



BEng Electrical & Electronic Engineering
Applied Programmable Logic Controllers
Factory Control

By Aneurin Creed

Lecturer: Dr. Gary Jones

Project submitted as part of the requirements for the award of BEng (Hons)
Electrical and Electronic Engineering

May 2021

Contents

Contents	1
Introduction	2
Literature Review	3
Programmable Logic Controllers	3
Basics of Ladder Logic	4
Planning & Strategy	6
Flowchart Planning	6
Warnings and Alarms Strategy	7
Warnings are required when:	7
Alarms are required when:	7
Solution	8
Factory IO Design & Ladder Logic Programming	8
Layout	8
Counters	12
Warnings & Alarms	15
Human Machine Interface	17
Discussions & Conclusions	18
References	19

Introduction

This report will document the design and control for a simulated factory system of which the main objective is to sort 4 different packages by their size and weight. The proposed factory has been designed using the programme Factory IO, the system is driven by Siemens SIMATIC S7-1200 PLC and Siemens TIA software was used to programme the PLC. In order to achieve the main objective, the system must be able to recognise sensor inputs and respond accordingly without external user input. The factory must contain and control the following as a minimum:-

- 6 conveyor belts
- 2 pushers
- 1 conveyor scale
- 2 Diffuse sensors
- 1 Emergency Stop
- 2 Buttons
- 1 Siren.
- 2 Stack lights

The system has also been designed to incorporate a Human Machine Interface (HMI), with the following displays and controls displayed on the root screen:-

- Buttons for starting and stopping the system.
- An emergency stop button
- A button to reset all counters
- 4 counters
- A replication of the stack lights
- Fault and warning indicators

In addition to this, there are warning and alarm protocols in place in order to alert the user of any misplaced parcels and to indicate the number of parcels that have completed the sorting process.

Literature Review

Programmable Logic Controllers

PLCs can be described as small industrial computers with modular components designed to automate control processes. PLCs are the controllers behind almost all modern industrial automation. [1]

Below are some of the existing PLC models in industry:



Our Bulletin 5069 CompactLogix™ 5480 controllers are real-time controllers with Windows 10 IoT Enterprise running in parallel to the Logix control engine. Our CompactLogix 5480 uses the same re-architected control engine deployed in the ControlLogix® 5580 and CompactLogix 5380 controllers. This line of Logix controllers supports embedded Windows applications, such as analytics, data gathering, and predictive computations. The ability to collect data and make decisions at the machine level helps to support the Connected Enterprise and benefits both end users and OEMs.

Fig 1 - Allen Bradley, CompactLogix 5480 Controller [2]



Memory
16,000 step EEPROM memory built-in. Battery-free and maintenance-free.
(Program capacity is 4,000 steps)
Processing speed
Basic instruction: 0.21 μ s/instruction
Application instruction: 0.5 to 100 μ s/instruction
Device
Auxiliary relay: 1,536 pts Timer: 138 pts Counter: 67 pts Data register: 3,000 pts

Fig 2 - Mitsubishi Electric, FX3s [3]



The modular SIMATIC S7-1200 controller is at the core of our offering for simple but highly precise automation tasks. The SIMATIC S7-1200 controller is modular and compact, versatile, a secure investment, and is perfectly suited to a whole range of applications. The S7-1200 CPUs with Safety Integrated handle both standard and safety-related tasks. A compact design with integrated IO, communication interfaces that meet the highest industry requirements and a range of powerful integrated technological functions make this controller an integral part of a comprehensive automation solution.

Fig 3 - Siemens, SIMATIC S7-1200 [4]



Modicon IIoT-native edge controllers manage complex interfaces across assets and devices or directly into the cloud, with embedded safety and cybersecurity. Modicon provides performance and scalability for a wide range of industrial applications up to high-performance multi-axis machines and high-available redundant processes.

Fig 4 - Schneider Electric, M241 24 [5]

A table outlining the features of each device.

PLC	Allen Bradley CompactLogix 5480 Controller	Mitsubishi Electric FX3s	Siemens SIMATIC S7-1200	Schneider Electric M241 24
Image				
Features	-Uses a full version of windows 10 for its operating system -3MB-20MB of memory. -Up to 31 local Bulletin 5069 Compact I/O™ modules	- Depending on the purpose, various special function blocks can be connected -Function can be easily added by mounting a function expansion board to the main unit.	-Communication modules increase the versatility of it offering several open communication protocols as serial communication. -Technological integrated functions such as HSC, PTO/PWM and Motion Control.	-IP20 with protective cover in place. -Non isolated serial link serial 1 with RJ45 connector and RS232/RS485 interface. -128 MB built-in flash memory for backup of user programs

Basics of Ladder Logic

Ladder logic is primarily used as the language to programme PLCs. This is because early control system designers used relay control circuits and so preferred using ladder diagrams which are similar in design, especially when compared to text based programming such as C, BASIC or Pascal. Ladder diagrams are also used due to factory maintenance staff being familiar with relay control circuits, meaning that any control system problems were easy to troubleshoot. [6]

There are several differences between ladder logic diagrams and relay logic circuits; firstly, ladder logic diagrams use symbolic notation to express logic operations whereas relay logic circuits use circuit components. Secondly, rails are used in both - in relay logic circuits they represent the supply wires of the circuit. However, in a ladder diagram, rails represent the start and the end to the line of symbolic code. Finally, rungs represent the wires connecting the components together within a relay circuit, whereas the rungs within a ladder logic diagram represent the logic as it flows through the symbolic code. [6]

Alongside the rails and rungs, there are also several other basic parts used within ladder logic diagrams that are important to be aware of. These parts are: [6]

- Inputs - generally, these are physically hardwired to the PLC terminals and are externally controlled actions which are usually triggered by something like a push button being pushed. However, an input can be the result of an output being high, or a memory location being high. This results in a software version of a physical action where either an output or memory will then

cause the input to be true. Inputs are represented within a ladder diagram by either a normally open (NO) contact or normally closed (NC) contact symbol.

- Outputs - these are usually external devices that are physically hardwired to the PLC but can also be digital outputs such as a human machine interface (HMI) display component or, as previously mentioned, a memory location. Outputs are turned on if the logic along a rung connects the start rail to the output. Outputs are represented within ladder logic diagrams by a relay coil symbol.
- Logic expressions - these formulate desired control operation depending on the combination of inputs and outputs. For example, if a rung contains a single input and a single output this can be considered as an 'IF' function, where if the input is true or false 'THEN' the output is as a result also true or false. Given that two inputs on a rung are in series, this implies that both have to be true and thus an 'AND' function has been used. Given that two rungs each with an input are connected to one output, only one of the inputs has to be true to connect the output to the start rail, thus 'OR' logic is applied.
- Address notation and tag name - the address is used to describe the memory location of a tag and the tag name given is the description given to an address.

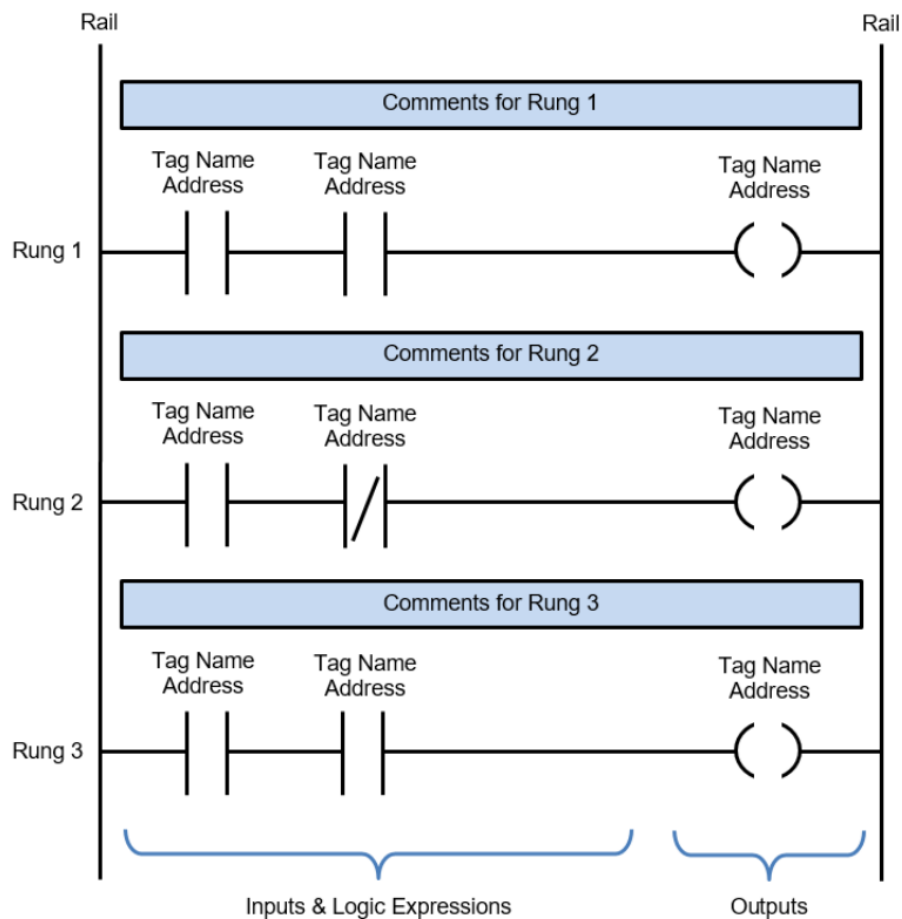


Fig 5 - parts of a ladder logic diagram [6]

Planning & Strategy

Flowchart Planning

Below is a flow chart which was used as an outline for the planning process. It demonstrates the key points where the system must acknowledge and respond accordingly. The flowchart was a tool used when designing the ladder logic that controls the factory system. Within the flow chart there are 7 diamonds that represent the ladder logic looking for a signal from the factory system. At each of these diamonds a decision is made depending on the factory's systems status. The status along with the decision of the program chooses the output necessary.

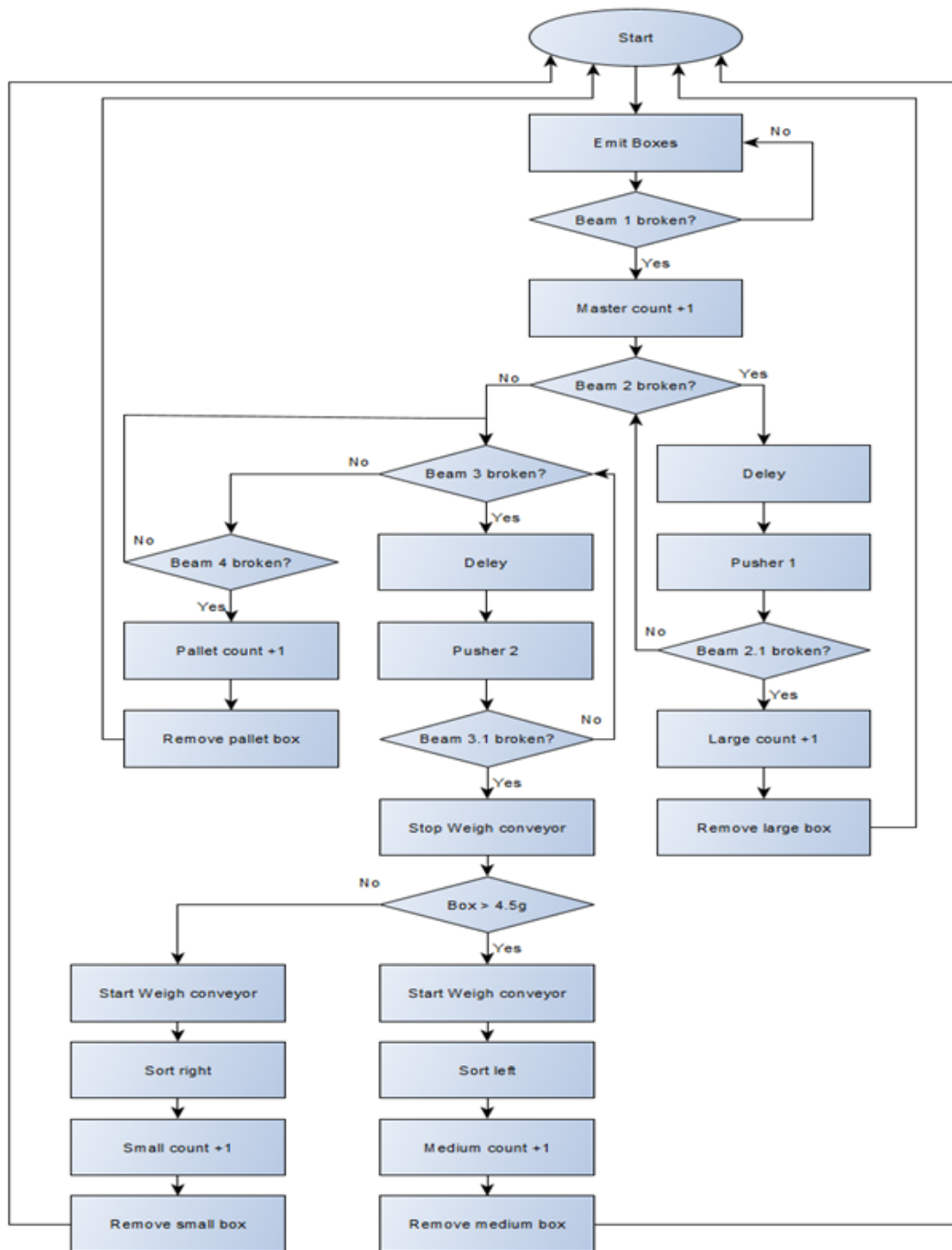


Fig 6 - Flow chart

Warnings and Alarms Strategy

The purpose of the warnings and alarms was to enable the user to set the system's parameters and leave the system to run automatically. If a warning is triggered, then this indicates that an error has been recognised but production can continue or human intervention may be required soon. If an alarm is triggered, then this results in an immediate emergency stop of the full system, this is to prevent any possible further action.

The factory was specified to sort 4 different packages by size and weight. This specification results in 4 separate final outputs so 4 different package bins. It is the factory system's role to ensure that the packages are sorted correctly. Presumably, the factory could do this with an infinite number of packages. However, the user needs to input exactly how many packages and what amount of each different package the system sorts. Thus, a strategy was developed.

Warnings are required when:

Package bins reach an almost full level: - The level amount can be inputted by the user. A yellow warning light near the location of each package bin will illuminate and a master light at the control unit will change to indicate when a package bin has reached the almost full level.

A minimum number of packages are misplaced: - Each section of the factory that changes the route of a package will require a warning protocol that changes the master light to yellow, indicating when a minimum number of packages has become unaccounted for.

A minimum difference to packages accounted for is exceeded: - This follows on from the above warning protocol and measures the difference in number of boxes emitted to the number of boxes accounted for. If the difference exceeds the minimum amount the master light changes to yellow.

Alarms are required when:

Package bins reach full level: - The level amount can be inputted by the user. A red alarm light near the location of the package bin will illuminate and a master light at the control unit will change to indicate when a package bin has reached the full level, as well as an immediate emergency stop of the full system.

A maximum number of packages are misplaced: - Each section of the factory that changes the route of a package will require an alarm protocol that changes the master light to red, indicating when a maximum number of packages has become unaccounted for. This will also trigger an immediate emergency stop of the full system.

A maximum difference to packages accounted for is exceeded: - This follows on from the above alarm protocol and measures the difference in number of boxes emitted to the number of boxes accounted for. If the difference exceeds the maximum amount, the master light changes to red as well as triggering an immediate emergency stop of the full system.

Three separate warnings have been triggered:- If three separate warnings have been triggered then human intervention is required as there are several faults that have been recognised by the system.

Solution

Factory IO Design & Ladder Logic Programming

Layout

Using Factory IO, a virtually simulated factory has been designed. The factory consists of:

- three 6m conveyor belts
- three 2m conveyor belts
- two straight spur conveyor belts
- one curved belt conveyor
- one inclined conveyor
- one conveyor scale
- one pop up wheel sorter
- two pushers
- one emitter
- four removers
- five stack lights
- one alarm siren
- thirteen diffuse sensors
- an electric switch board with three buttons; start, stop and emergency stop
- multiple aligners alongside the conveyors to guide the moving parcels.

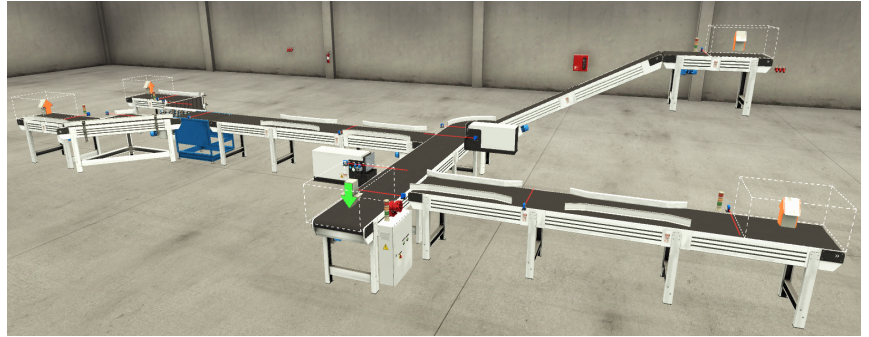


Fig 7- Factory Layout

Firstly, the conveyor route was laid out as seen in fig.7. The layout needed to accommodate the split of the incoming stream of random parcels into their 4 different box types; pallet, small, medium and large, into four designated box bins.

Fig 8 shows the initial network that starts the factory. For this network to be functional, two buttons had to be added to the factory's design. These buttons were to start and stop the factory. The network in Fig 8 shows four inputs and six outputs. The two 'NO' contact inputs are in parallel with one another and the 'NC' contacts are in series with both of them. If the 'NO' contact 'Start button(start1)' is high then the connection from the rail will go through the two 'NC' contacts 'Stop button (start2)', 'Alarm rep' and reach the output coils. If the output coil 'mem 1' is high then this causes the second 'NO' contact to also be high, meaning the output coil 'mem 1' will stay high. If either the 'NC' contact 'Stop button (start2)' or 'Alarm rep' is triggered, then the connection to the outputs will be disconnected making 'mem 1' low. The use of 'mem 1' in this way is known as a 'latch'. This start network is used to turn on and hold 'mem 1' high, until the user turns it off. All of the other output coils are used to turn each stack light green to signify the system is running.

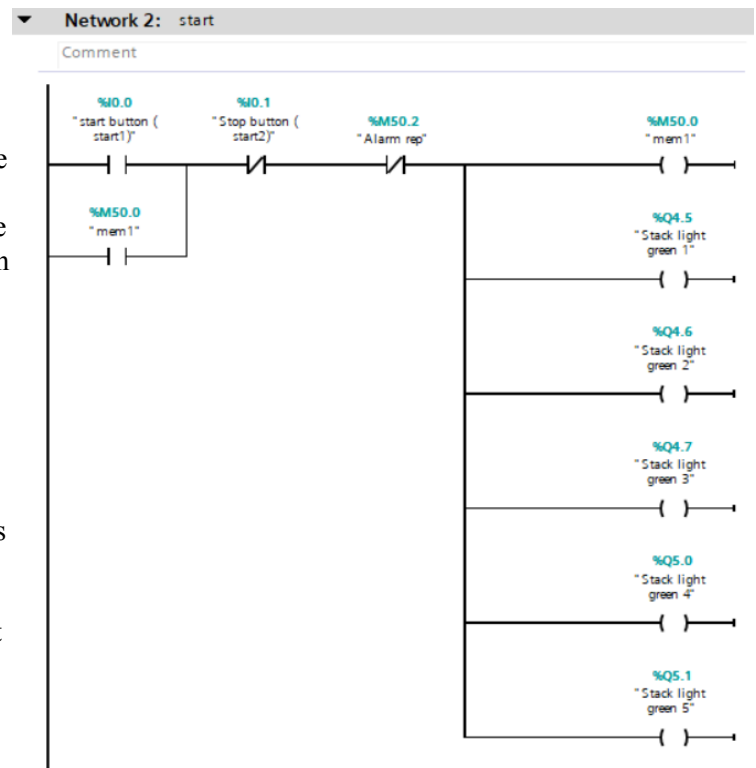


Fig 8 - Start Network

This start network provides a way to control the on/off status of the factory. Network 3 shown in fig 9 uses the output 'mem 1' as the tag for the only input. When this 'NO' contact is closed due to 'mem 1' being high it causes the alarm siren to sound for 2 second and starts the first 6m belt conveyor.

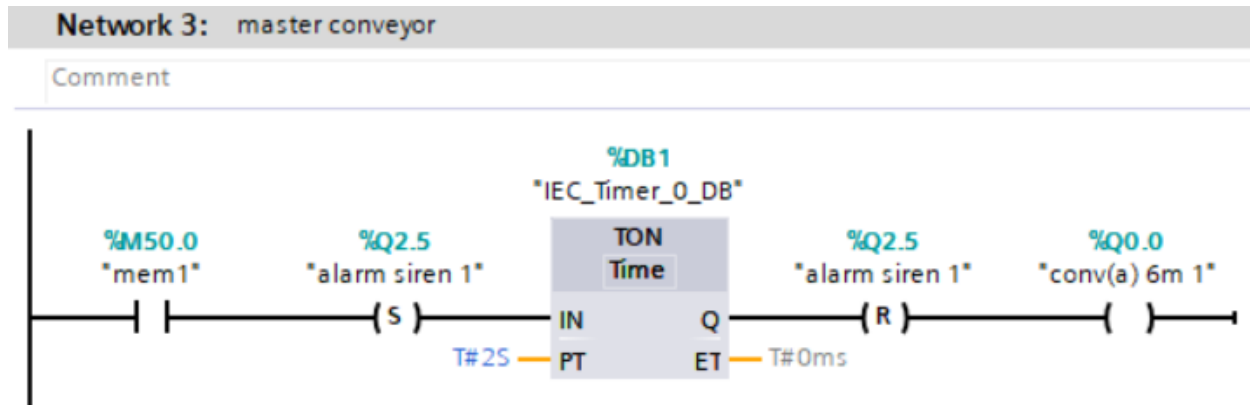


Fig 9 - Master Conveyor Network

By using the output from network 3 as the input in network 4 shown in fig 10, all of the conveyors starting depends on the status of 'mem 1'. When 'mem 1' becomes high due to the button within Factory IO being pressed, this causes the first 6m belt conveyor to start and, subsequently, all other conveyors start too. However, the conveyors after the first 6m belt conveyor do not all start at the same time. This is achieved by placing timers in the rung before each output coil. This has limited the current draw during the start up of the factory.

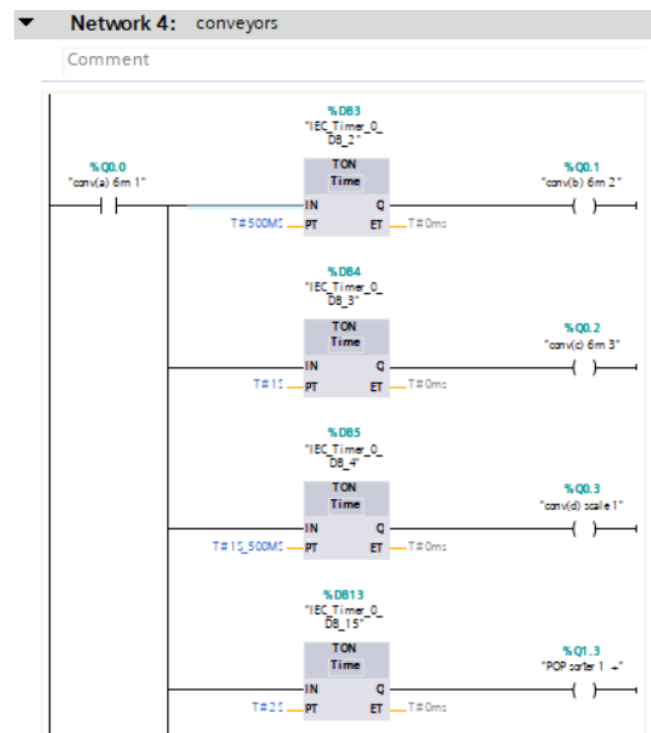


Fig 10 - Conveyors Network

At the beginning of the first conveyor there is an emitter, which releases a random box every three seconds. The boxes then traverse along the first 6m conveyor towards a diffuse sensor which has been placed purposefully at a height so only the large boxes can be detected by the sensor. When a large box triggers the sensor it is then pushed by the first pusher onto a separate conveyor belt and towards a designated large box sorting bin, represented within Factory IO by a remover. Network 5 is what controls this process, as shown in fig 11, both 'mem 1' and 'diffsensor 1' need to be triggered in order to activate the output coil 'pusher 1'. The signal reaching pusher 1 is controlled by timers to ensure pusher 1 pushes at the right time.

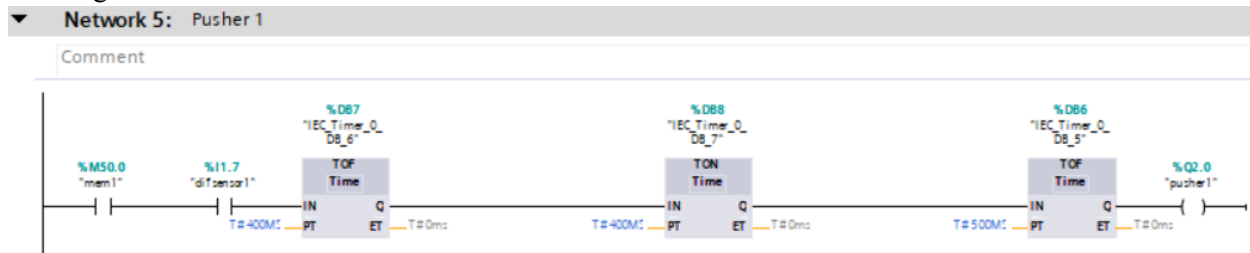


Fig 11 - Pusher 1 Network

Before the second pusher there is another diffuse sensor; this one is positioned slightly lower so only the medium and small boxes trigger it. When triggered, the same process as discussed above is implemented but for pusher 2.

When the small and medium boxes are pushed onto their separate conveyor they travel along it towards a scale conveyor, as shown in fig 12. The scale conveyor is used here to weigh each parcel and subsequently split the small and medium boxes by their weights. To do this the scale must stop momentarily when a parcel is on it. It accomplishes this using a diffuse sensor which is positioned at the midway point of the conveyor scale. When the diffuse sensor is triggered, it causes the conveyor scale to stop momentarily so it can record the weight of the parcel accurately.

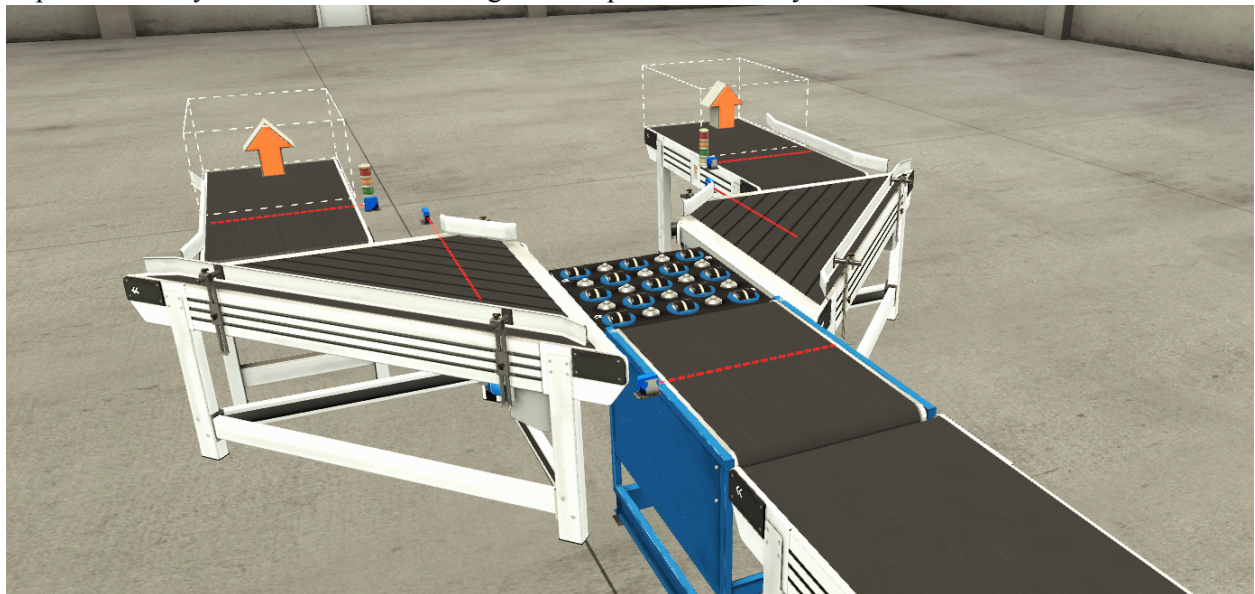


Fig 12 - Scale Conveyor & Pop Up Wheel Sorter

The scale conveyor does not measure the weight of the package. Shown in fig 13 is the driver tag location of the conveyor scale. This input requires a 'D' location as the value recorded is of type 'Real' meaning it required a DWORD input. The value produced by the conveyor scale is actually a voltage between 0V and 10V.

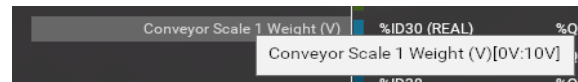


Fig 13 - Factory IO Conveyor Scale Driver Location

Fig 14 shows the first half of network 7. This image shows the input contacts 'mem 1' and 'diffsensor 7', both of which must be high for the reset coil 'conv(d) scale 1' to be activated causing the conveyor scale to stop. When stopped the 'NC' contact can 'move' the input voltage value from the scales to a memory location called 'Weight'. This all happens within half a second due to the 'TON' timer which continues the connection onwards.

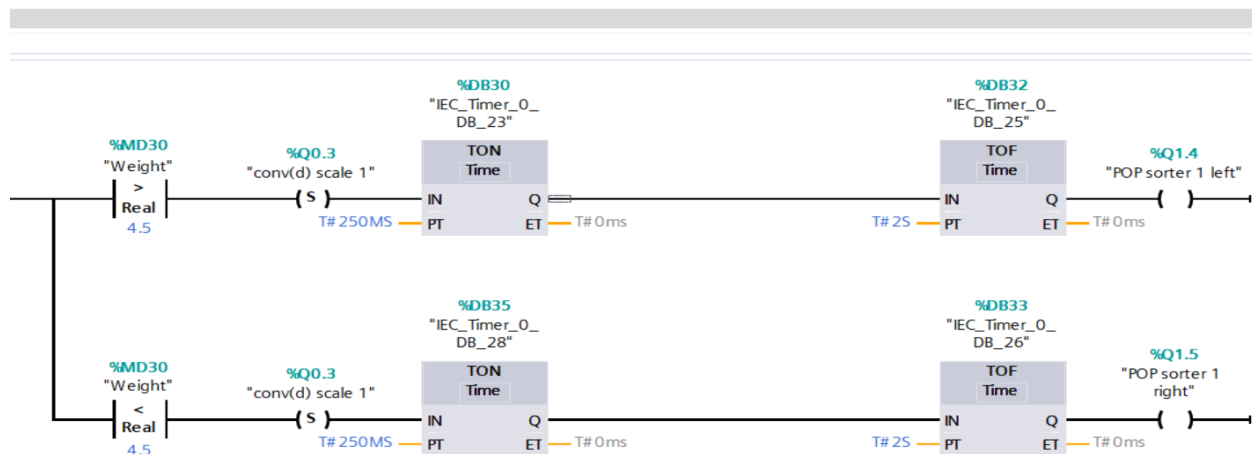


Fig 14 - Storing Scale Input In Memory Location

Fig 15 shows the second half of network 7, where the 'Weight' value stored in a memory location is compared to the number 4.5. If it is greater than 4.5 then the conveyor scale starts and the parcel is moved to the left via the pop up wheel sorter. If the value is less than 4.5 then the conveyor scale starts and the parcel is moved to the right via the pop up wheel sorter. After the pop up sorter, the parcels are moved by two straight spur conveyor belts, onto two 2m conveyor belts which move each parcel to their designated box bin, represented within Factory IO by a remover.

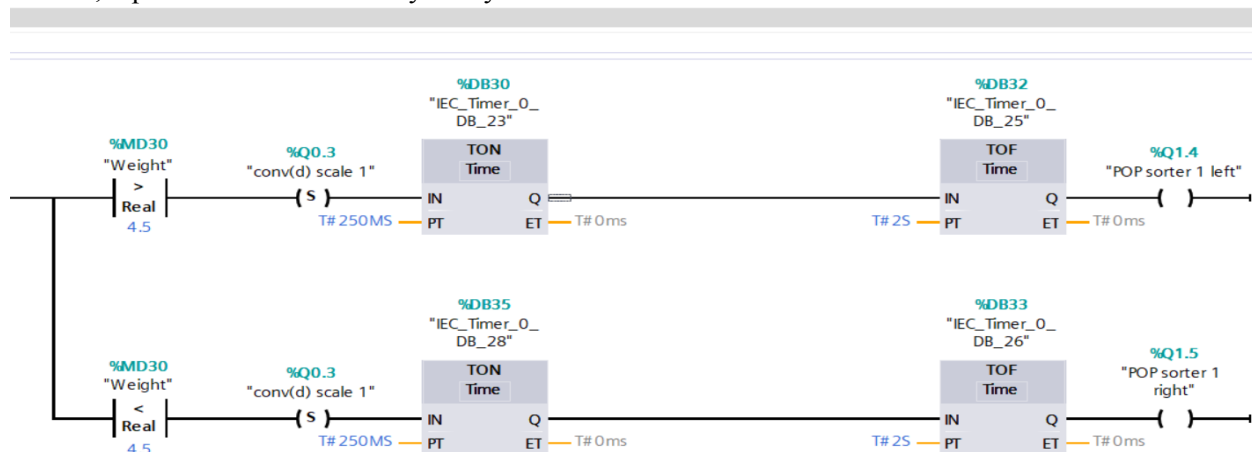


Fig 15 - Comparing Weight Value To Sort Parcels

The pallet boxes pass underneath both of the diffuse sensors used to detect the large, medium and small boxes. This means that after the small, medium and large boxes have been sorted onto separate conveyor belts the only boxes to get passed the two pushers are the pallet boxes. This means that there is no need to move the pallet boxes onto a separate conveyor line and they can continue along the same route to their sorting bin. Before the pallet boxes reach their designated pallet box bin, represented within Factory IO by a remover, they are turned by a curved belt conveyor and raised by an inclined conveyor, as shown in fig 16. The reason the pallet boxes route turns and raises the boxes is to allow for future use of a palletiser designed to stack pallet boxes efficiently onto a pallet.



Fig 16 - Pallet Corner & Incline

Counters

Once the program could successfully sort the four different boxes, the next challenge was to count the parcels continuously throughout their journey. First, the incoming parcels are counted - this is achieved by using a diffuse that is positioned just after the emitter. This sensor is triggered as every parcel passes it. Fig 17 shows network 8 'Master Counter', and how it is used to count the parcels. When the contact 'diff sensor 3' is triggered it connects the start rail through a 'TOF' timer to the input of a counter. This results in the counter increasing every time the diffuse sensor is triggered. A memory location, 'counter reset', was created which was linked to all of the reset inputs of the counters. This enabled the ability to manually reset the counters value. In addition to this, another memory location 'Master count' was created to store the value of this counter.

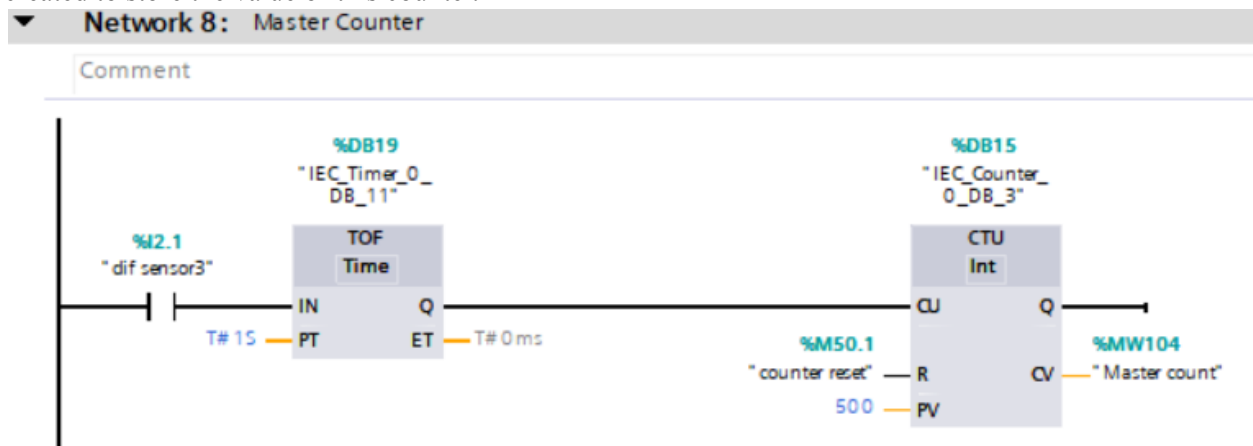


Fig 17 - Master Counter Network

A similar network was programmed to count the number of times each pusher activated. As shown in fig 18, the input to the counter here is linked to the contact 'pusher 1', this results in the counter incrementing every time the pusher pushes. This process is identical for the second pusher.

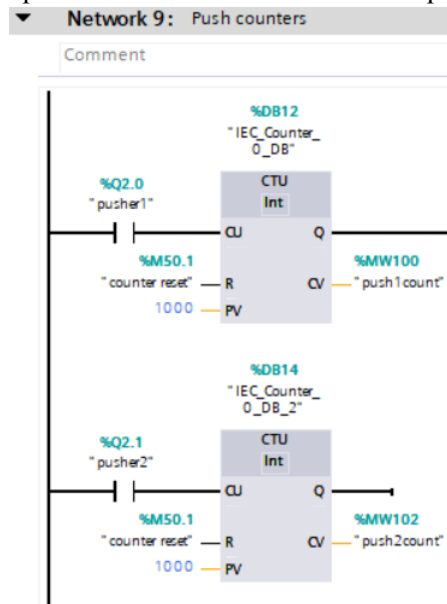


Fig 18 - Push Counters

Fig 19 shows the networks used to count the parcels once they have been separated. Each network is programmed the same as they all require the input coil to go high, this is dependent on the specific diffuse sensor for the individual package. When the 'NO' contact goes high it sends the signal towards the respective counter for that package. Each counter uses the same reset input to reset the count value. For each counter, an individual memory location has been created to store the counted value.

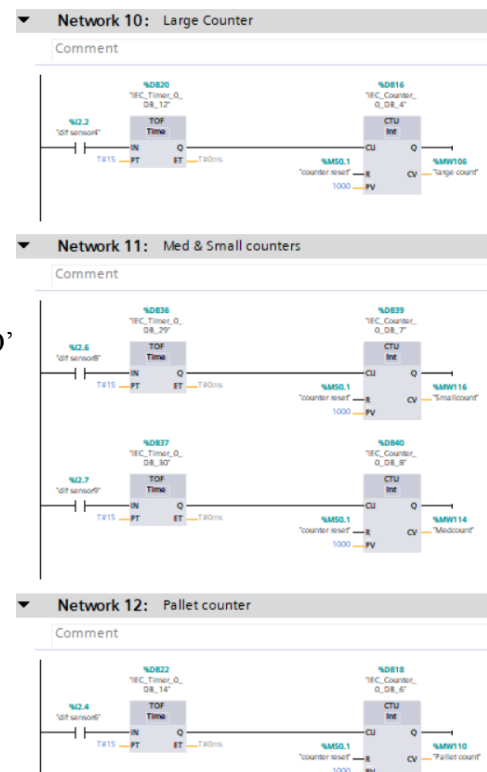


Fig 19 - Box Counter Networks

When all the boxes reach their respective removers they are also counted using individual diffuse sensors. Fig 20 shows how the counters were programmed to count the parcels as they are being removed. This was to compare the number of each type of parcel successfully completing its sortation to the number of each parcel that has been sorted.

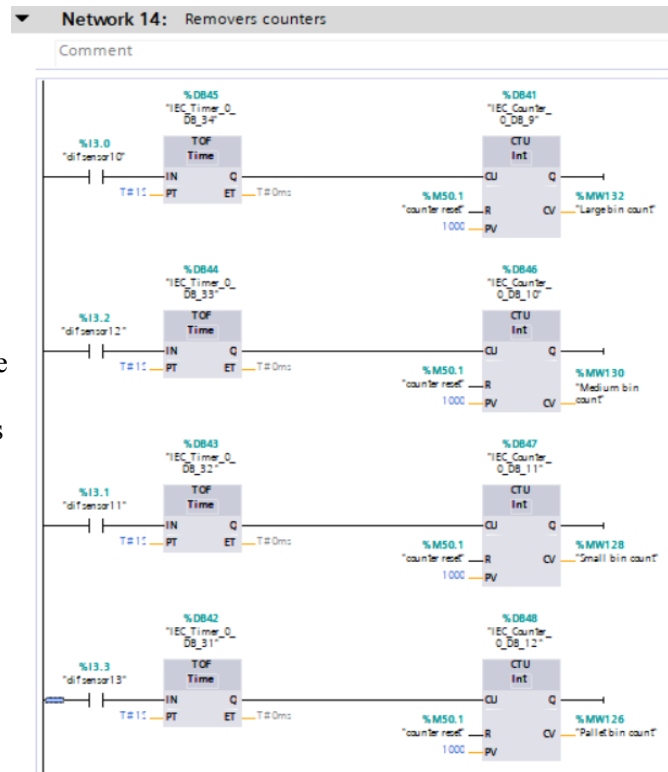


Fig 20 - Removers Counters Network

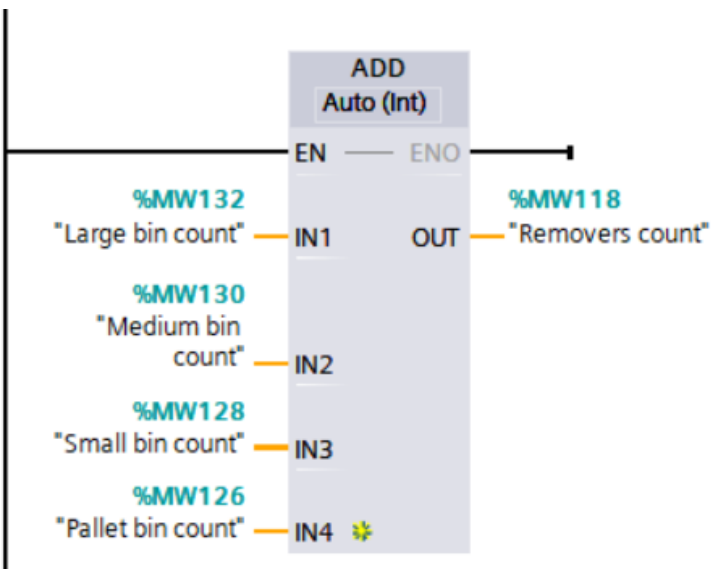


Fig 21 - Removers Count

Fig 21 shows how a math function has been used to add the four bin count values to give a total bin value called 'Removers count'. This value can be used to see the total number of parcels which have reached all the parcel bins.

Warnings & Alarms

Once all the counters had been programmed, they could then be used to implement warnings and alarms.

The first warning that was programmed looks at the number of large boxes that have reached the bin. As shown in fig 22, if the count value is greater than or equal to the value of 'Large bin warning limit', then the signal will reach the two output coils, 'Warning 1' and 'Stacklight yellow 5'. 'Warning 1' tag is a memory location that is used later to identify to the user that a warning has been triggered. 'Stacklight yellow 5' is an output connected to a stack light positioned at the end of the conveyor belt leading to the large remover, this stack light turns yellow to signify where the warning has occurred. This warning protocol is the same for all of the box sizes, with their respective stack light and own specific warnings.

Fig 23 shows the alarm protocol for 'Alarm 1'. It follows on from 'Warning 1' but the value that the 'Large bin count' is compared to is 'Large bin alarm limit'. When the count value is greater than or equal to the value of 'Large bin alarm limit' then the signal will reach the two output coils, Alarm 1' and 'Stacklight red 5'. As with the warning, 'Alarm 1' is a memory location that is used to signify to the user that an alarm has been triggered. 'Stacklight red 5' is connected to the same stack light for the warning, but turns the light red to indicate where the alarm has occurred. This alarm protocol is the same for all of the box sizes, with their respective stack light and own specific alarms.

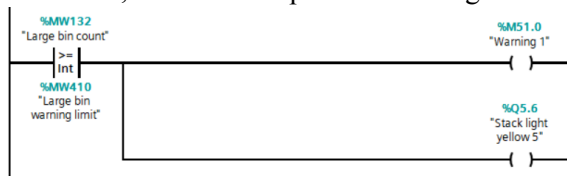


Fig 22 - Bin Count Warning

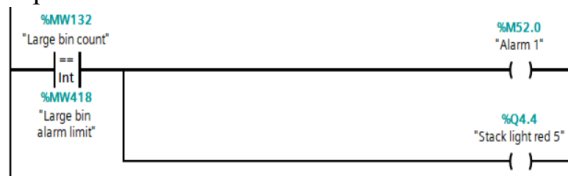


Fig 23 - Bin Count Alarm

Fig 24 shows how 'Warning 5' was programmed. Using a math function as the initial input it subtracts the 'Large bin count' from the 'large count' and the difference is stored in a memory location called 'Large diff'. Then 'Large diff' is compared with the preset integer 2, if 'Large diff' becomes greater than 2 then 'Warning 5' and 'Stacklight yellow 5' are activated. This warning protocol detects if boxes fall and become unaccounted for. This protocol is applied to all of the boxes with their respective warnings and stack lights.

Fig 25 shows the alarm protocol for the box differences. However, the box difference has to be greater than 3 to trigger the alarm. Due to a limited number of available alarms, only one box size has to exceed the preset amount and the same alarm, 'Alarm 6' is triggered.

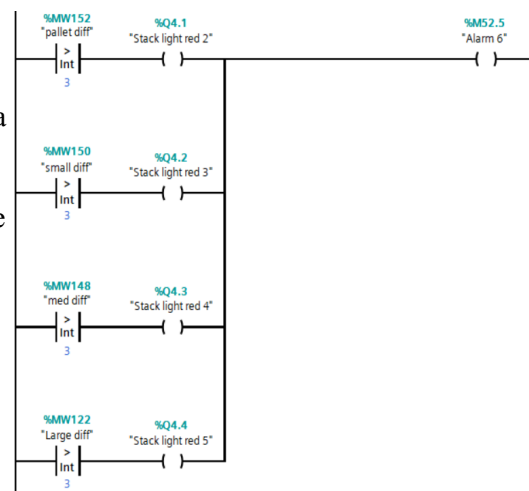


Fig 25 - Box Difference Alarm

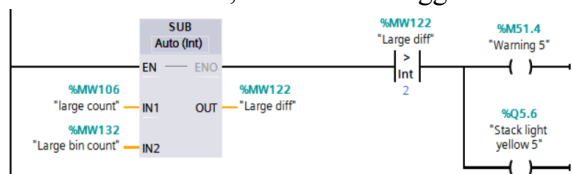


Fig 24 - Box Difference Warning

If any warnings are triggered then the comparator in fig 26 becomes high and connects the signal to the outputs 'Warning rep' and 'Stacklight yellow 1'. 'Warning rep' is used to signify that a warning has been triggered. 'Stacklight yellow 1' is connected to the stack light positioned at the start of the factory where the boxes are emitted. This is to signify that a warning has been triggered.

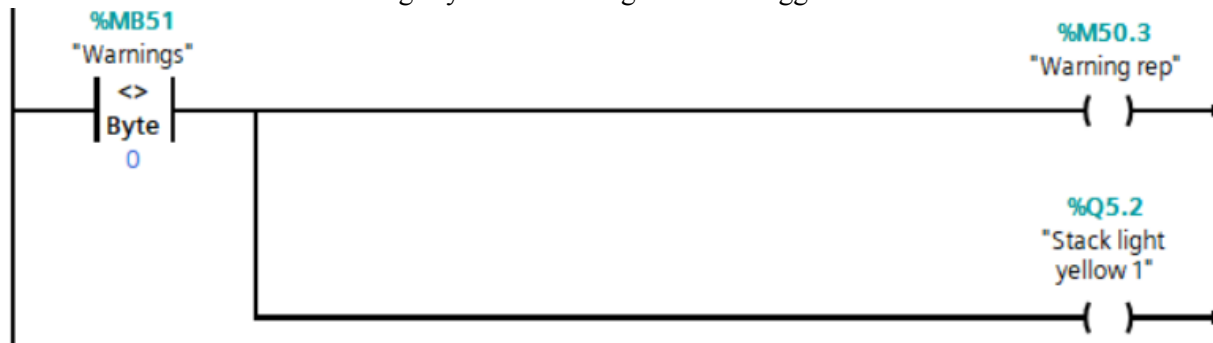


Fig 26 - Warning Repeater

Fig 27 shows the programming for 'Alarm 7', if either 'E-Stop' or 'HMI E-Stop' are triggered then this connects the start rail to the output 'Alarm 7', 'Alarm 7' signifies .

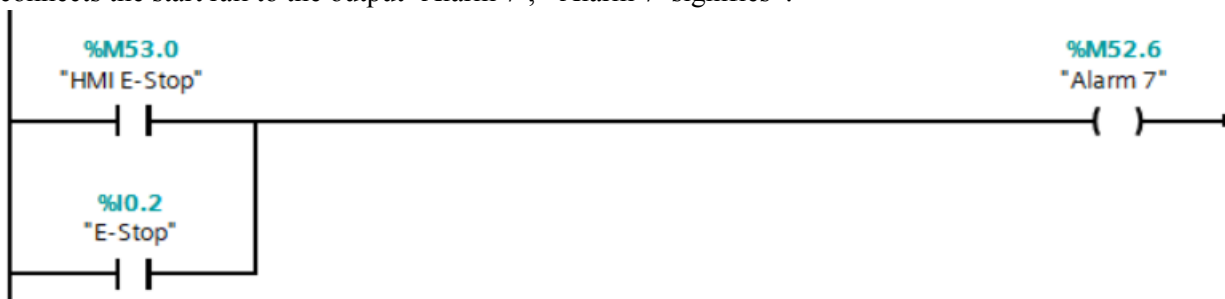


Fig 27 - E-Stop, HMI E-Stop

If any alarms are triggered, then the comparator in fig 28 becomes high and connects the signal to the outputs 'Alarm rep', 'Alarm 8 Master' and 'Stacklight red 1'. Alarm rep' is used to signify that an alarm has been triggered. 'Alarm 8 Master' is the eighth alarm output and has to be manually deactivated by a user. 'Stacklight red 1' is connected to the stack light positioned at the start of the factory where the boxes are emitted, this is to signify that an alarm has been triggered.

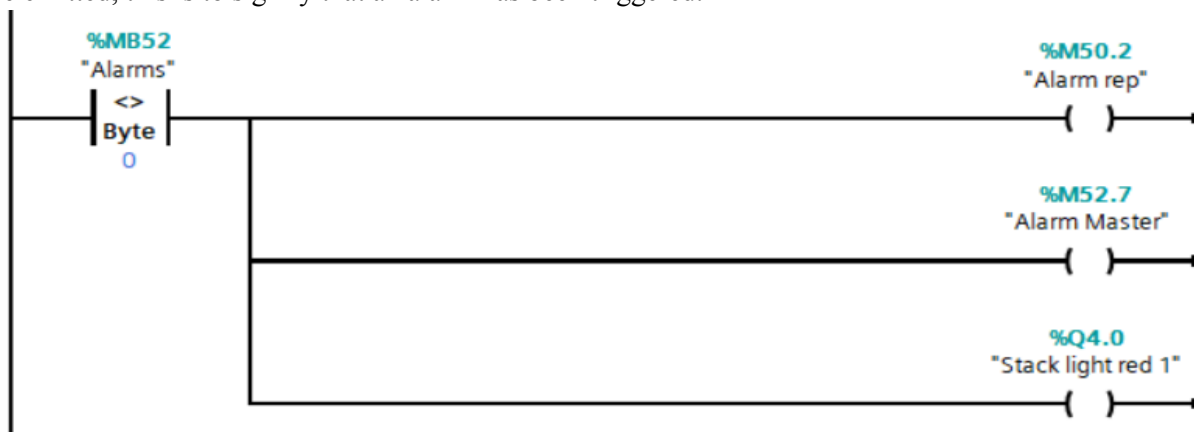


Fig 28 - Alarm Repeater

Human Machine Interface

Fig 29 shows the HMI screen, 'Root Screen'. This screen is the main control interface for the operator and allows the user to start and stop the factory by pressing either the 'Start' or 'Stop' buttons. The 'Start' and 'Stop' buttons will illuminate to indicate that they have been activated. This screen also allows the operator to reset all of the counters within the program by pressing the 'Counter Reset' button. There are two buttons, 'Warnings' and 'Alarms', that will indicate if a warning or alarm has been triggered. When pressed, they also activate the screen 'Warnings and Alarms'. There is a navigation button called 'Engineer', this activates the 'Engineer' screen. There are three 'Status' lights on the HMI that correspond with 'Stack light 1' as it is activated within the factory. There are several labelled counters on the screen which display the counter values within the program.

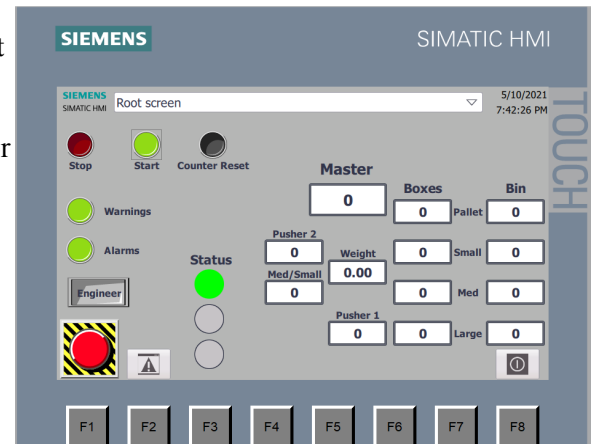


Fig 29- Root Screen

Fig 30 shows the HMI screen 'Warnings and Alarms'. This screen is used to show which warnings or alarms have been triggered by illuminating a labeled button. As explained previously, if any alarm is triggered then the 'Alarm Master' is also triggered. This has been done purposefully as when a situation causes an alarm to be triggered, an operator then has to identify and correct the issue, which will cause the alarm light to turn off. However, the 'Alarm Master' will stay illuminated until the operator has pressed it - this ensures that any potential hazards have been addressed. This screen also has two navigation buttons that activate the respective screens.

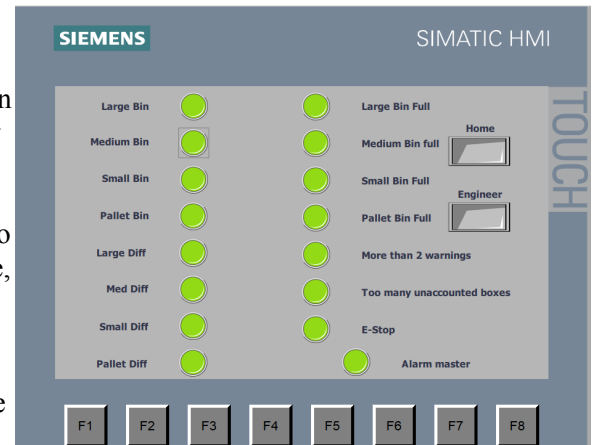


Fig 30 - Warnings and Alarms Screen

Fig 31 shows the HMI screen 'Engineer'. This screen has eight data entry points which allow the operator to pre-set the values at which the warnings and alarms will activate for the number of boxes in a particular bin. It also has a button which when pressed will activate the 'Root Screen'.

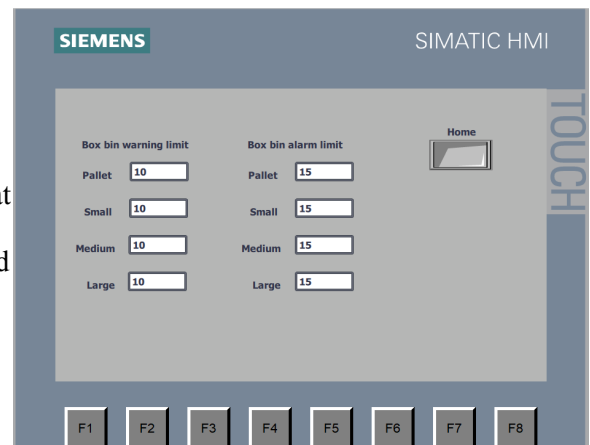


Fig 31 - Engineer Screen

Discussions & Conclusions

This virtual factory design was successful in sorting incoming boxes into four different parcel bins. However, in reviewing its final outcome there are some improvements worth discussing. Firstly, one of the alarm aims was that if at one time three separate warnings were triggered then this would activate an alarm. This was not achieved as a solution to this was not found. Going forward this would be a vast improvement to the factories operational safety and function, as this could help to mitigate further failures and prevent faults escalating.

Another potential improvement could be the incorporation of RFID within the factory to recognise parcels. This would improve the accuracy of both sorting boxes and locating/counting boxes at different locations throughout the factory. However, for this to be achieved it would require the parcels to be initially tagged so that they could then be sorted depending on their tag information and would have further implications in the complexity of the factory's design. Despite this, the factory still meets the requirements set out and all aspects run as required.

References

- [1] automation.com. 2021. A Beginner's PLC Overview, Part 1 of 4: Introduction to PLCs. [online] Available at:
<<https://www.automation.com/en-us/articles/2017/a-beginners-plc-overview-part-1-of-4-introduction>>
- [2] Rockwell Automation. 2021. CompactLogix 5480 Controllers | Allen-Bradley. [online] Available at:
<<https://www.rockwellautomation.com/en-gb/products/hardware/allen-bradley/programmable-controllers/small-controllers/compactlogix-family/compactlogix-5480-controllers.html>>
- [3] MITSUBISHI ELECTRIC Global Website. 2021. MITSUBISHI ELECTRIC Global website. [online] Available at:
<https://www.mitsubishielectric.com/fa/products/cnt/plc_fx/pmerit/contents/plc/index.html>
- [4] Siemens. 2021. SIMATIC S7-1200. [online] Available at:
<<https://new.siemens.com/global/en/products/automation/systems/industrial/plc/s7-1200.html>>
- [5] Se.com. 2021. TM241C24R - controller M241 24 IO relay | Schneider Electric UK. [online] Available at: <<https://www.se.com/uk/en/product/TM241C24R/controller-m241-24-io-relay/>> [Accessed 11 May 2021].
- [6] Ladder Logic World. 2021. Ladder Logic Basics - Ladder Logic World. [online] Available at:
<<https://ladderlogicworld.com/ladder-logic-basics/#:~:text=When%20implementing%20a%20ladder%20logic,essential%20and%20others%20are%20optional.>>