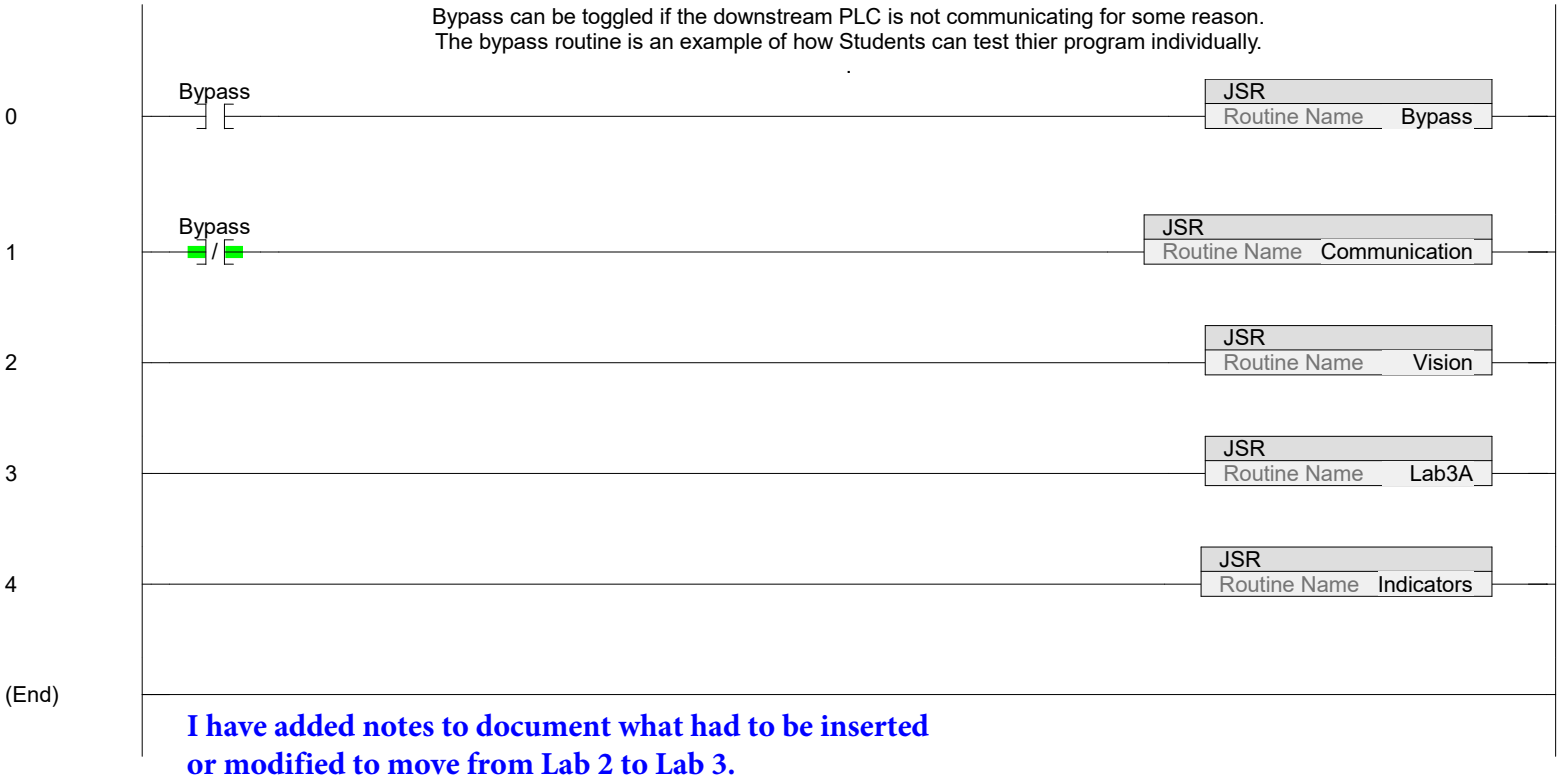


I have added notes to document what had to be inserted or modified to move from Lab 2 to Lab 3.

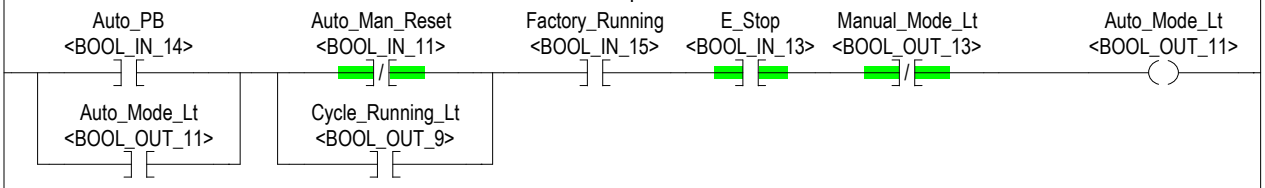
Added Routines:

- **Bypass (allows upstream to run without downstream)**
- **Communication (rungs to transfer data from PLC_1 to PLC_2)**
- **Vision (new vision sensors detects blue and green bases and lids)**

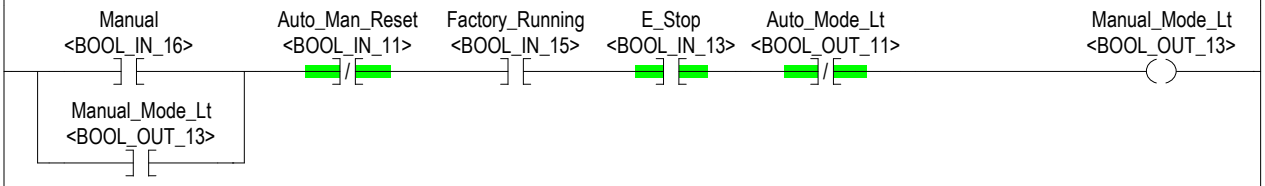
The main routine contains unconditional jumps to Vision, Lab3A and Indicators. If the downstream PLC is not connected or not ready, just toggle the Bypass Bit to transfer the data internally).

Auto, Manual and Cycle Rungs.
It should be noted that Cycle Start after Cycle Stop may only work smoothly if the Emitters are controlled, otherwise some jams and collisions may occur.
The Lab 2 instruction suggests keeping the same Automatic Program as the Lab 1 to keep the focus on Manual Mode Operation and the addition of Position Indication.
Lab 3 requires a few changes to separate Blue and Green components.

0



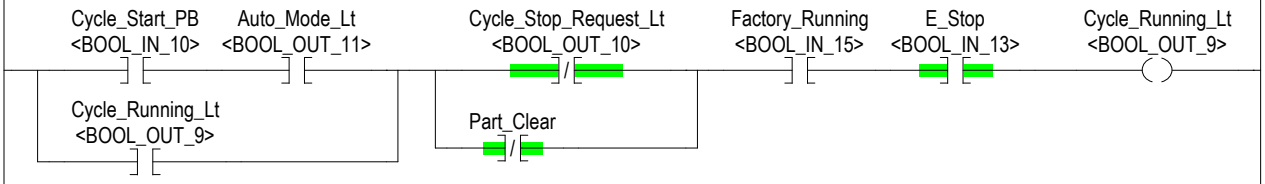
1



2

The intent of the Lab was to add Automatic & Manual Modes as well as Indicators & Lights without major changes to the original Automatic Code.
The logic after this point is the same as the original solution with two slight changes. One to allow an indicator to turn on (see Rung 8) and another for smoother operation (See Rung 19). Both are not necessary for Automatic Operation to function.

3



New rung for Lab 3.
Do not turn on Emitters unless there are no lids detected.
Emitter time must be adjusted for this to work correctly

4

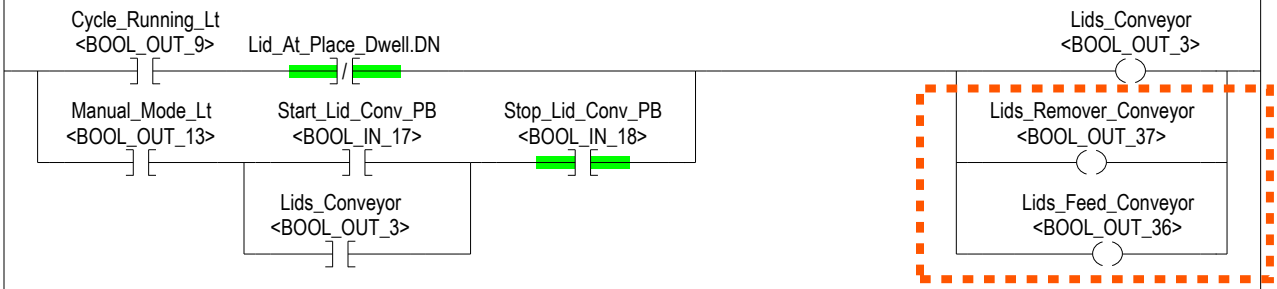
New rung for Lab 3.
Do not turn on Emitters unless there are no bases detected.
Emitter time must be adjusted for this to work correctly

5

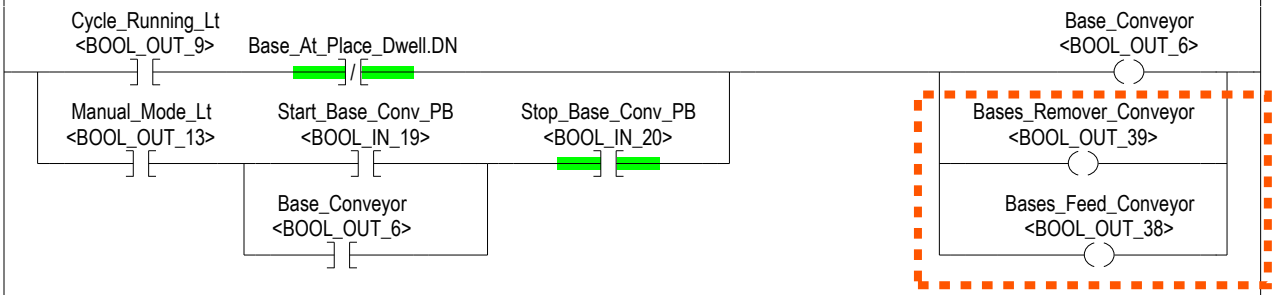
Additonal conveyors can just run in Parallel with the Main Conveyor.
These were added to unclutter the simulation.

..

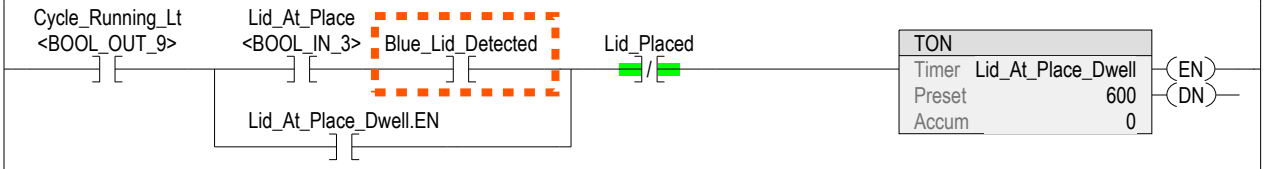
6



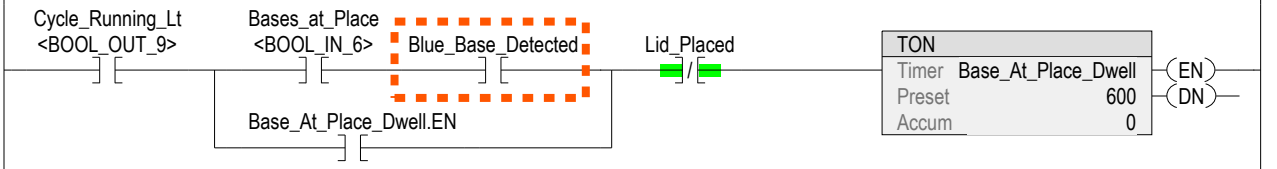
7



8



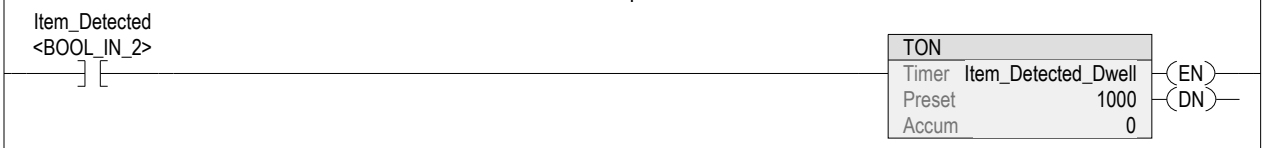
9



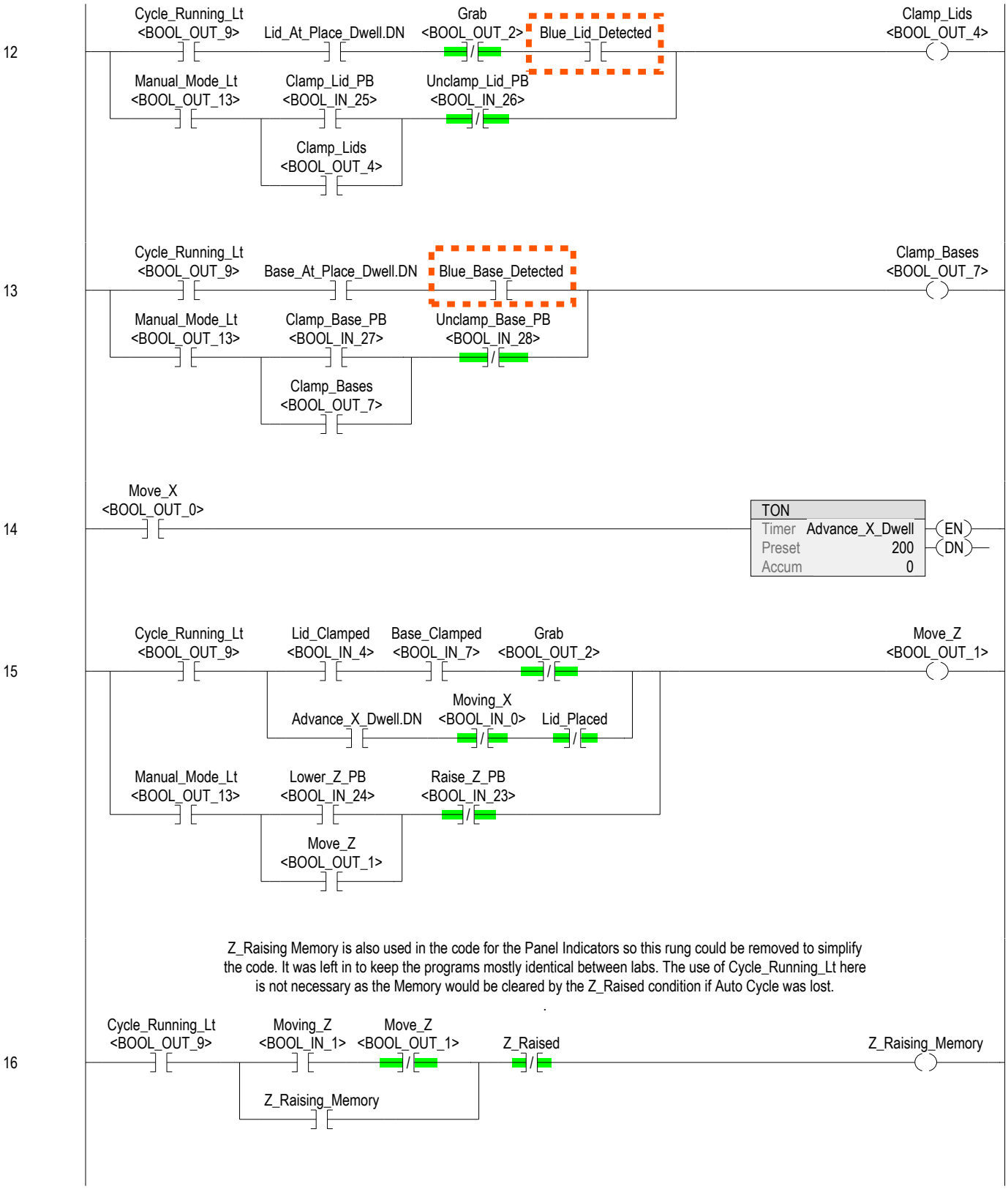
Added Item Detected Dwell as a way to ensure the Lowered Indicator has time to light up on the panel. The timing of the advance was too fast in the previous version.

.

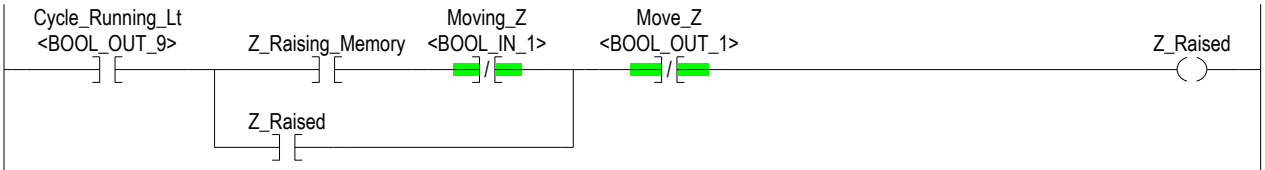
10



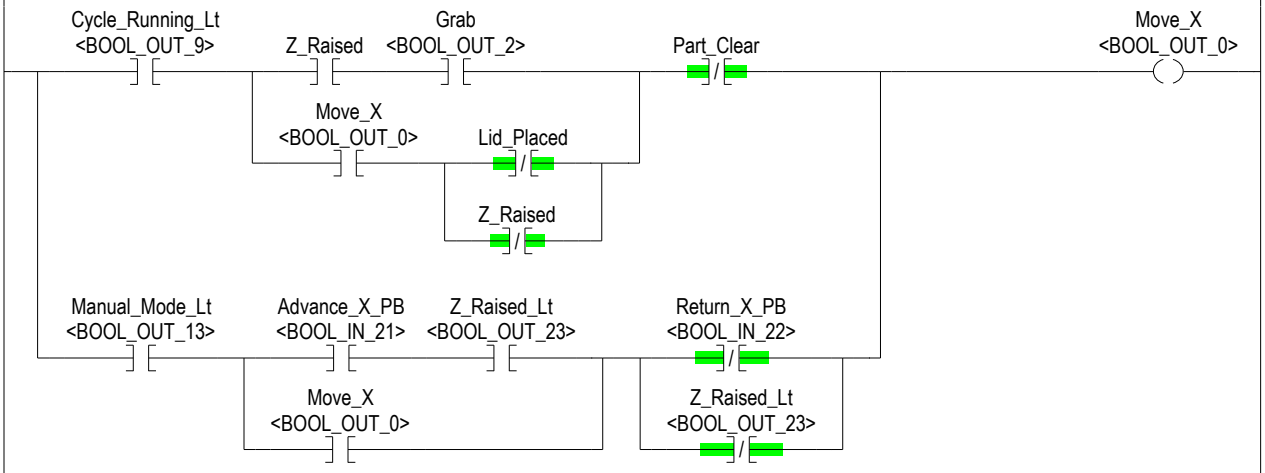
11


12
Cycle_Running_Lt <BOOL_OUT_9>
Lid_At_Place_Dwell.DN
Grab <BOOL_OUT_2>
Blue_Lid_Detected
Clamp_Lids <BOOL_OUT_4>
Manual_Mode_Lt <BOOL_OUT_13>
Clamp_Lid_PB <BOOL_IN_25>
Unclamp_Lid_PB <BOOL_IN_26>
Clamp_Lids <BOOL_OUT_4>
13
Cycle_Running_Lt <BOOL_OUT_9>
Base_At_Place_Dwell.DN
Blue_Base_Detected
Clamp_Bases <BOOL_OUT_7>
Manual_Mode_Lt <BOOL_OUT_13>
Clamp_Base_PB <BOOL_IN_27>
Unclamp_Base_PB <BOOL_IN_28>
Clamp_Bases <BOOL_OUT_7>
14
Move_X <BOOL_OUT_0>
TON
Timer Advance_X_Dwell
Preset 200
Accum 0
EN
DN
15
Cycle_Running_Lt <BOOL_OUT_9>
Lid_Clamped <BOOL_IN_4>
Base_Clamped <BOOL_IN_7>
Grab <BOOL_OUT_2>
Move_Z <BOOL_OUT_1>
Moving_X <BOOL_IN_0>
Advance_X_Dwell.DN
Lid_Placed
Manual_Mode_Lt <BOOL_OUT_13>
Lower_Z_PB <BOOL_IN_24>
Raise_Z_PB <BOOL_IN_23>
Move_Z <BOOL_OUT_1>
Z_Raising Memory is also used in the code for the Panel Indicators so this rung could be removed to simplify the code. It was left in to keep the programs mostly identical between labs. The use of Cycle_Running_Lt here is not necessary as the Memory would be cleared by the Z_Raised condition if Auto Cycle was lost.
16
Cycle_Running_Lt <BOOL_OUT_9>
Moving_Z <BOOL_IN_1>
Move_Z <BOOL_OUT_1>
Z_Raised
Z_Raising_Memory
Z_Raising_Memory

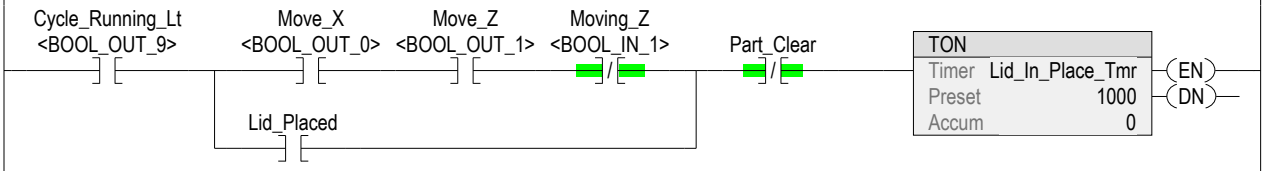
17



18



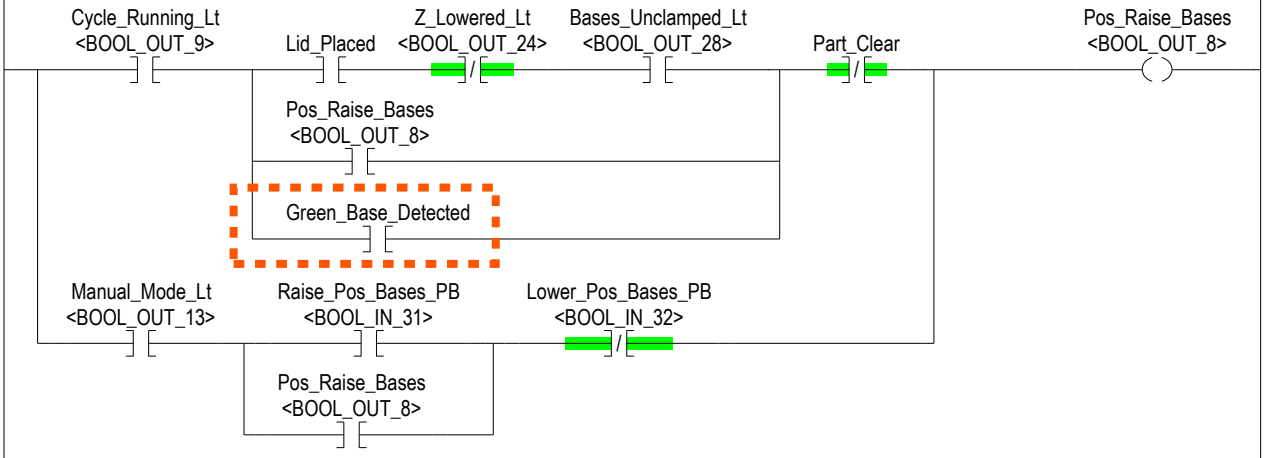
19

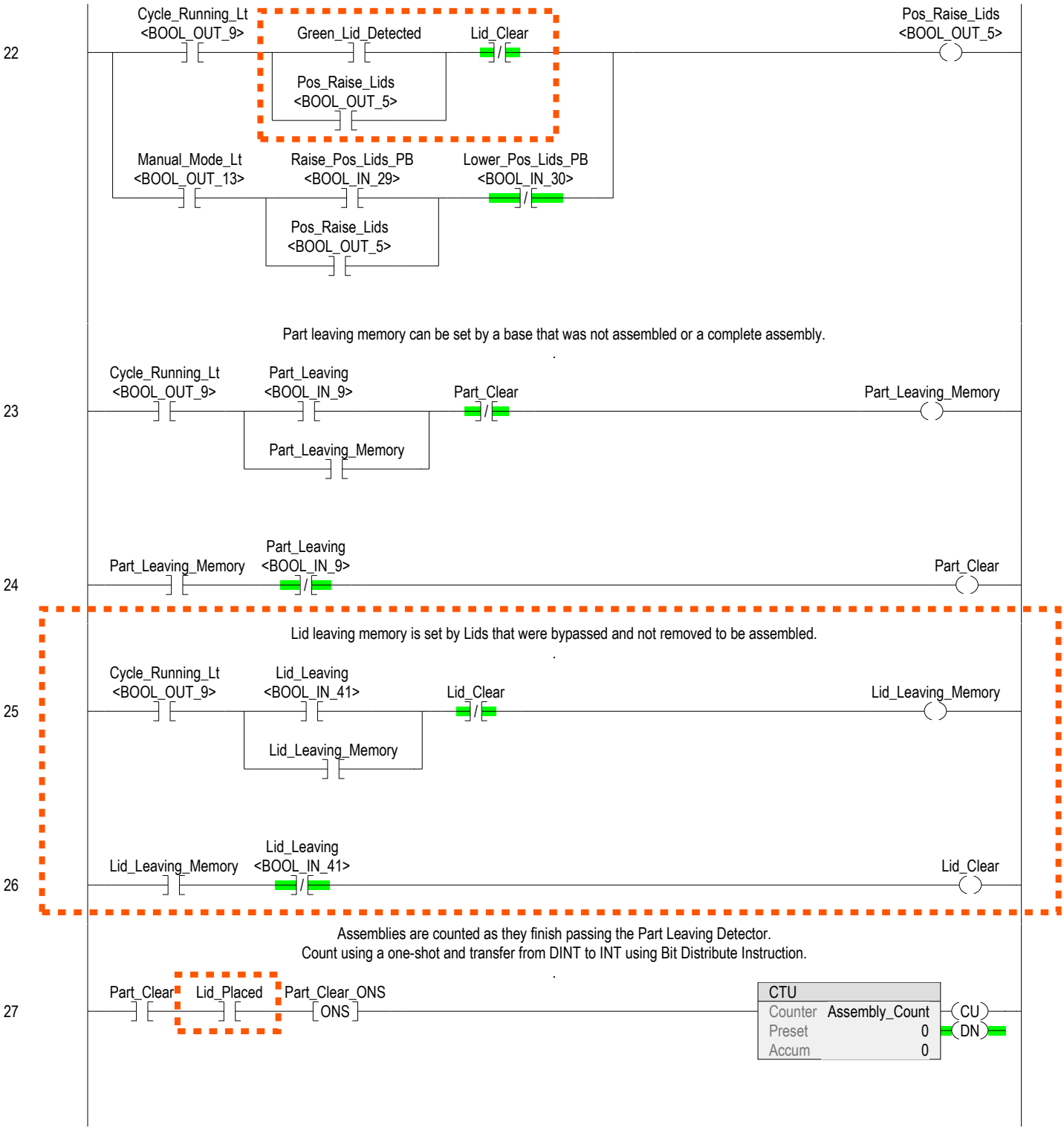


20

Added Unclamped Bases Lt. to Raise the Positioner as a way to slightly modify the program to run more realistically athough this change is not required.

21




22
Cycle_Running_Lt
<BOOL_OUT_9>
Green_Lid_Detected
Lid_Clear
Pos_Raise_Lids
<BOOL_OUT_5>
Pos_Raise_Lids
<BOOL_OUT_5>
Manual_Mode_Lt
<BOOL_OUT_13>
Raise_Pos_Lids_PB
<BOOL_IN_29>
Lower_Pos_Lids_PB
<BOOL_IN_30>
Pos_Raise_Lids
<BOOL_OUT_5>
Part leaving memory can be set by a base that was not assembled or a complete assembly.
23
Cycle_Running_Lt
<BOOL_OUT_9>
Part_Leaving
<BOOL_IN_9>
Part_Clear
Part_Leaving_Memory
Part_Leaving_Memory
24
Part_Leaving_Memory
Part_Leaving
<BOOL_IN_9>
Part_Clear
Lid leaving memory is set by Lids that were bypassed and not removed to be assembled.
25
Cycle_Running_Lt
<BOOL_OUT_9>
Lid_Leaving
<BOOL_IN_41>
Lid_Clear
Lid_Leaving_Memory
Lid_Leaving_Memory
26
Lid_Leaving_Memory
Lid_Leaving
<BOOL_IN_41>
Lid_Clear
Assemblies are counted as they finish passing the Part Leaving Detector.
Count using a one-shot and transfer from DINT to INT using Bit Distribute Instruction.
27
Part_Clear
Lid_Placed
Part_Clear_ONS
ONS
CTU
Counter Assembly_Count
Preset 0
Accum 0
CU
DN

28
BTD
Source Assembly_Count.ACC
0
Source Bit 0
Dest Part_Count
<INT_OUT_0>
0
Dest Bit 0
Length 16
Factory_Running
<BOOL_IN_15>
29
Assembly_Count
RES
(End)

Position Indication: Lab instructions were to add Lights and Pushbuttons with the intend of not altering the Automatic Code (for ease of implementation).

.

Conveyor Running Light can be run directly from the Lids and Bases Conveyor Output. This would be similar to a real machine since the Output would only seal in if the Physical Contactor energized (ie: overloads not tripped).

.

0

Lids_Conveyor <BOOL_OUT_3> — Lids_Conv_Running_Lt <BOOL_OUT_17>

1

Base_Conveyor <BOOL_OUT_6> — Bases_Conv_Running_Lt <BOOL_OUT_19>

X Axis Position Indicators.
The techniques used to detect the Position of the Axis is somewhat unique to the simulation. X_Advancing_Memory or X_Returning_Memory only turns ON when the simulation returned the "Moving" bit back to the PLC. Moving is used for both directions, however, since Move_X determines the direction of motion, the combination of the two bits can be utilized. Once the X Axis is no longer Moving, the Directional Moving Memory is used to turn ON the correct panel light.

.

.

2

Move_X <BOOL_OUT_0> · Moving_X <BOOL_IN_0> · X_Adanced_Lt <BOOL_OUT_21> (/) · Factory_Running <BOOL_IN_15> → X_Advancing_Memory

X_Advancing_Memory (seal-in)

3

X_Advancing_Memory · Moving_X <BOOL_IN_0> (/) · Move_X <BOOL_OUT_0> · Factory_Running <BOOL_IN_15> → X_Advanced_Lt <BOOL_OUT_21>

X_Adanced_Lt <BOOL_OUT_21> (seal-in)

4

Move_X <BOOL_OUT_0> (/) · Moving_X <BOOL_IN_0> · X_Returned_Lt <BOOL_OUT_22> (/) · Factory_Running <BOOL_IN_15> → X_Returning_Memory

X_Returning_Memory (seal-in)

In order to detect the axis position on "power up", a detection Limit was added. Returning Memory is redundant but left in to demonstrate both methods. It should be noted that, if the Switch was removed, the X-Axis would have to be Advanced and then Returned (in Auto or Manual) for the Returned Light to turn ON.

.

5

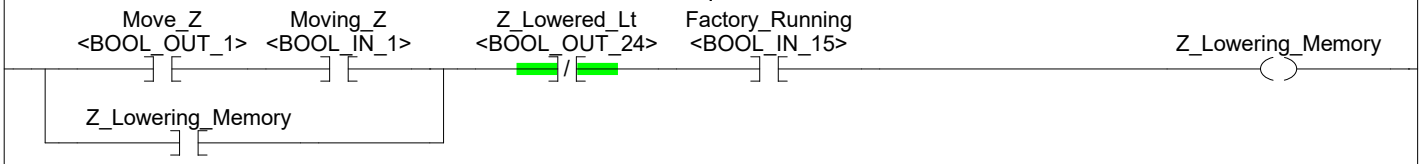
X_Returning_Memory · Moving_X <BOOL_IN_0> (/) · Move_X <BOOL_OUT_0> (/) · Factory_Running <BOOL_IN_15> → X_Returned_Lt <BOOL_OUT_22>

X_Returned_Prox <BOOL_IN_33>

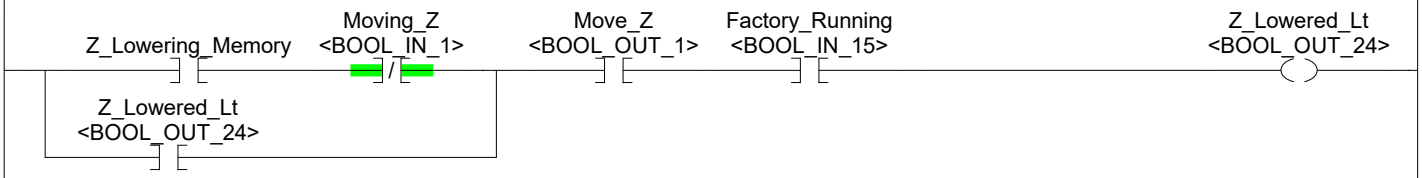
X_Returned_Lt <BOOL_OUT_22> (seal-in)

Z Axis Position Indicators.
Same as X Axis.

6

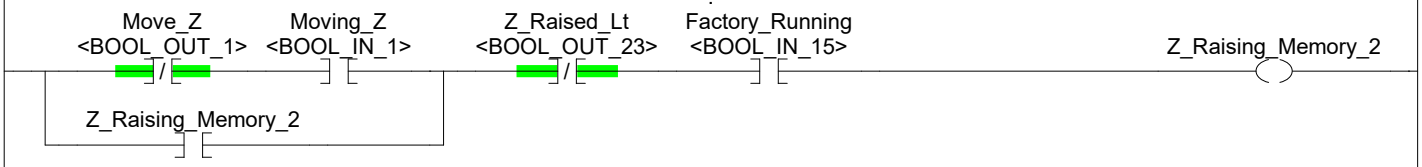


7

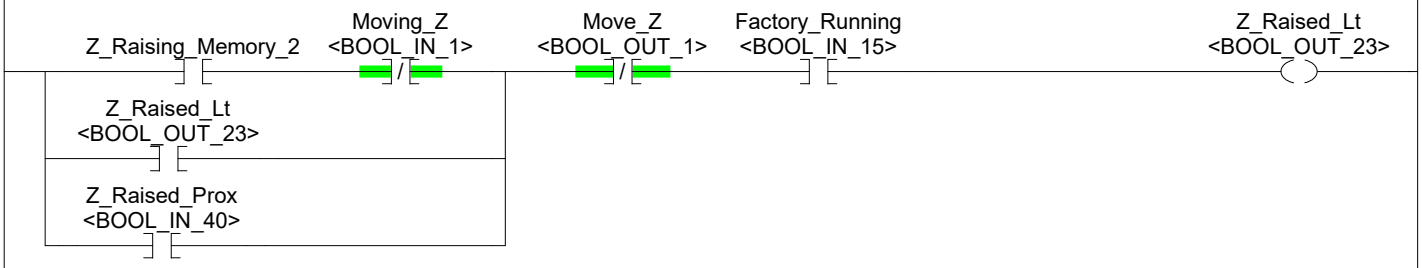


Using Z_Raising_Memory_2 so as to not conflict with Automatic Code.

8

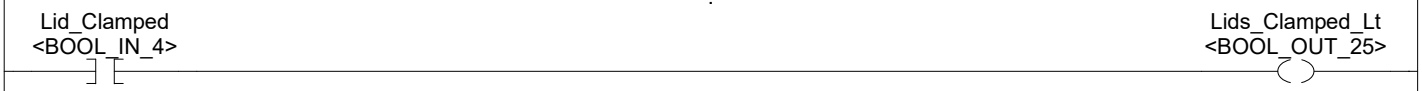


9



Lids Clamp Position Indicators.
The Clamped operation has an input from the Simulation that represents "Clamped" status. This is similar using a Pressure Switch on the Pneumatic Feed Line to detect when the part has been Captured allowing for parts of various sizes to be used.

10



Since the Simulation Scene does not come standard with an Unclamped Proximiaty Switch and to avoid adding a switch both the Lowered and Raised Position of the Positioner, a timer is used as an "Assumption" of motion completion.

11

12

Lids_Unclamped_Prox <BOOL_IN_35> | Clamp_Lids <BOOL_OUT_4> (/) | Factory_Running <BOOL_IN_15> | Lids_Unclamped_Lt <BOOL_OUT_26>

Lids_Unclamped_Dwell.DN

Lids_Unclamped_Lt <BOOL_OUT_26>

Bases Clamp Position Indicators.
Same explanation as the Lids Clamp.

13

Base_Clamped <BOOL_IN_7> | Bases_Clamped_Lt <BOOL_OUT_27>

14

Clamp_Bases <BOOL_OUT_7> (/) | Base_Clamped <BOOL_IN_7> (/)

TON
Timer Bases_Unclamped_Dwell
Preset 800
Accum 800
EN
DN

15

Bases_Unclamped_Prox <BOOL_IN_38> | Clamp_Bases <BOOL_OUT_7> (/) | Factory_Running <BOOL_IN_15> | Bases_Unclamped_Lt <BOOL_OUT_28>

Bases_Unclamped_Dwell.DN

Bases_Unclamped_Lt <BOOL_OUT_28>

Lids Pos Position Indicators.
The Positioners output an "In Position" status which can be used in the same way the Z and X Axis Moving are. However, in this example, the Lowered Status is read directly from a user added Proximity Switch (recall the Z and X axis Returning Memory Programming, although redundant, was left in the program).

16

Lids_Pos_Lowered_Prox <BOOL_IN_34> | Lids_Pos_Lowered_Lt <BOOL_OUT_30>

17

Pos_Raise_Lids <BOOL_OUT_5> | Lids_Pos_at_Limit <BOOL_IN_36> (/) | Lids_Pos_Raised_Lt <BOOL_OUT_29> (/) | Factory_Running <BOOL_IN_15> | Lids_Pos_Raising_Memory

Lids_Pos_Raising_Memory

18

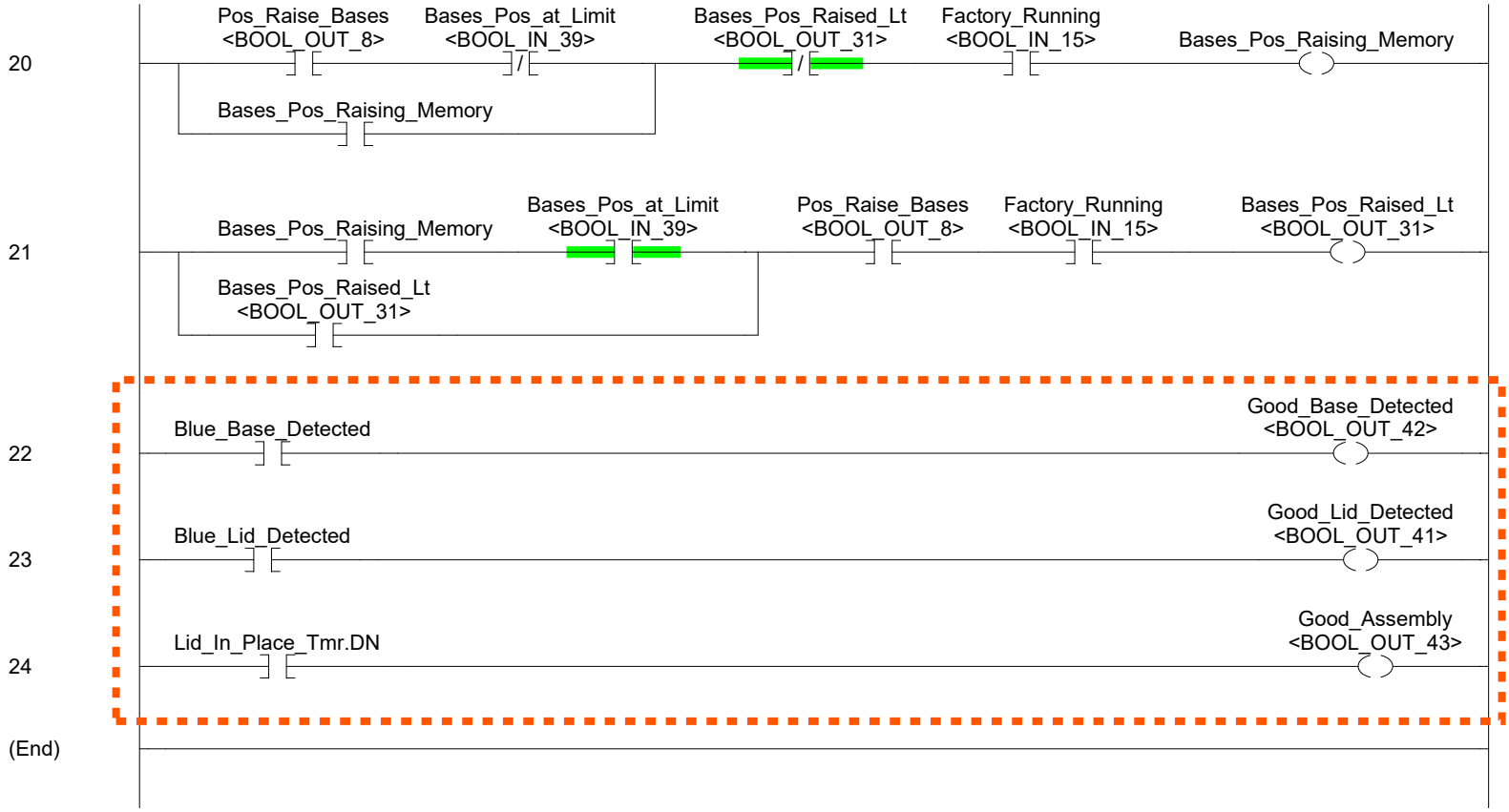
Lids_Pos_Raising_Memory | Lids_Pos_at_Limit <BOOL_IN_36> | Pos_Raise_Lids <BOOL_OUT_5> | Factory_Running <BOOL_IN_15> | Lids_Pos_Raised_Lt <BOOL_OUT_29>

Lids_Pos_Raised_Lt <BOOL_OUT_29>

Bases Pos Position Indicators

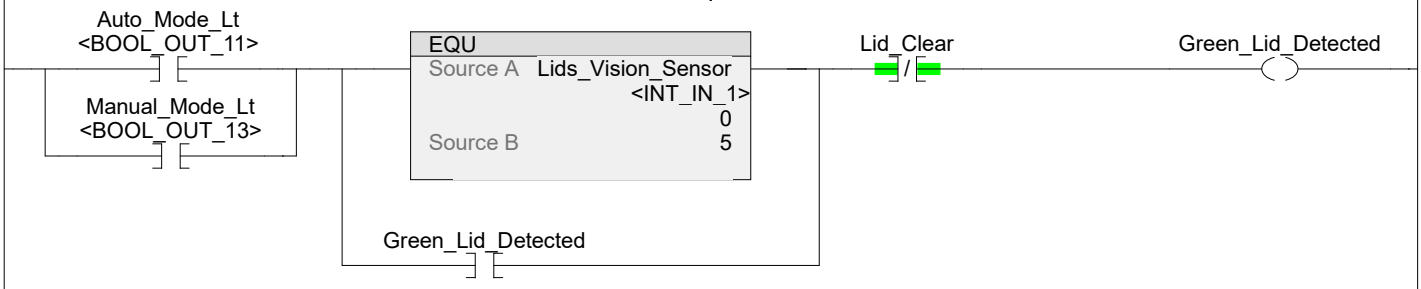
19

Bases_Pos_Lowered_Prox <BOOL_IN_37> | Bases_Pos_Lowered_Lt <BOOL_OUT_32>


20
Pos_Raise_Bases
<BOOL_OUT_8>
Bases_Pos_at_Limit
<BOOL_IN_39>
Bases_Pos_Raised_Lt
<BOOL_OUT_31>
Factory_Running
<BOOL_IN_15>
Bases_Pos_Raising_Memory
Bases_Pos_Raising_Memory
21
Bases_Pos_Raising_Memory
Bases_Pos_at_Limit
<BOOL_IN_39>
Pos_Raise_Bases
<BOOL_OUT_8>
Factory_Running
<BOOL_IN_15>
Bases_Pos_Raised_Lt
<BOOL_OUT_31>
Bases_Pos_Raised_Lt
<BOOL_OUT_31>
22
Blue_Base_Detected
Good_Base_Detected
<BOOL_OUT_42>
23
Blue_Lid_Detected
Good_Lid_Detected
<BOOL_OUT_41>
24
Lid_In_Place_Tmr.DN
Good_Assembly
<BOOL_OUT_43>
(End)

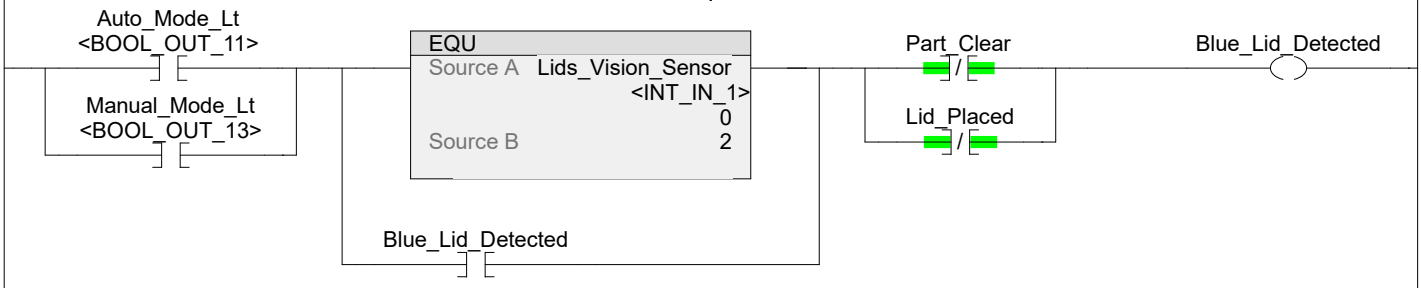
Green Lid detection when vision = 5.
Seal in status until the Green Lid by-pases the clamp.

0



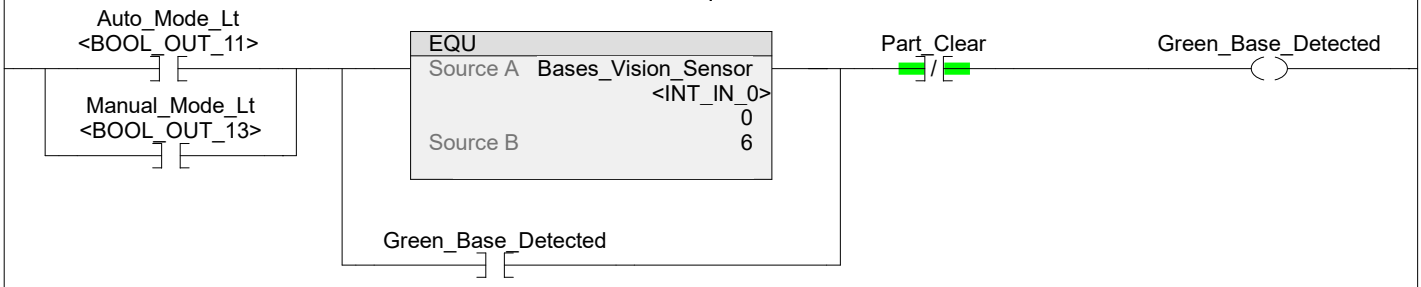
Blue Lid detection when vision = 2.
Seal in status until the Part is Clear but only if the Part is an assembly. Green bases also trigger the Part Clear Bit.
Lid Clear could be added if a Blue Lid somehow did not stop and continued down the conveyor.

1



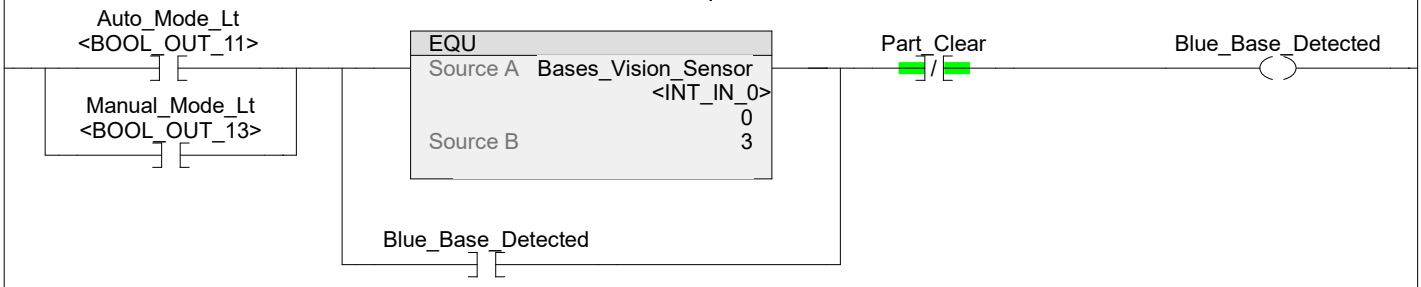
Green Base detection when vision = 6.
Seal in status until the any part passes the exit proximity switch.

2



Blue Base detection when vision = 3.
Seal in status until the any part passes the exit proximity switch.

3



(End)

Send the value of the part detected in the Remover to the Downstream PLC.

0

MOV
Source Lids_Remover_Detected_Parts <INT_IN_4> 0
Dest Lids_53_to_52 0

Data from the Downstream PLC confirms receipt of data.
Send data to Remover.

1

MOV
Source Lids_52_to_53 0
Dest Lids_Remover_Parts_to_Remove <INT_OUT_4> 0

Send the value of the part detected in the Remover to the Downstream PLC.

2

NEQ
Source A Bases_Remover_Detected_Parts <INT_IN_3> 0
Source B 1152

MOV
Source Bases_Remover_Detected_Parts <INT_IN_3> 0
Dest Bases_53_to_52 0

Data from the Downstream PLC confirms receipt of data.
Send data to Remover.

3

MOV
Source Bases_52_to_53 0
Dest Bases_Remover_Parts_to_Remove <INT_OUT_3> 0

4

EQU
Source A Bases_Remover_Detected_Parts <INT_IN_3> 0
Source B 1152

MOV
Source Bases_Remover_Detected_Parts <INT_IN_3> 0
Dest Bases_Remover_Parts_to_Remove <INT_OUT_3> 0

(End)

This routine sends the Detected Base or Part Value directly back to the Remover.
Note: Lids and Bases are both PARTS.
The "bases" the remover refers to are Pallets in the Simulation.
This is useful when testing without communication Functioning.

0

MOV
Source Bases_Remover_Detected_Parts
<INT_IN_3>
0
Dest Bases_Remover_Parts_to_Remove
<INT_OUT_3>
0

1

MOV
Source Lids_Remover_Detected_Parts
<INT_IN_4>
0
Dest Lids_Remover_Parts_to_Remove
<INT_OUT_4>
0

(End)