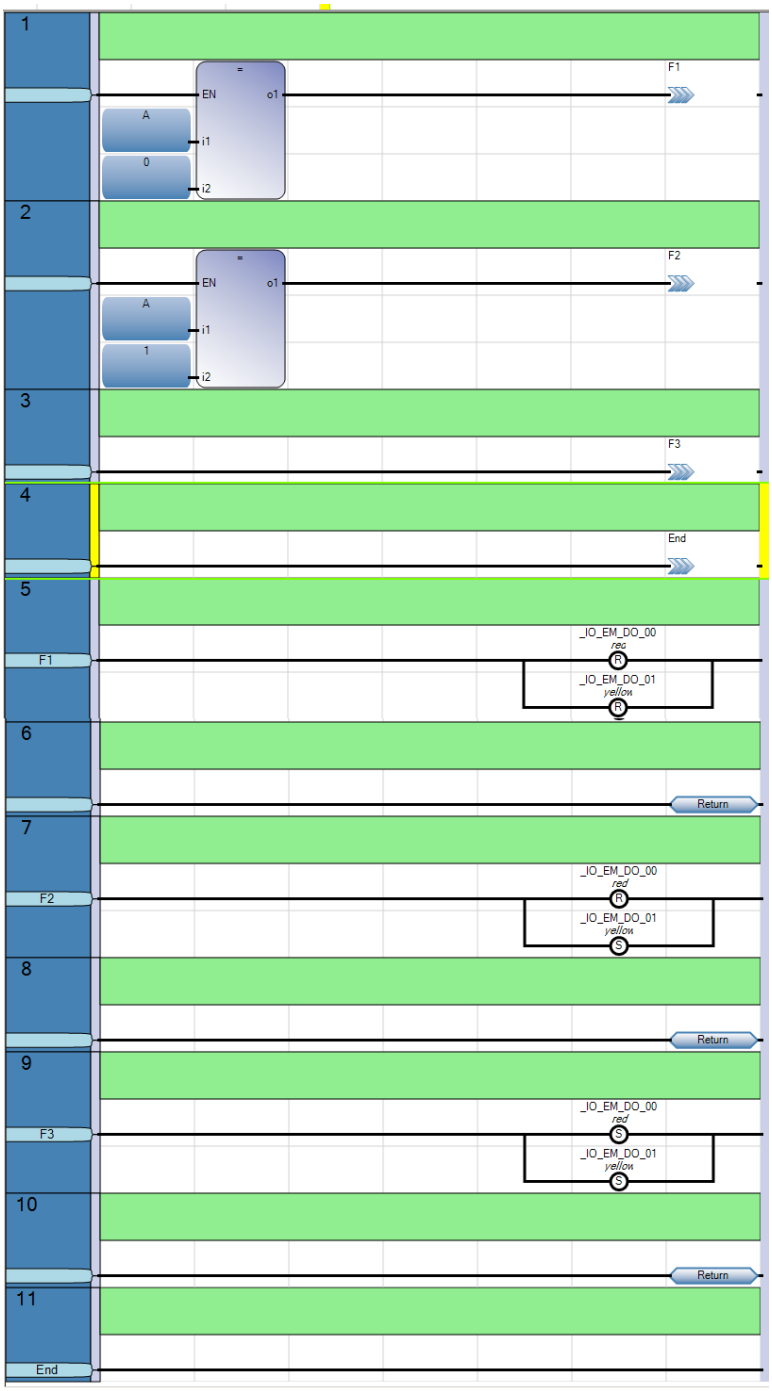
Loops and Counters

If Statement:

```
if (A == 0) then do function F1
if (A == 1) then do function F2
do function F3
```

Ladder Logic Code:

A return statement takes you to the next line of code. This checks if A=0. If so, it calls function F1.



do {} while Loops:

```
do {
   function F1
   } while (A < 10)
```

You really can't implement do}{ and while{} loops with ladder logic. The program must finish every 10ms and then restart from line 1. You can't get stuck in the program waiting for something to happen.

What you can do is switch between states with if( ) statements though.

for..next loops

Here, use a counter. For example, write a program for a conveyor belt:

Input:

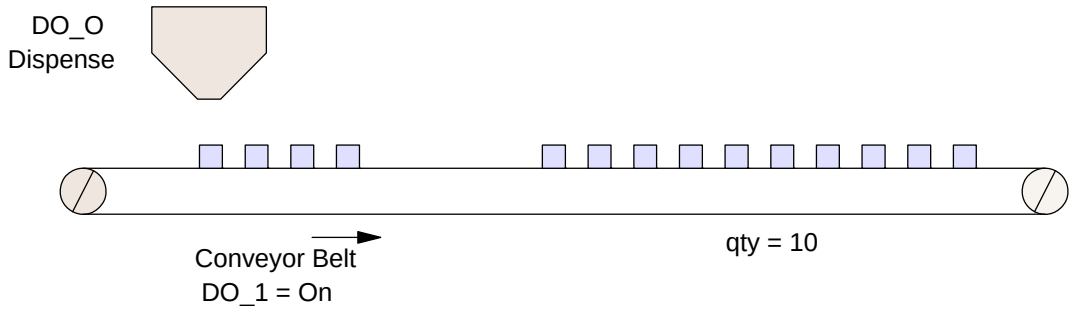
- IN_0: Start
- IN_1: Stop (Kill the process)

Output:

- OUT_0: Dispense a pill
- OUT_1: Turn on the conveyor belt

Wait until you press the start button. Once pressed,

- Turn on the conveyor belt.
- While running, dispense ten pills by pulsing the red light on for 500ms and off 2 seconds.
- When you get to 10 pills, turn off the conveyor belt for 2 seconds.
- Turn back on the conveyor belt and dispense 10 more pills
- repeat



Ladder Diagram:

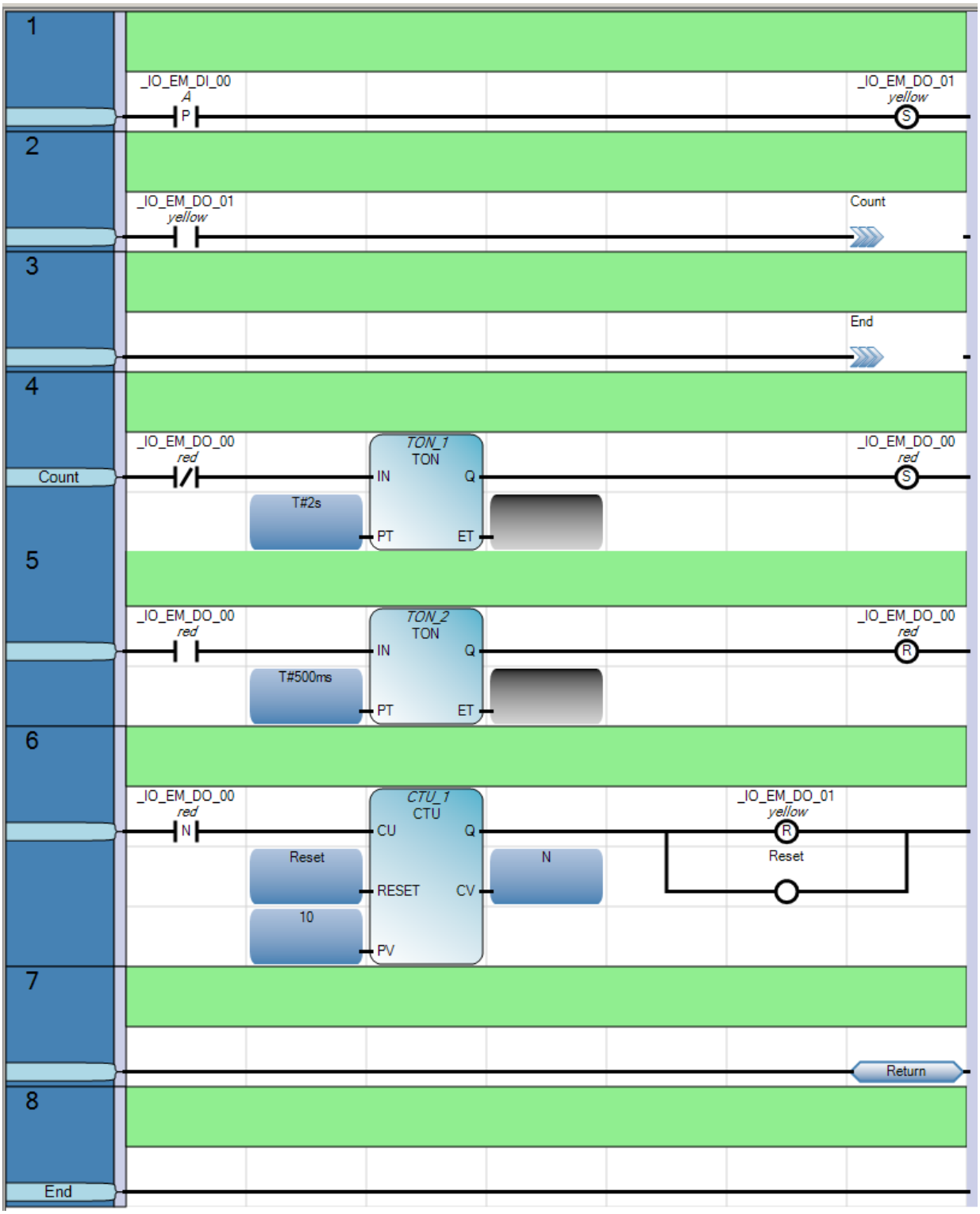
Line 1: When the Start button is pressed, the conveyor belt is turned on.

Line 2: While running, 10 pills are dispensed

Line 3: end of program

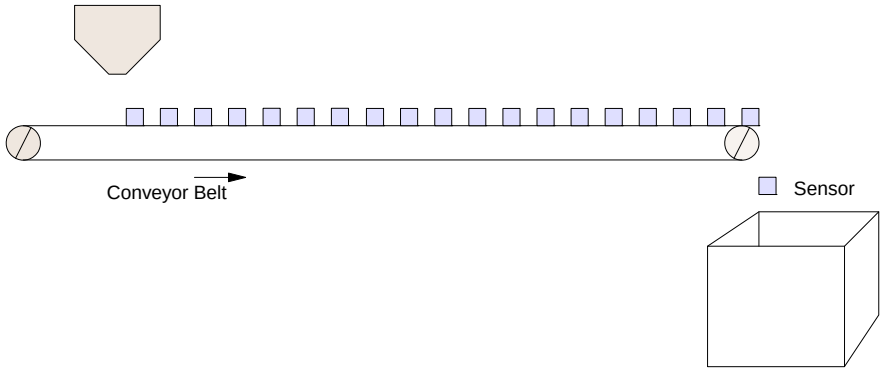
Line 4/5: T-flip flop (variation) which turns off the light for 2 seconds, on for 500ms

Line 6: Count each pill as it is output on the falling edge. Once you get to 10, clear the counter and restart the conveyor belt.



State Transition & Flow Charts

Write a program which puts 10 items in a box from a conveyor belt



Inputs:

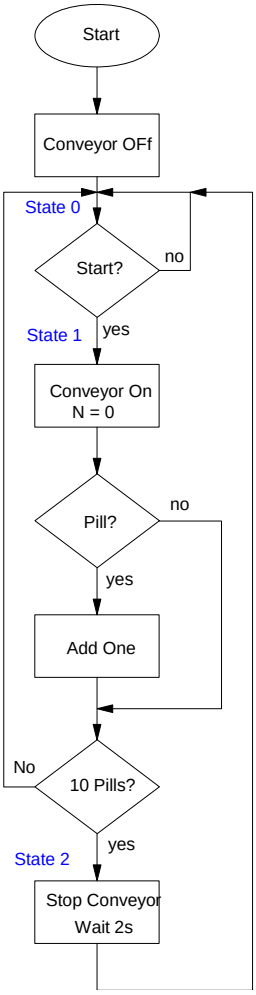
- 0: Master On
- 1: Sensor. Pulse means a pill fell into the box
- 2: Master Kill. Stop the process

Outputs:

- 0: Conveyor Belt On

Flow Chart: One way to do a flow chart is shown to the right. The problem with this flow chart is you have a couple of loops that take longer than 10ms (the cycle time for the PLC). The PLC needs to finish the programs every 10ms in order to operate, however.

To fix this, change the program slightly (next page)

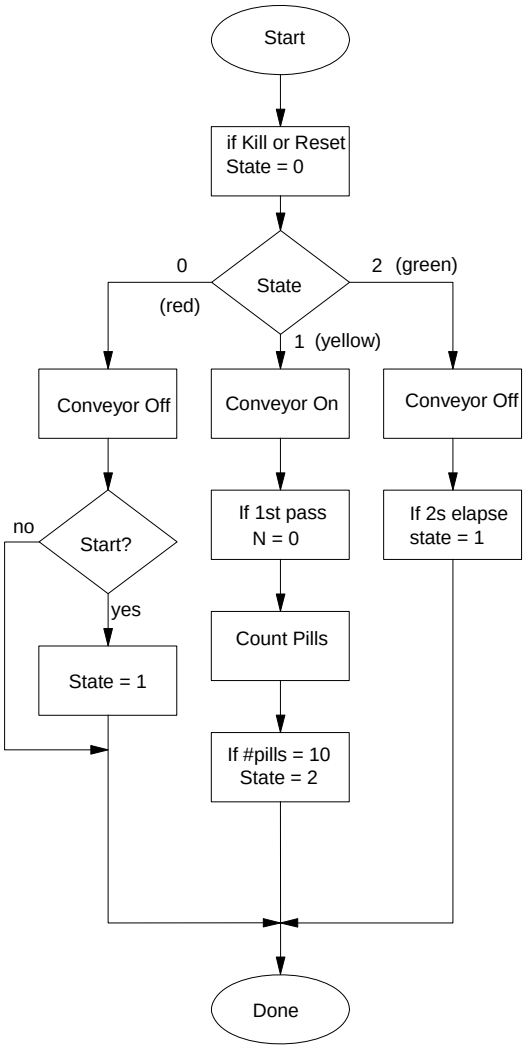


Kind of along the same idea, have three paths (states):

0: On power up, you are in State 0. Here, you are waiting for the Start button to be pressed. Once pressed, you switch to state 1 (run mode).

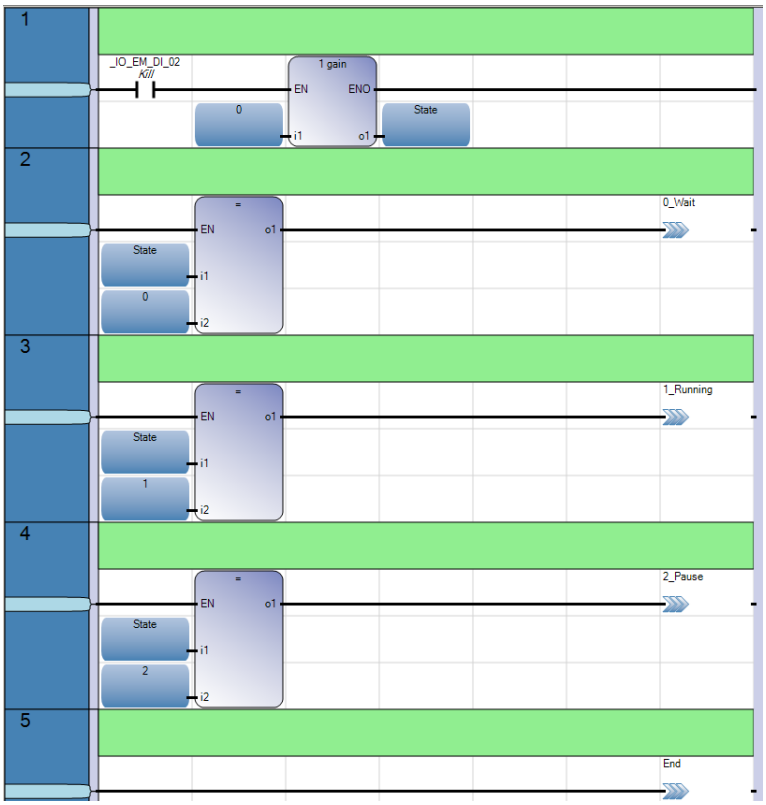
1: When running, you turn on the conveyor belt and start counting how many pills pass by the sensor. Once you get to 10 pills, change the state to 2

2: Once you get to 10 pills, turn off the conveyor belt, wait 2 seconds, then go back to state 1 (restart the conveyor belt and reset the counter to 0).



Ladder Diagram:

Main Loop: Lines 1-5



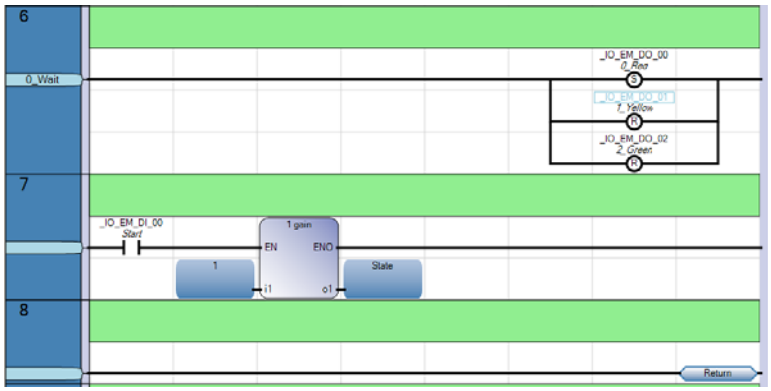
Lines 1: Master Kill. If button 2 is pressed, you always go back to state 0 (conveyor off).

By placing the kill switch in the main loop, it is always being checked.

Line 2 / 3 / 4: Branch to different sections of the code. Each is terminated with a return statement, so it continues with the next line of code.

Line 5: At the end of the main routine, you exit (and restart 10ms later)

State 0: Wait



In this state, you are waiting for the start button

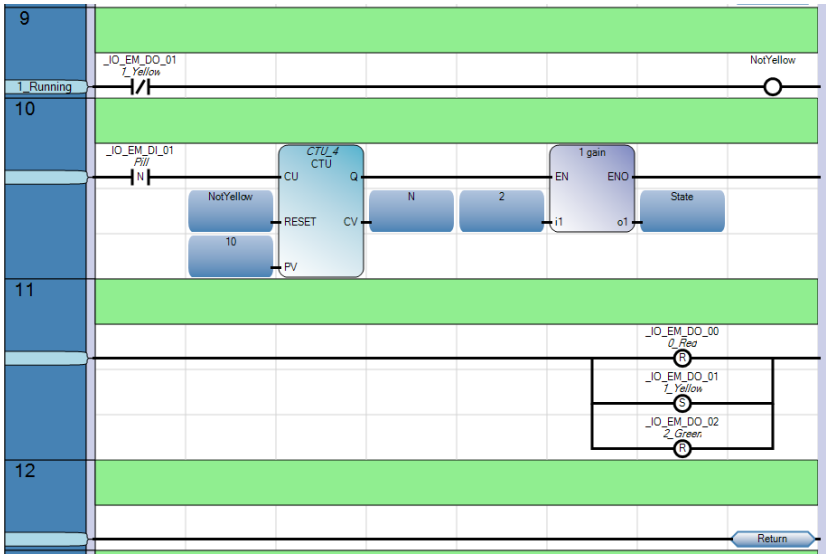
Line 6: Turn on the red LED so you know you are in state 0.

Line 7: If you hit the start button, go to State 1.

Line 8: End of subroutine

State 1: Running

Turn on the conveyor belt and count pills. Keep counting until you get to 10 pills.



Line 11: Turn on the yellow LED to indicate that you are in State 1. Note that this is at the end of the routine so you can detect the 1st time you are going through the routing.

Line 9: The first pass through this loop, you need to clear the counter. This is indicated by the yellow light being off.

Line 10: Each time a pill falls into the box, in increment a counter. When the count gets to 10, change the state to 2.

Note that the Reset on the counter has to be true the first pass to clear the counter. This is the reason the LEDs are turned on / off at the end of the routine.

State 2: Pause

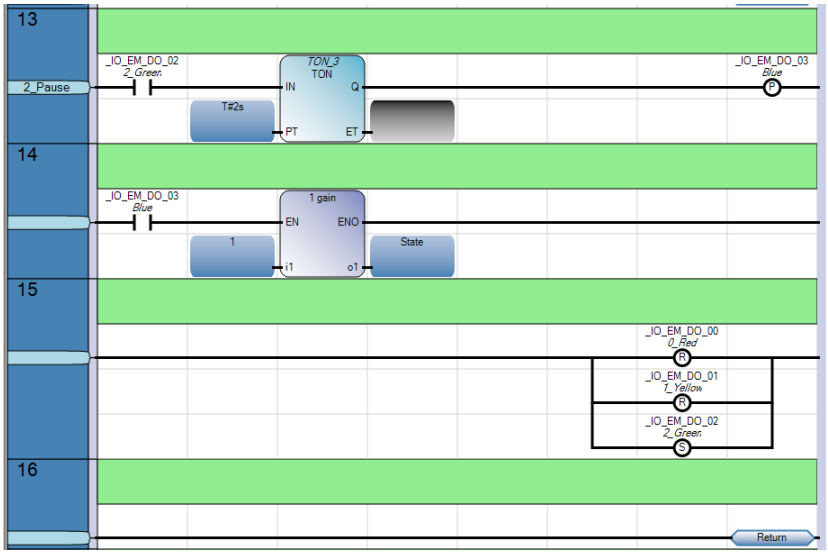
Turn off the conveyor belt and wait 2 seconds.

Line 15: Turn on the green LED to indicate you are in State 2: Pause.

Note that this is at the end of the routine so that the 1st pass will see the green LED as being off.

Line 13: On the first pass, clear the timer. From then on, count up to 2 seconds.

Line 14: Once you get to 2 seconds, make the next state is 1: Running



PLC Problems:

1) PLC Controlled Car:

Input:

- DI_00 Sensor to detect an object in front of the car

Output:

- DO_00 - DO_03 H-Bridge driving a motor
- 0000 Off
- 1010 Forward
- 0101 Reverse

Function: Go forward until you hit something (DI_00 = 1). When that happens

- Stop for 1 second
- Go in reverse for 1 second (which turns the car through a mechanical design)
- Stop for one second
- Go forward again

2) Drive a conveyor belt which is connected to a motor.

Input:

- DI_00 Optical Sensor (1 indicates object present)

Output:

- DO_00 Motor. Closed turns on the conveyor belt.

A part on the conveyor is to be positioned underneath an optical sensor. The operation is

- Drive the conveyor
- When the sensor detects an object,
  - Keep the conveyor on for 1.5 seconds
  - Then stop the conveyor for 2.0 seconds
  - Then restart the conveyor.

3) Accept / Reject Sorting. Repeat problem #2.

Input:

- DI_00 Optical Sensor (1 indicates object present)
- DI_01 Good / bad sensor (1 indicates part is bad - problem 3)

Output:

- DO_00 Motor. Closed turns on the conveyor belt.
- DO_01 Pneumatic Actuator: Closed pushes the object off the conveyor belt into the trash bin.

A part on the conveyor is to be positioned underneath an optical sensor. The operation is

- Drive the conveyor
- When the sensor detects an object,
  - Keep the conveyor on for 1.5 seconds
  - Then stop the conveyor for 2.0 seconds
  - 0.5 second after the conveyor stops, the good/bad sensor is checked. If the part is bad, the pneumatic actuator turns on for 1.0 second and pushes the object off the conveyor belt into the trash can.
  - Then restart the conveyor.

4) Power Line Fault Remover. To remove a fault on a power line (short to ground), the power lines turned off and then turned on three times to try to clear the fault (burn off the tree branch). If the fault isn't cleared after three tries, the transformer turns off and a fault is indicated.

Input:

- DI_00 Fault detected (1)

Output:

- DO_00 0 = line open, 1 = line powered
- DO_01 0 = OK, 1 = fault

(Loop and counter)

5) Counter: Count out 10 pills and fill a box

Input:

- DI_00 Sensor. 1 = pill, 0 = no pill

Output:

- DO_00: 1 = First conveyor belt (pills)
- DO_01: 1 = Second conveyor belt (boxes)

Turn on the conveyor belt, counting the number of boxes that pass by.

When 10 pills are counted

- Stop the conveyor belt for 5.0 seconds
- Turn on the second conveyor belt for 2.0 seconds to bring forth the next box
- Clear the count and restart the conveyor belt