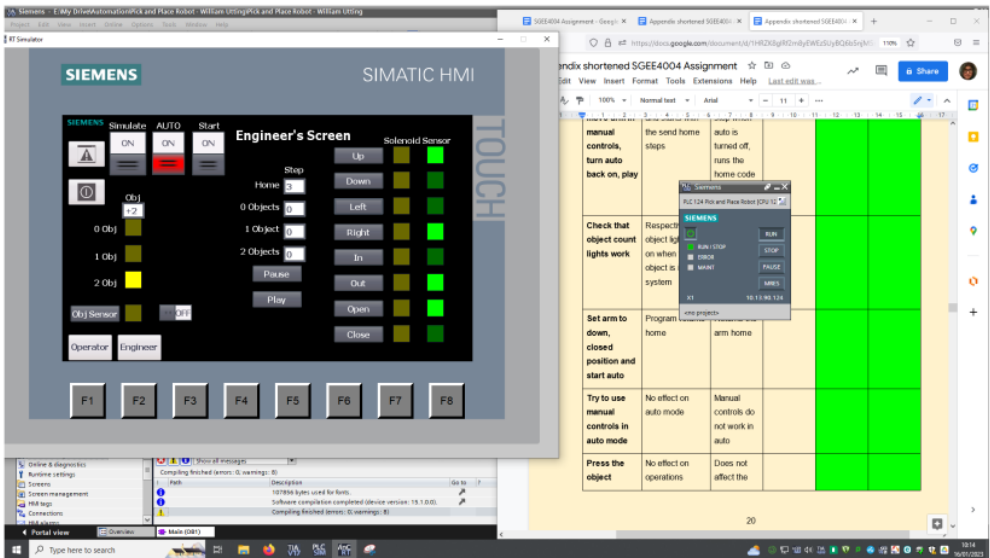


Figure E17

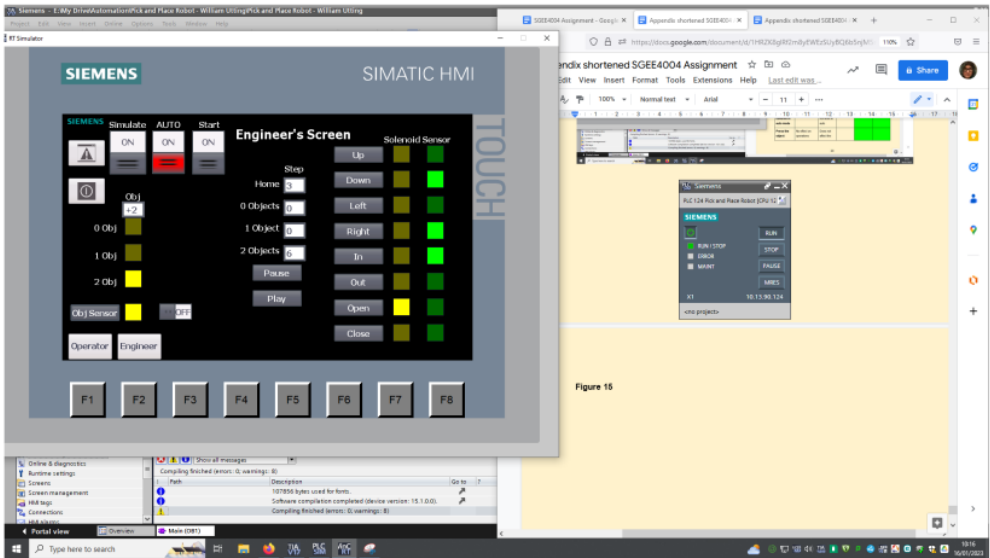


Figure E18 - program goes into start mode

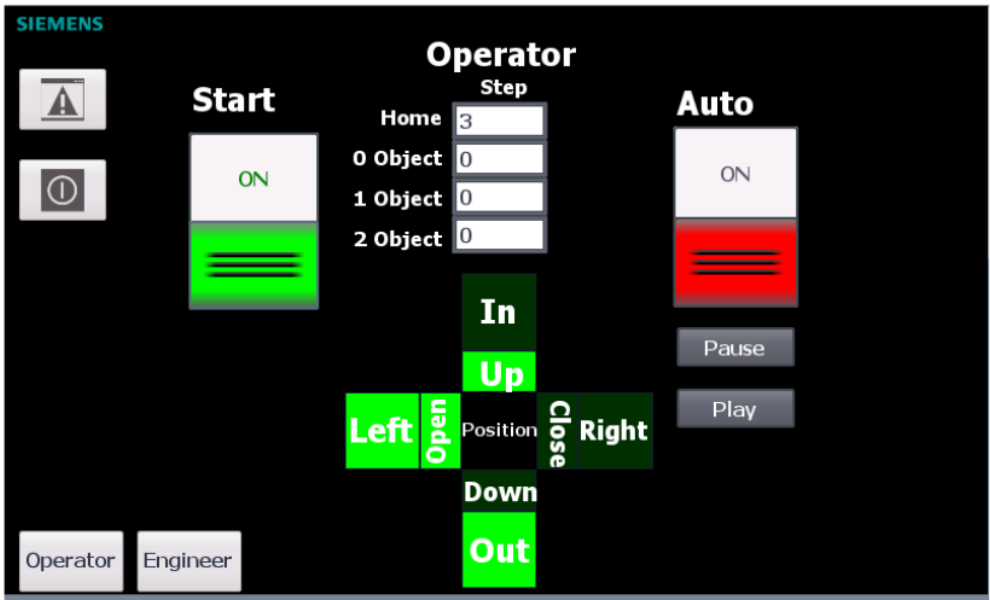


Figure E19

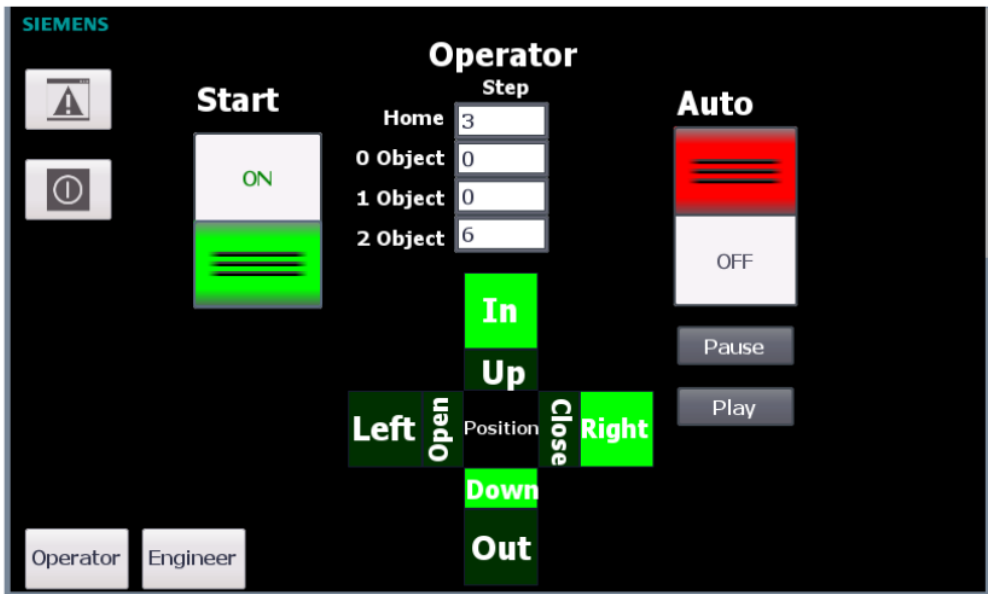


Figure E20 - All correct when arm is home

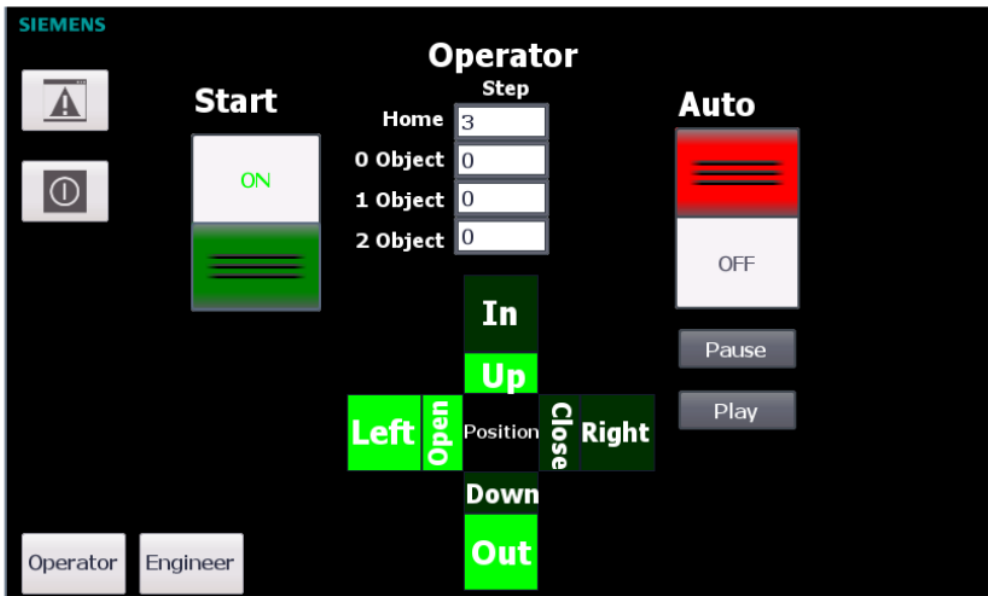


Figure E21 - arm sent to home position



Figure E22 - arm malfunction, picking up in the exit chute

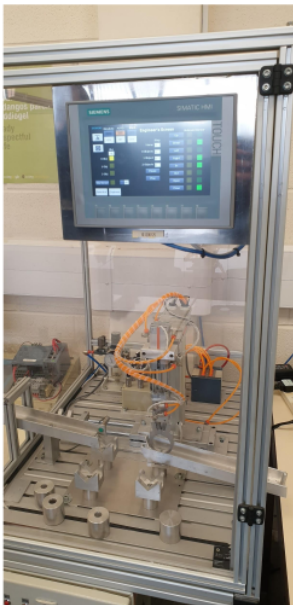


Figure E23 - safety interlocks preventing arm moving

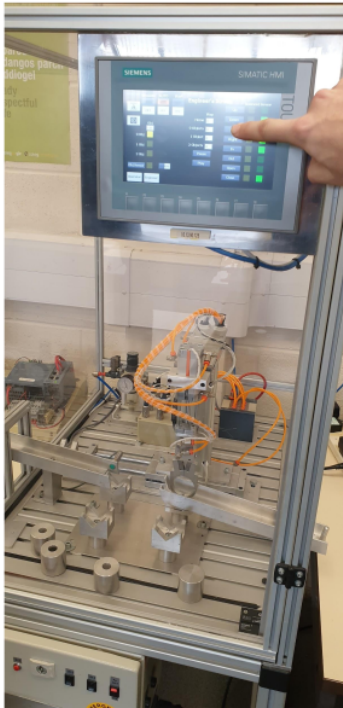


Figure E24 - auto and start switches

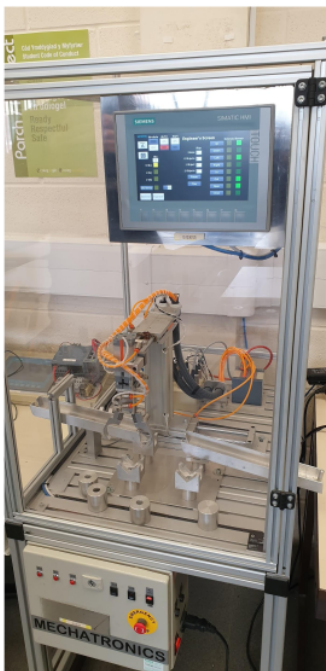


Figure E25 - object insertion error, caused by hardware + start switch

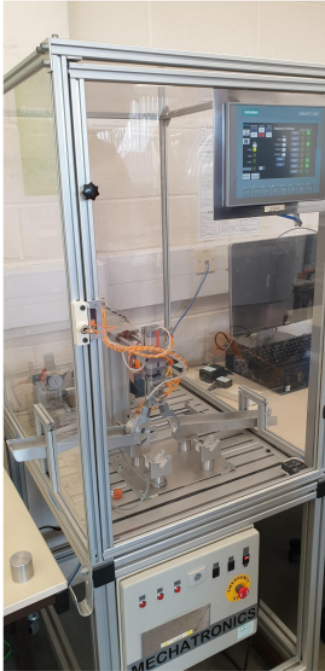


Figure E26 - arm paused while executing 1 object instructions



Figure E27 - object count

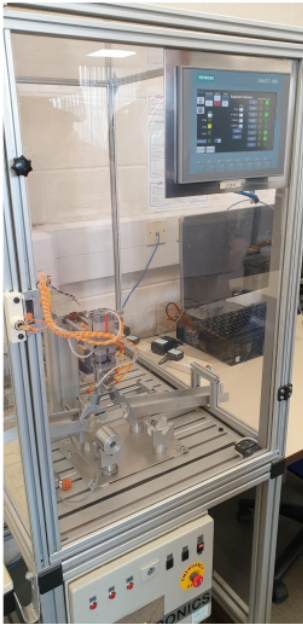


Figure E28 - object sensor hardware issue + 2 objects in system

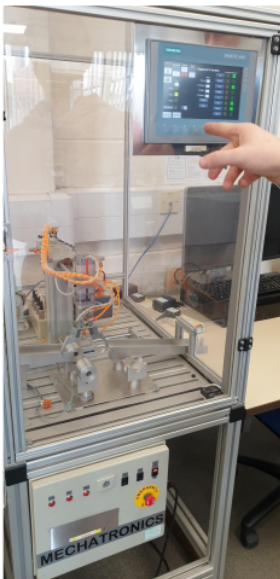


Figure E29 - executing instructions for 2 objects

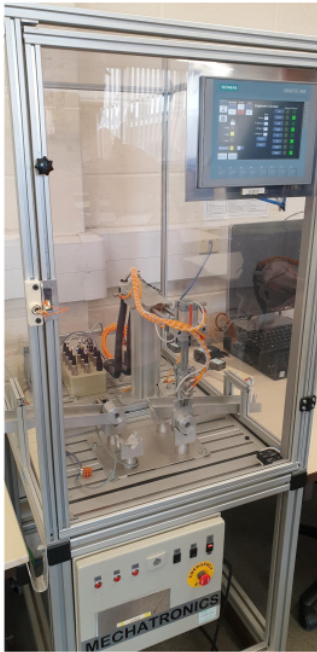


Figure E30 - manual controls

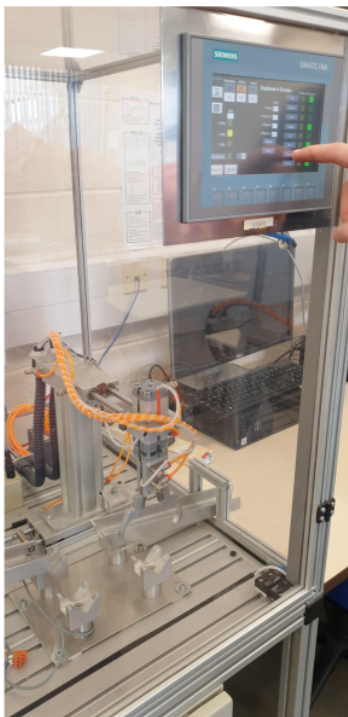


Figure E31 - object sensor button

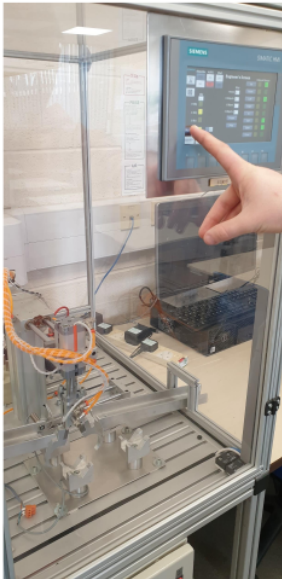


Figure E32



Figure E33 - operator screen, position and switches

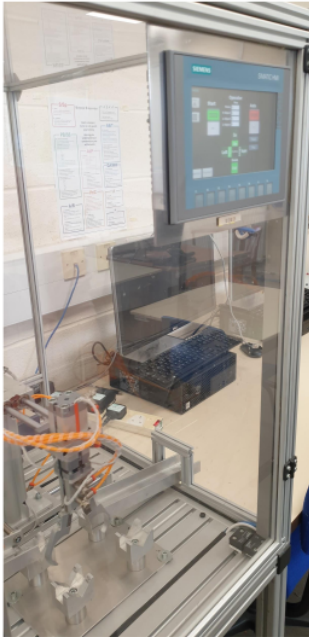


Figure E34 - step number, text boxes need to be larger

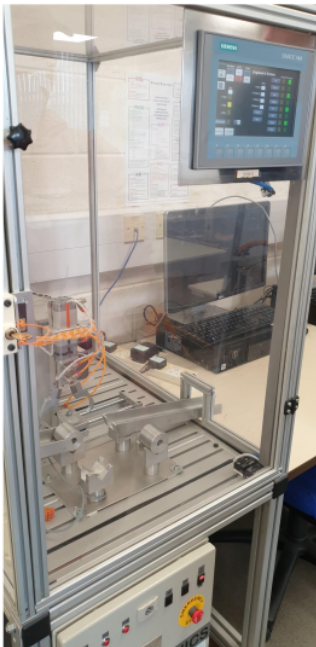
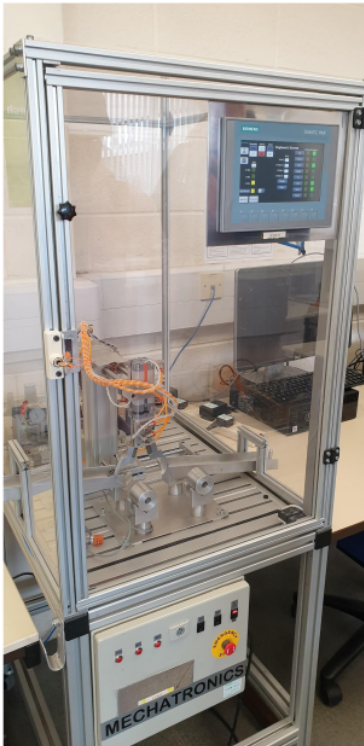


Figure E35 - simulation error



Figure E36



FINAL GRADE

GENERAL COMMENTS

Instructor

75/100

PAGE 1

PAGE 2

QM

Good Structure

Report is structured exactly as stated in the assignment brief.

PAGE 3

Text Comment. Sounds like a good plan

PAGE 4

Text Comment. this citation should contain a comma

PAGE 5

QM

Good conclusions

You have shown understanding here from these observations and deductions

Text Comment. You have presented a reasoned argument

PAGE 6

Text Comment. blank page ?

PAGE 7

QM

Good explanation given

You have clearly put much effort into this explanation, well done.

QM

Good explanation given

You have clearly put much effort into this explanation, well done.

QM

Good explanation given

You have clearly put much effort into this explanation, well done.

Text Comment. start should be a push button not a switch

Text Comment. There are controls on this screen which have not been asked for, you must learn to stick to the design specification

Text Comment. A reasonable range of sources listed correctly

PAGE 23

PAGE 24

PAGE 25

PAGE 26

PAGE 27

PAGE 28

PAGE 29

Text Comment. your ladder code lacks network titles and comments

PAGE 30

PAGE 31

Text Comment. poor image

PAGE 32

PAGE 33

PAGE 34

PAGE 35

PAGE 36

PAGE 37

PAGE 38

PAGE 39

PAGE 40

PAGE 41

PAGE 42

PAGE 43

PAGE 44

PAGE 45

PAGE 46

PAGE 47

PAGE 48

PAGE 49

PAGE 50

PAGE 51

PAGE 52

PAGE 53

PAGE 54

PAGE 55

GRADING FORM: NEW CSG-RUBRIC

WILLIAM UTTING

75

1. ENGAGEMENT WITH LITERATURE



A good research section with evidence of reading from a range of sources cited and listed mostly correctly using the Harvard method. 12/20

12

2. KNOWLEDGE AND UNDERSTANDING



Excellent knowledge and understanding demonstrated in your research findings, hardware investigation, IO list and program design

14

explanations. However, your ladder code contains no titles or comments. 14/20

3. COGNITIVE AND INTELLECTUAL



Good simulation test results analysis, but you did have issues with your practical results yet your discussion and conclusions/recommendations show that you have learnt much from your mistakes. 17/20

17

4. PRACTICAL APPLICATION



A good program design yet too cumbersome and complex as you have stated. Very good simulation and practical testing methods and presentation of results with good attempts to fix your bugs. 17/20

17

5. TRANSFERABLE SKILLS



A good work plan and explanation of team working. Report is well structured, written and presented with clear diagrams. Word count not exceeded. However, your style of explaining your hardware is like a magazine article which is not appropriate for a report like this. 15/20

15