MFET 670 - AUTOMATED CAR WASH

To: Professor David Cicero

From:
ZEYAR WIN

OUTLINE

Problem Statement:

Automation industry makes the resources we use daily basis cheaper and more reliable than ever before. In this class, we learn concepts in the control of factory automation systems, skills in PLC programming, and hardware and software for user interfaces and networking control system. The final project requires the overall PLC knowledge for implementation in real life scenario. In this project, Car wash automation system is created using two PLCs with produced and consumed interfaces. Besides doing the project with PLC ladder logic program, the additional program is used for its convenience. When the car arrives, the system automates opening door, moving car, closing door, cleaning the car, and showing the number of cars waiting and being washed on different displays.

Approach

The approach starts how to know the car arrive in the lab without real car washing systems. Theoretical simulation works with using momentary button and/or switch as the car coming into automated car wash and starting the system. When the car arrives, Cylinder opens the door. Stepper Motor moves 1 revolution to move the car into zone 1. The cylinder is used again to close the door. The light is flashed 5 times to signal the presoak to run. Fan acts a run foam applicator. After some time, DC motor is oscillating clockwise and counterclockwise to run scrubber. The car is then moved into zone 2 doing 1 revolution of stepper motors in both PLCs. The other PLC starts running from this point forward. Oscillating

DC motor makes run rinse application. Fan acts as running dryer. Cylinder opens exit door when stepper motor moves the car out of zone 2. Cylinder is again used to close the exit door. The display shows the number of cars waiting in zone 1 PLC display, and other one does show those finished in zone 2 PLC display.

In this lab, first approach was doing simple thing zone 1. Testing the car arrives, do the simple thing as required. After those simple parts are done, the flashing times is attempted using the Structure Text. Since it is not working well as expected, FBD is utilized instead for full functionality. The logic implementation is tested step by step. The output simulation can be observed clearly with cylinder, light, fan and DC motor running. Rockwell manual for PLC has useful resources for this lab project.

Car wash function	**Simulation**
Car arrives	Momentary switch button
System starts	Selector Switch
Open door	Cylinder
Close door	Cylinder
Move car	1 revolution of stepper motor
Run presoak	Flashing light 5 times
Run foam	Fan
Run wax applicator in zone 2	Flashing light 5 times
Run dryer	Fan
Display number of cars waiting and finish being washed	PLC displays

Add on Instruction

Initially structure text is tried to run the light flashing 5 times. It flashes the light well however the counter logic in structure text is not working. FBD (Function Block Diagram) is used to simulate the light flashing. FBD is easier to implement with counter, timer on, and pulses as required. The small issue is when to reset, the FBDs need to be placed on the program with RESET value being 1.

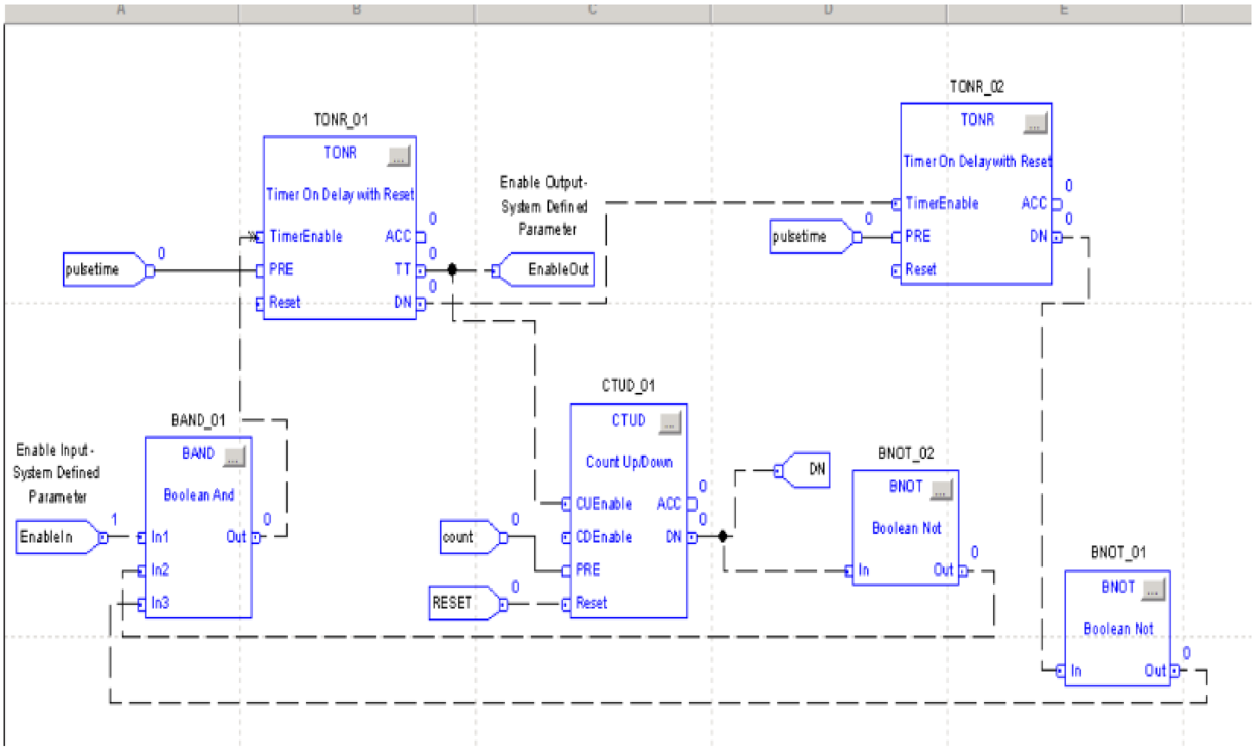
Initial add-on

```
if (EnableIn) then
     Timeron.PRE:=On_pulse;
     TONR (Timeron);
     Timeron.TimerEnable:=1;
end_if;

if (EnableIn) & (Timeron.ACC =On_pulse) then
     Timeroff.PRE:=Off_pulse;
     TONR(Timeroff);
     Timeroff.TimerEnable:=1;
end_if;

if(Timeroff.ACC =Off_pulse) or not(EnableIn) then
     Timeron.TimerEnable:= 0;
     Timeroff.TimerEnable:=0;
     Timeroff.ACC:=0;
end_if;

if(Timeron.TT) then
     EnableOut:=1;
     else if not(Timeron.TT) then
     EnableOut:=0;
     end_if;
end_if;


if EnableIn & count < 100 then
     for count := 1 to 100 by 1 do
     value := count;
     end_for;
end_if;
```

- TONR Structure; Timer stopped working

- CTUD Structure

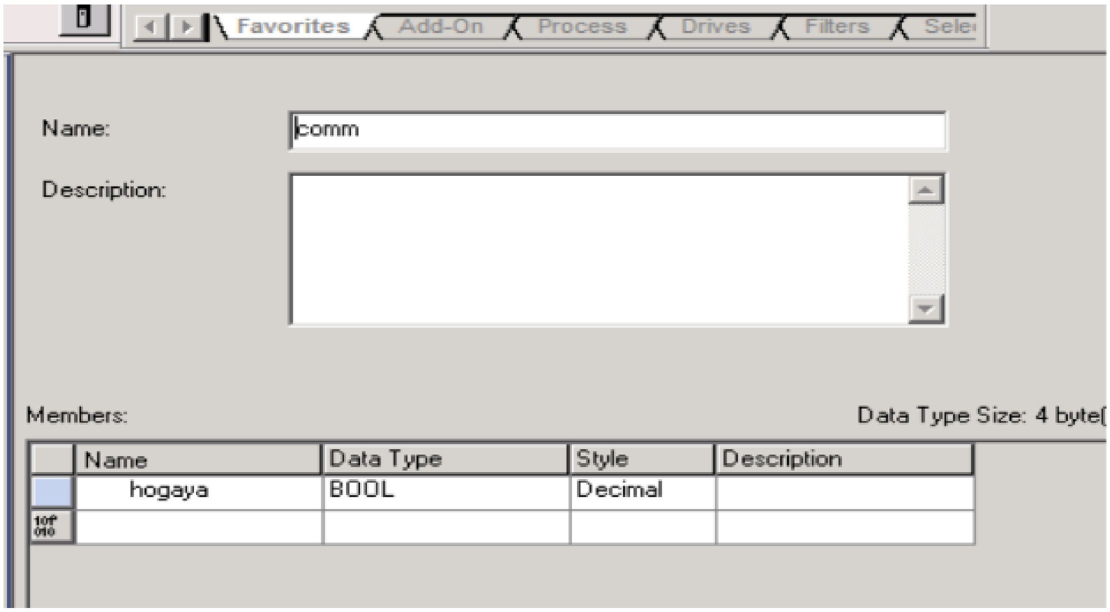
The add-on instruction in use



UDT

The UDT is used in this lab to facilitate the program process. Only one UDT is created. That UDT is interface for communicating the PLC 1 and PLC 2. When PLC 1 is done, the Boolean data bit will be TRUE and PLC 2 can start afterwards.

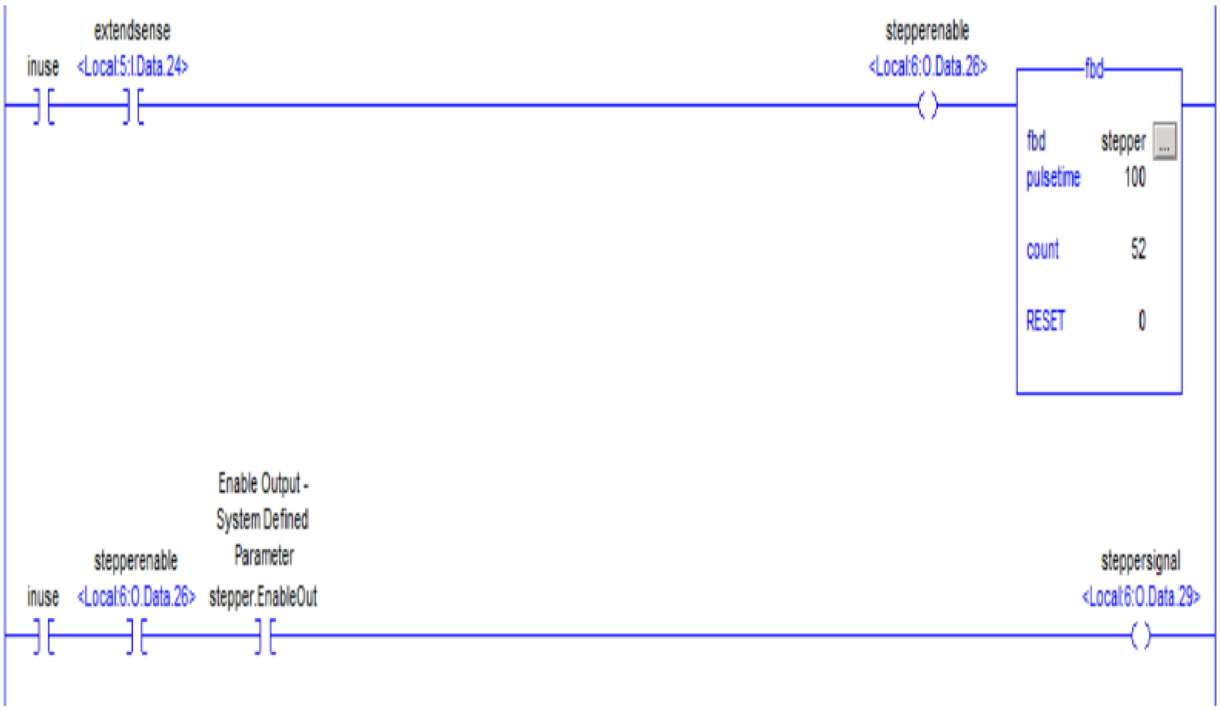
UDT



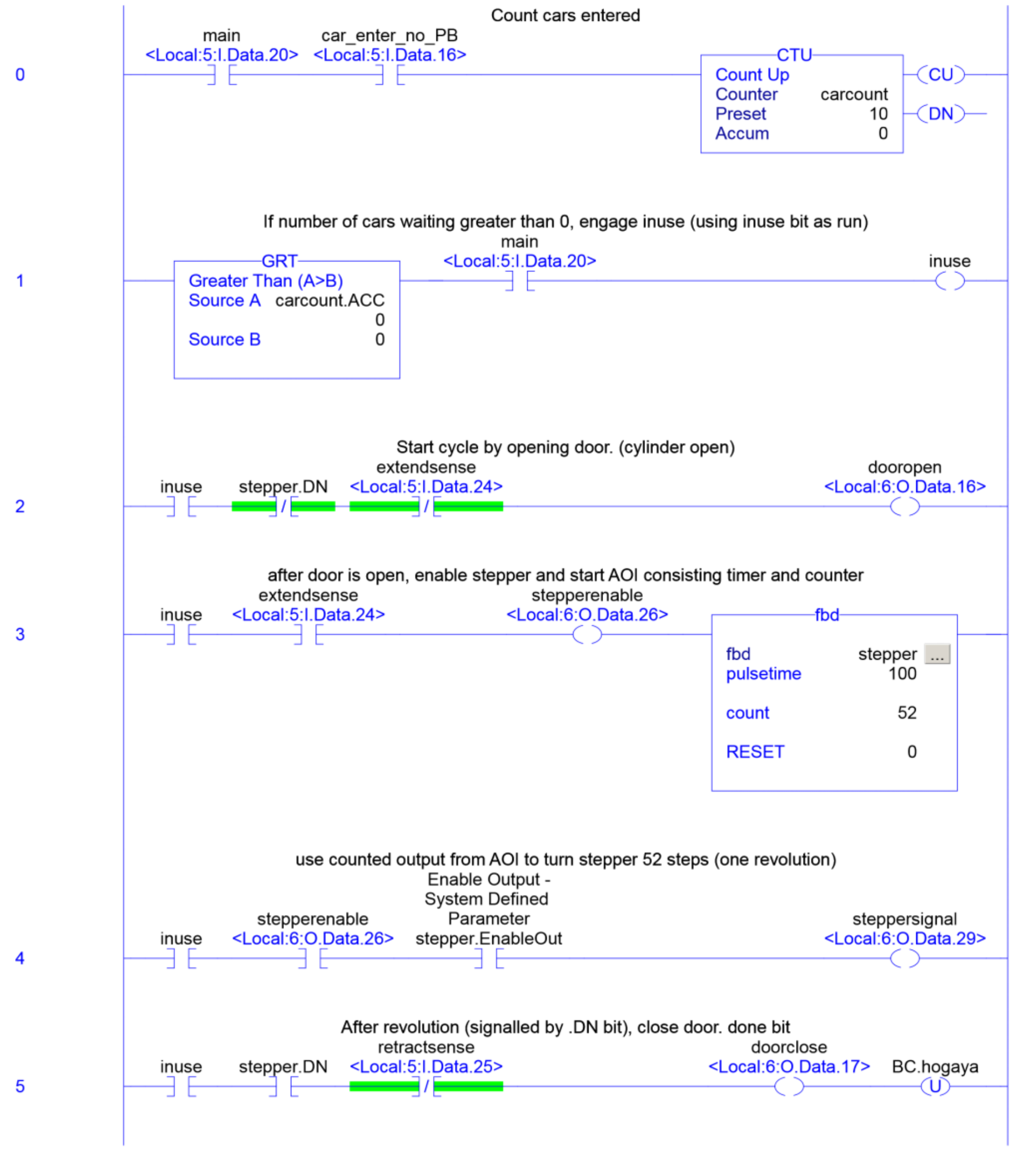
Logic Overview

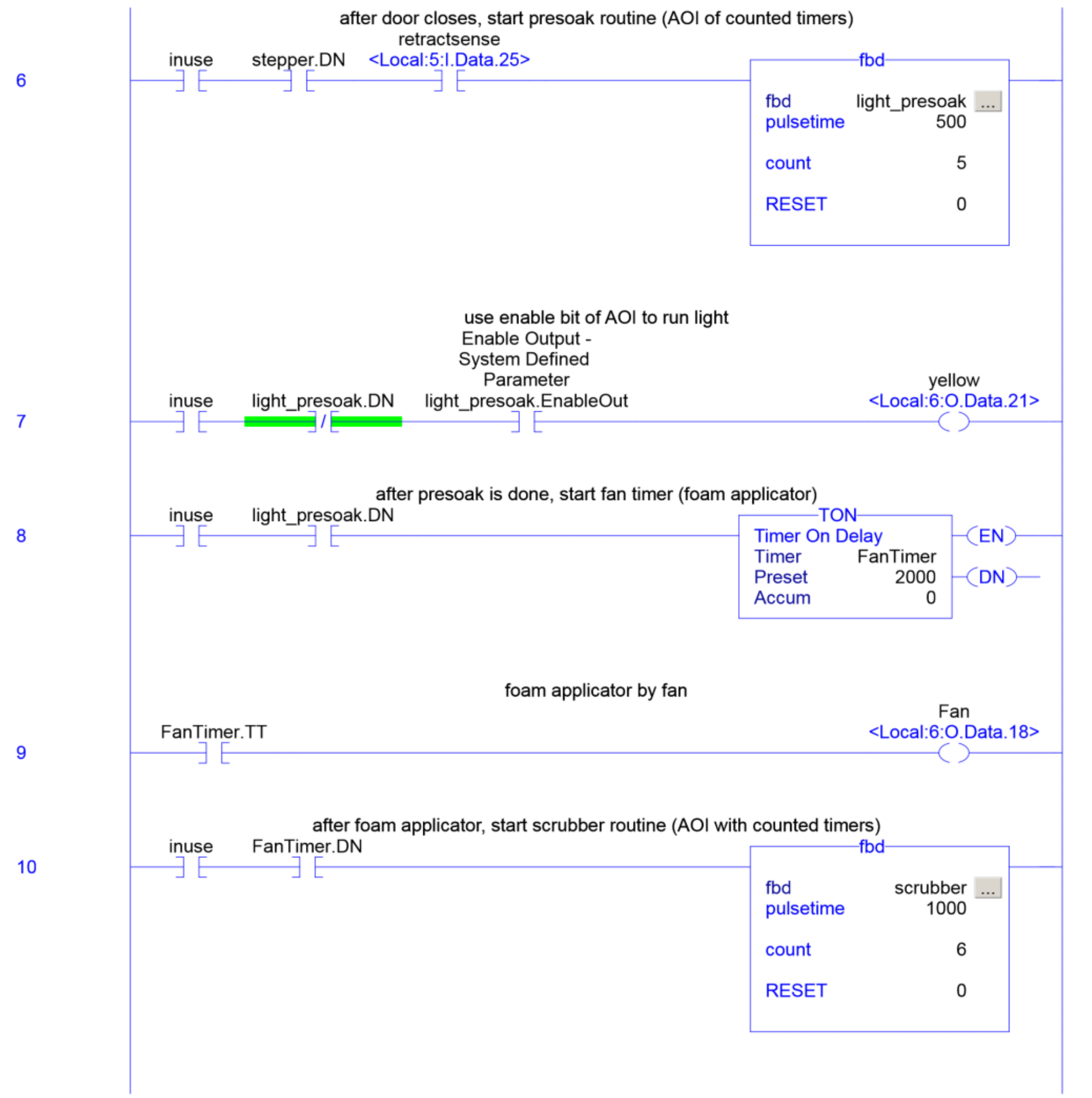
Ladder logic is mainly used in this project. The ladder logic as simple as if the input is true, the output is true. There are some debugging needs to be done to make flashing light work well and communication between PLC 1 and 2 is smooth and working fine. For example, when PLC 1 doesn’t have any car moving into PLC 2, PLC 2 not should not start working. Producing pulse to make light flash had some issues.

Implementation of FDB in Logic

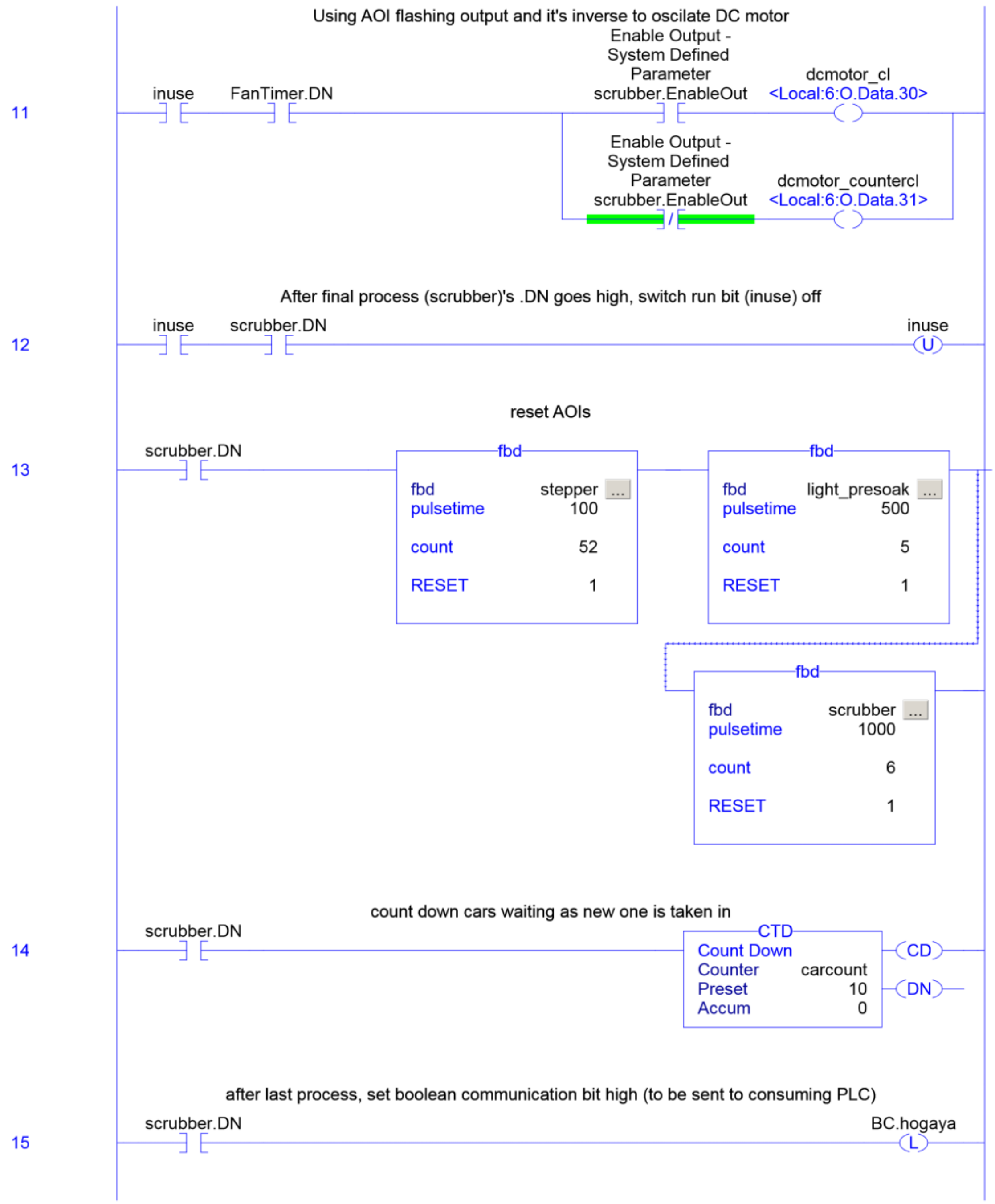


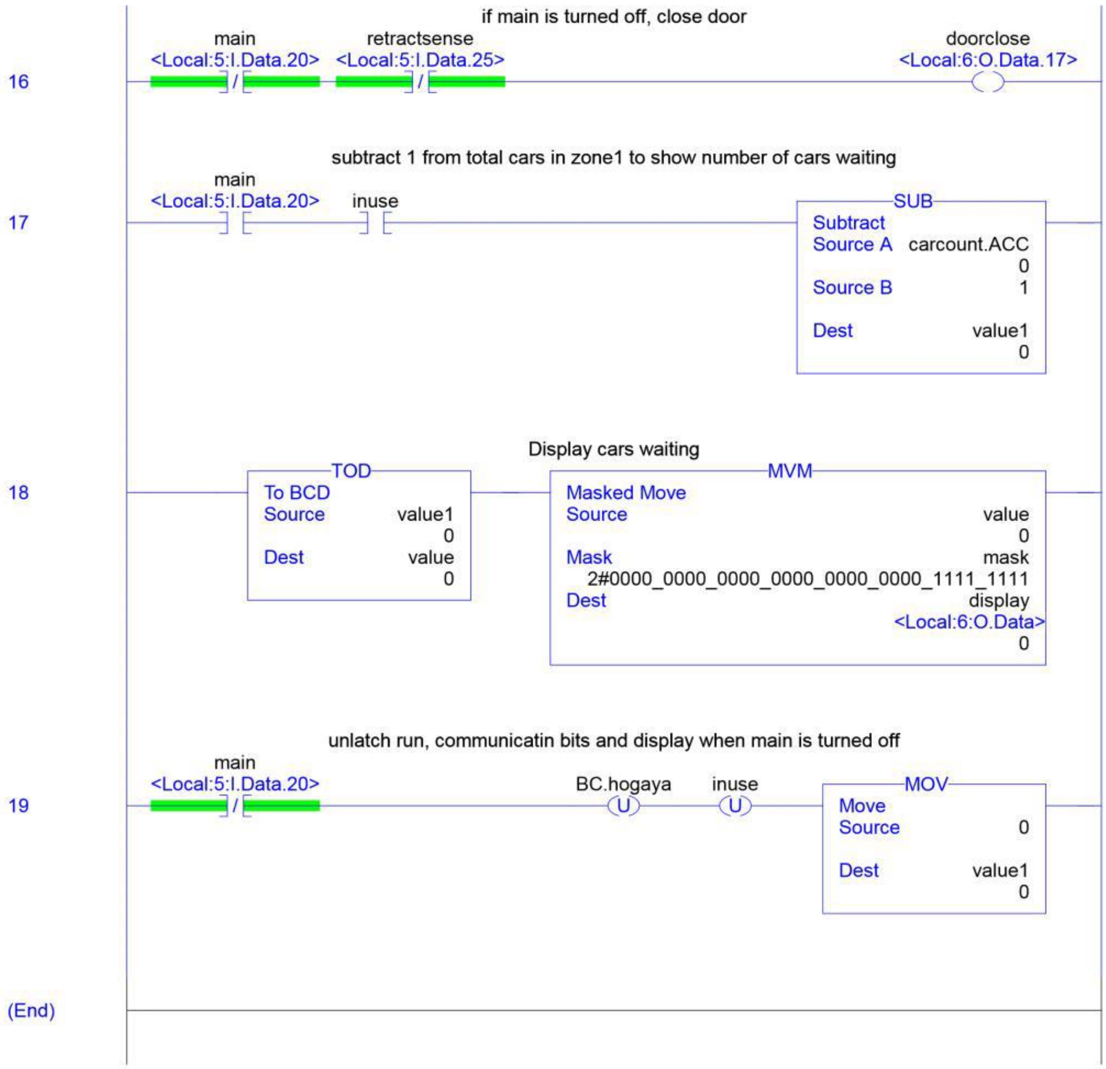
All of the logics used in this lab are as below.

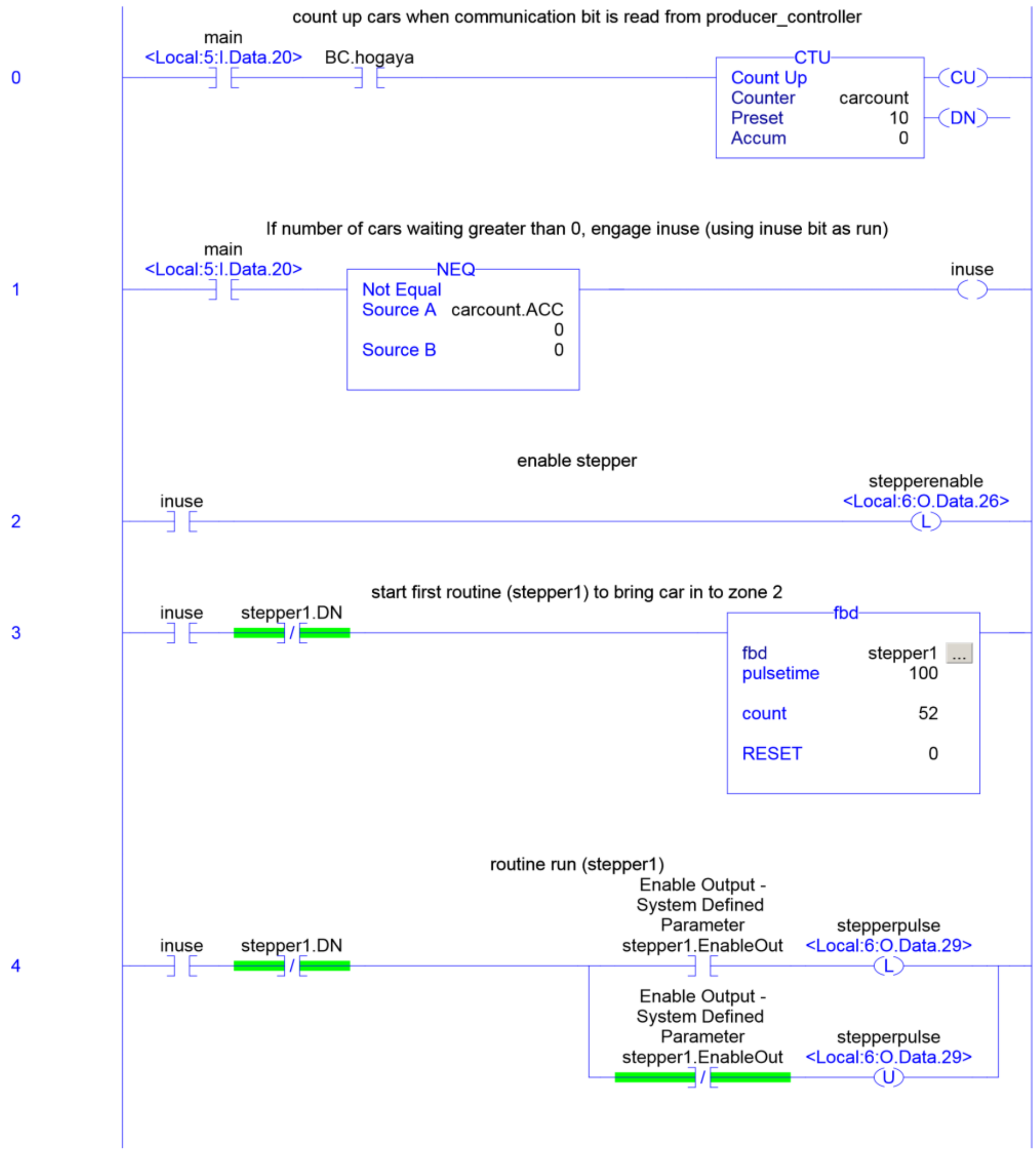

0
Count cars entered
main
<Local:5:I.Data.20>
car_enter_no_PB
<Local:5:I.Data.16>
CTU
Count Up
Counter carcount
Preset 10
Accum 0
CU
DN
1
If number of cars waiting greater than 0, engage inuse (using inuse bit as run)
GRT
Greater Than (A>B)
Source A carcount.ACC
0
Source B 0
main
<Local:5:I.Data.20>
inuse
2
Start cycle by opening door. (cylinder open)
inuse
stepper.DN
extendsense
<Local:5:I.Data.24>
dooropen
<Local:6:O.Data.16>
3
after door is open, enable stepper and start AOI consisting timer and counter
inuse
extendsense
<Local:5:I.Data.24>
stepperenable
<Local:6:O.Data.26>
fbd
fbd stepper
pulsetime 100
count 52
RESET 0
4
use counted output from AOI to turn stepper 52 steps (one revolution)
inuse
stepperenable
<Local:6:O.Data.26>
Enable Output -
System Defined
Parameter
stepper.EnableOut
steppersignal
<Local:6:O.Data.29>
5
After revolution (signalled by .DN bit), close door. done bit
inuse
stepper.DN
retractsense
<Local:5:I.Data.25>
doorclose
<Local:6:O.Data.17>
BC.hogaya
U

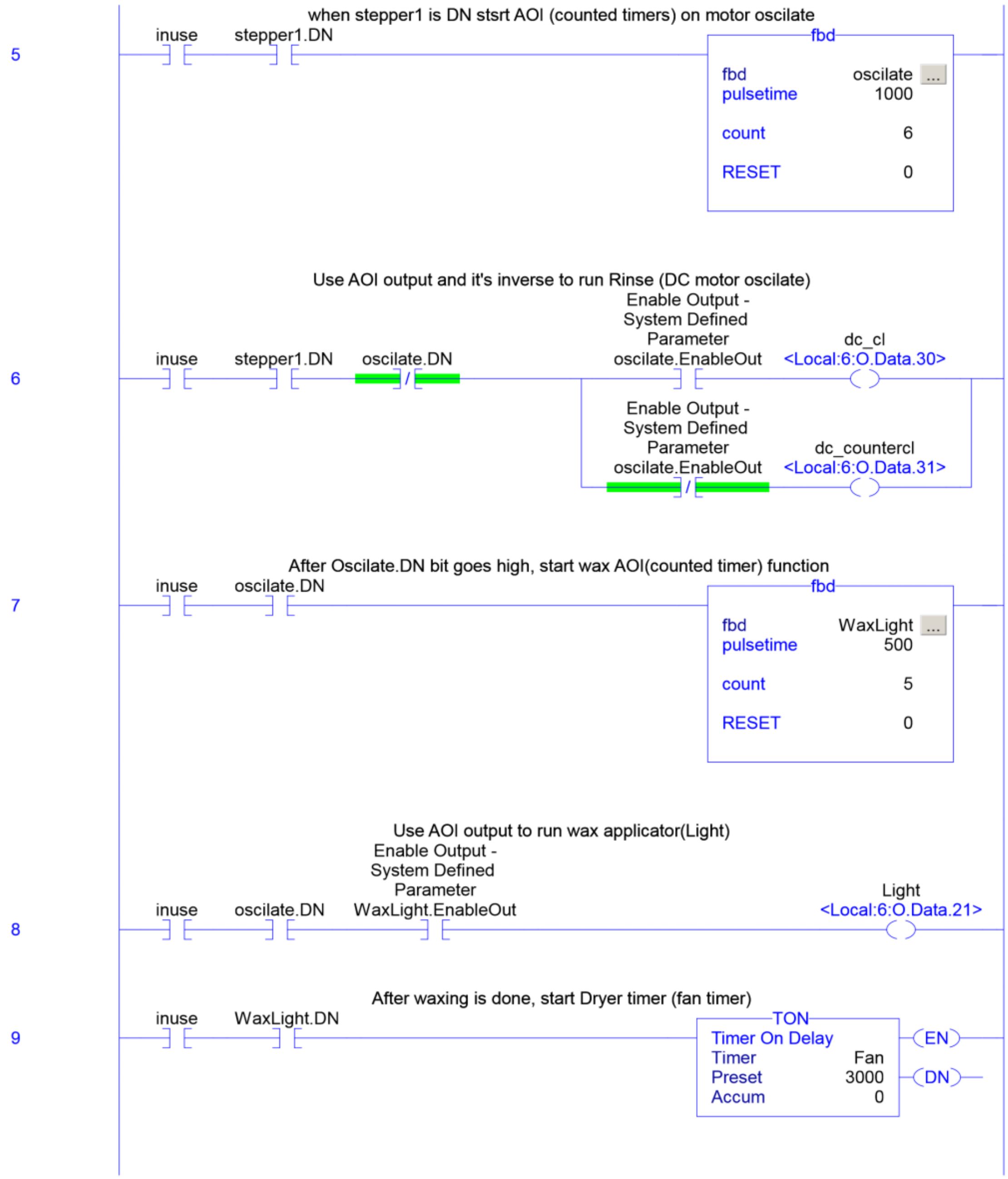


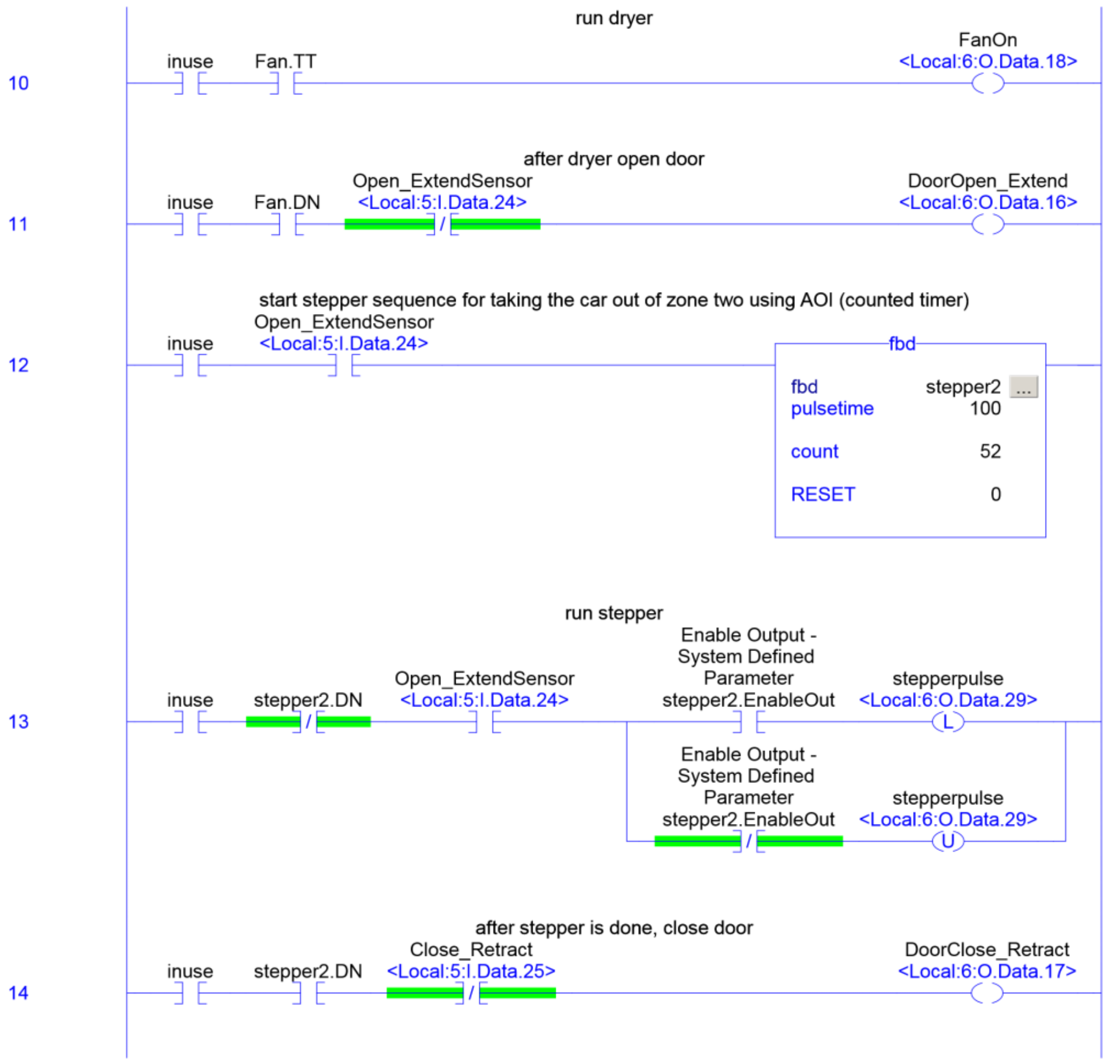
6
after door closes, start presoak routine (AOI of counted timers)
inuse
stepper.DN
retractsense
<Local:5:I.Data.25>
fbd
fbd light_presoak
pulsetime 500
count 5
RESET 0
7
use enable bit of AOI to run light
inuse
light_presoak.DN
Enable Output - System Defined Parameter
light_presoak.EnableOut
yellow
<Local:6:O.Data.21>
8
after presoak is done, start fan timer (foam applicator)
inuse
light_presoak.DN
TON
Timer On Delay
Timer FanTimer
Preset 2000
Accum 0
EN
DN
9
foam applicator by fan
FanTimer.TT
Fan
<Local:6:O.Data.18>
10
after foam applicator, start scrubber routine (AOI with counted timers)
inuse
FanTimer.DN
fbd
fbd scrubber
pulsetime 1000
count 6
RESET 0

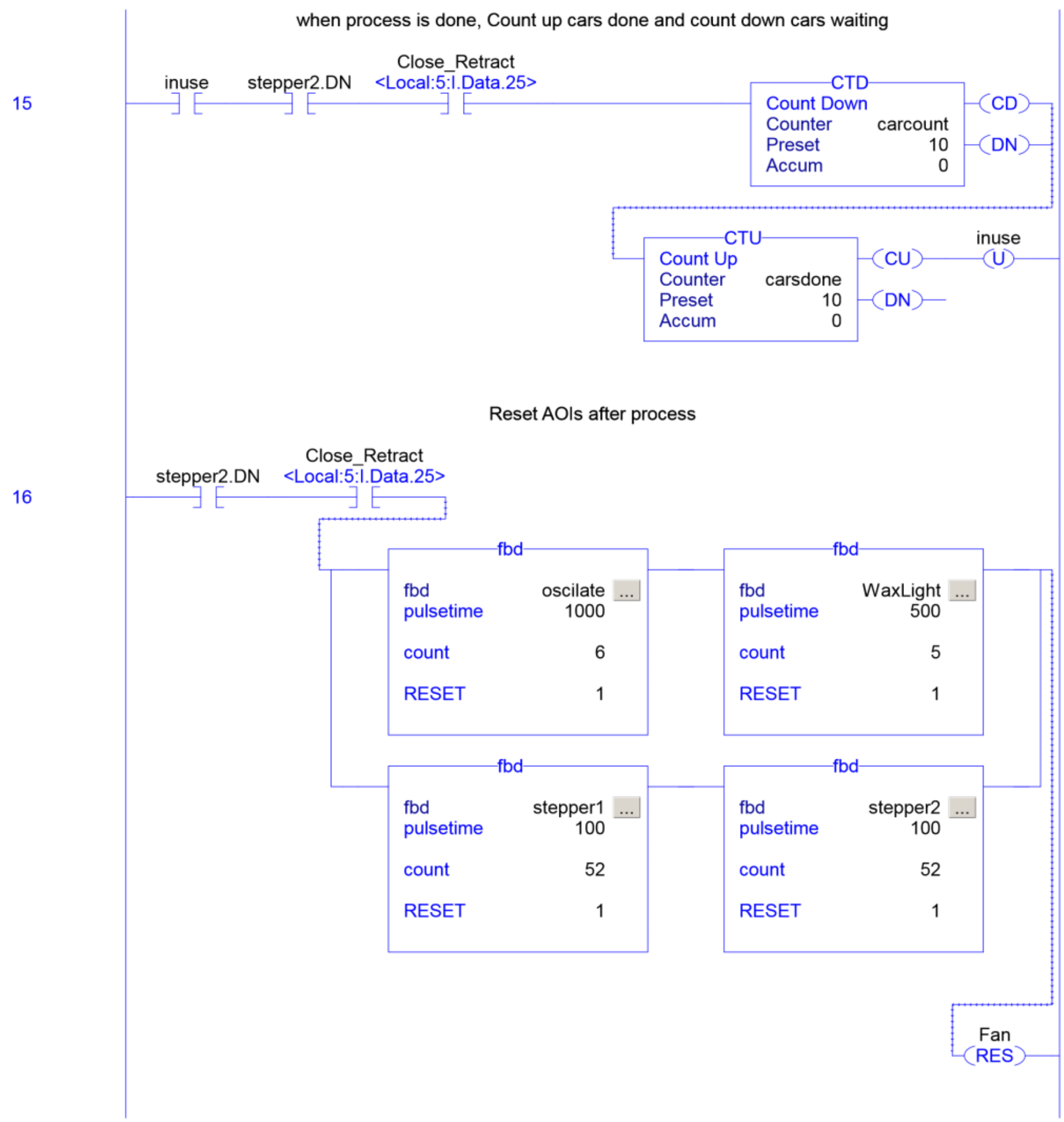

Using AOI flashing output and it's inverse to oscilate DC motor
11
inuse
FanTimer.DN
Enable Output - System Defined Parameter
scrubber.EnableOut
dcmotor_cl
<Local:6:O.Data.30>
Enable Output - System Defined Parameter
scrubber.EnableOut
dcmotor_countercl
<Local:6:O.Data.31>
After final process (scrubber)'s .DN goes high, switch run bit (inuse) off
12
inuse
scrubber.DN
inuse
U
reset AOIs
13
scrubber.DN
fbd
fbd stepper ...
pulsetime 100
count 52
RESET 1
fbd
fbd light_presoak ...
pulsetime 500
count 5
RESET 1
fbd
fbd scrubber ...
pulsetime 1000
count 6
RESET 1
count down cars waiting as new one is taken in
14
scrubber.DN
CTD
Count Down
Counter carcount
Preset 10
Accum 0
CD
DN
after last process, set boolean communication bit high (to be sent to consuming PLC)
15
scrubber.DN
BC.hogaya
L

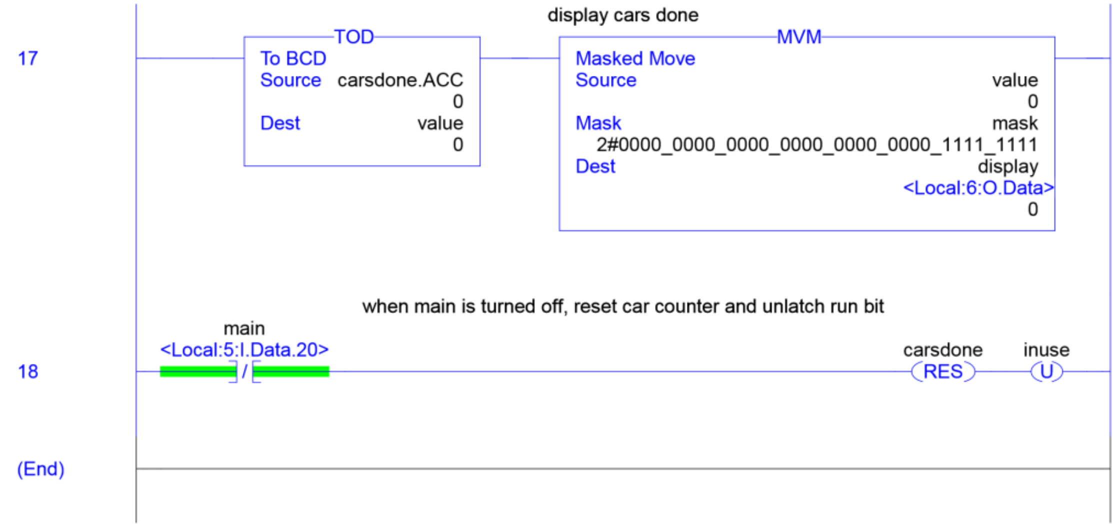

if main is turned off, close door
16
main
<Local:5:I.Data.20>
retractsense
<Local:5:I.Data.25>
doorclose
<Local:6:O.Data.17>
subtract 1 from total cars in zone1 to show number of cars waiting
17
main
<Local:5:I.Data.20>
inuse
SUB
Subtract
Source A carcount.ACC
0
Source B 1
Dest value1
0
Display cars waiting
18
TOD
To BCD
Source value1
0
Dest value
0
MVM
Masked Move
Source value
0
Mask mask
2#0000_0000_0000_0000_0000_0000_1111_1111
Dest display
<Local:6:O.Data>
0
unlatch run, communicatin bits and display when main is turned off
19
main
<Local:5:I.Data.20>
BC.hogaya
U
inuse
U
MOV
Move
Source 0
Dest value1
0
(End)


count up cars when communication bit is read from producer_controller
0
main
<Local:5:I.Data.20>
BC.hogaya
CTU
Count Up
Counter carcount
Preset 10
Accum 0
CU
DN
If number of cars waiting greater than 0, engage inuse (using inuse bit as run)
1
main
<Local:5:I.Data.20>
NEQ
Not Equal
Source A carcount.ACC
0
Source B 0
inuse
enable stepper
2
inuse
stepperenable
<Local:6:O.Data.26>
L
start first routine (stepper1) to bring car in to zone 2
3
inuse
stepper1.DN
fbd
fbd stepper1 ...
pulsetime 100
count 52
RESET 0
routine run (stepper1)
4
inuse
stepper1.DN
Enable Output -
System Defined
Parameter
stepper1.EnableOut
stepperpulse
<Local:6:O.Data.29>
L
Enable Output -
System Defined
Parameter
stepper1.EnableOut
stepperpulse
<Local:6:O.Data.29>
U


5
when stepper1 is DN stsrt AOI (counted timers) on motor oscilate
inuse
stepper1.DN
fbd
fbd oscilate
pulsetime 1000
count 6
RESET 0
6
Use AOI output and it's inverse to run Rinse (DC motor oscilate)
inuse
stepper1.DN
oscilate.DN
Enable Output - System Defined Parameter
oscilate.EnableOut
dc_cl
<Local:6:O.Data.30>
Enable Output - System Defined Parameter
oscilate.EnableOut
dc_countercl
<Local:6:O.Data.31>
7
After Oscilate.DN bit goes high, start wax AOI(counted timer) function
inuse
oscilate.DN
fbd
fbd WaxLight
pulsetime 500
count 5
RESET 0
8
Use AOI output to run wax applicator(Light)
inuse
oscilate.DN
Enable Output - System Defined Parameter
WaxLight.EnableOut
Light
<Local:6:O.Data.21>
9
After waxing is done, start Dryer timer (fan timer)
inuse
WaxLight.DN
TON
Timer On Delay
Timer Fan
Preset 3000
Accum 0
EN
DN


run dryer
10
inuse
Fan.TT
FanOn
<Local:6:O.Data.18>
after dryer open door
11
inuse
Fan.DN
Open_ExtendSensor
<Local:5:I.Data.24>
DoorOpen_Extend
<Local:6:O.Data.16>
start stepper sequence for taking the car out of zone two using AOI (counted timer)
12
inuse
Open_ExtendSensor
<Local:5:I.Data.24>
fbd
fbd stepper2
pulsetime 100
count 52
RESET 0
run stepper
13
inuse
stepper2.DN
Open_ExtendSensor
<Local:5:I.Data.24>
Enable Output -
System Defined
Parameter
stepper2.EnableOut
stepperpulse
<Local:6:O.Data.29>
L
Enable Output -
System Defined
Parameter
stepper2.EnableOut
stepperpulse
<Local:6:O.Data.29>
U
after stepper is done, close door
14
inuse
stepper2.DN
Close_Retract
<Local:5:I.Data.25>
DoorClose_Retract
<Local:6:O.Data.17>


when process is done, Count up cars done and count down cars waiting
15
inuse
stepper2.DN
Close_Retract
<Local:5:I.Data.25>
CTD
Count Down
Counter carcount
Preset 10
Accum 0
CD
DN
CTU
Count Up
Counter carsdone
Preset 10
Accum 0
CU
DN
inuse
U
Reset AOIs after process
16
stepper2.DN
Close_Retract
<Local:5:I.Data.25>
fbd
fbd oscilate
pulsetime 1000
count 6
RESET 1
fbd
fbd WaxLight
pulsetime 500
count 5
RESET 1
fbd
fbd stepper1
pulsetime 100
count 52
RESET 1
fbd
fbd stepper2
pulsetime 100
count 52
RESET 1
Fan
RES



17
display cars done
TOD
To BCD
Source carsdone.ACC
0
Dest value
0
MVM
Masked Move
Source value
0
Mask mask
2#0000_0000_0000_0000_0000_0000_1111_1111
Dest display
<Local:6:O.Data>
0
18
when main is turned off, reset car counter and unlatch run bit
main
<Local:5:I.Data.20>
carsdone
RES
inuse
U
(End)

Challenges in execution

Structure Text logic is straightforward in theory. When try to run it, the counter isn't working well. Function Block Diagram is working well but small issues resetting. After some fixing, the reset works fine. When FBD of the first output series in the rung resets first, the simulation doesn't work. The scan time between PLC 1 and 2, 20ms PRI, isn't small enough to catch OTE bit of PLC 1. In Zone2, branching input used to stepper pulse doesn't work. The two rungs are used instead.

Lessons learnt

The theoretical concepts are required to work on this project. The theory is applied to the project with simulation and implementation. Through trial and error, the silly mistakes are made to learn the correct logic. Debugging is repeated until finishing the programming process completely. Latch should be used when communication between PLCs are required. Branching input and output should be used carefully. It is also useful to know PLC has other languages to be used in addition to ladder logic diagram.

Conclusion

The simulation was successful after the error were fixed. The main objective of this lab is to know how to apply PLC knowledge in industries. In this project, there are many add on instructions to use besides ladder logic diagram. With simulation and the automated car wash response, the lab was completed theoretically and practically. Overall, the lab is finished completely without any major issues.