Siemens Counter Programs

INDEX

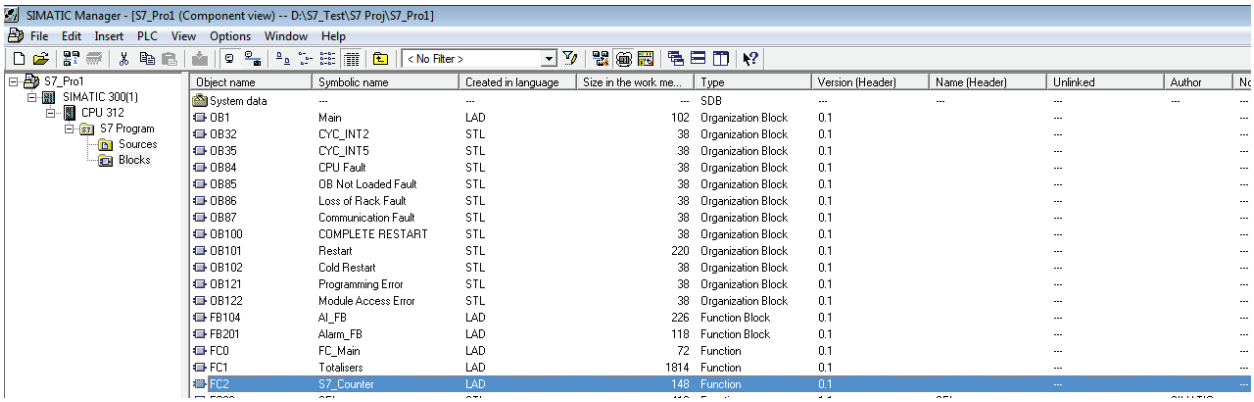
Following is the Counter logic, step by step explained with significance of each counter input.

A) How to Insert Counter, explained in point 1,2.

1) How to drag counter in program?

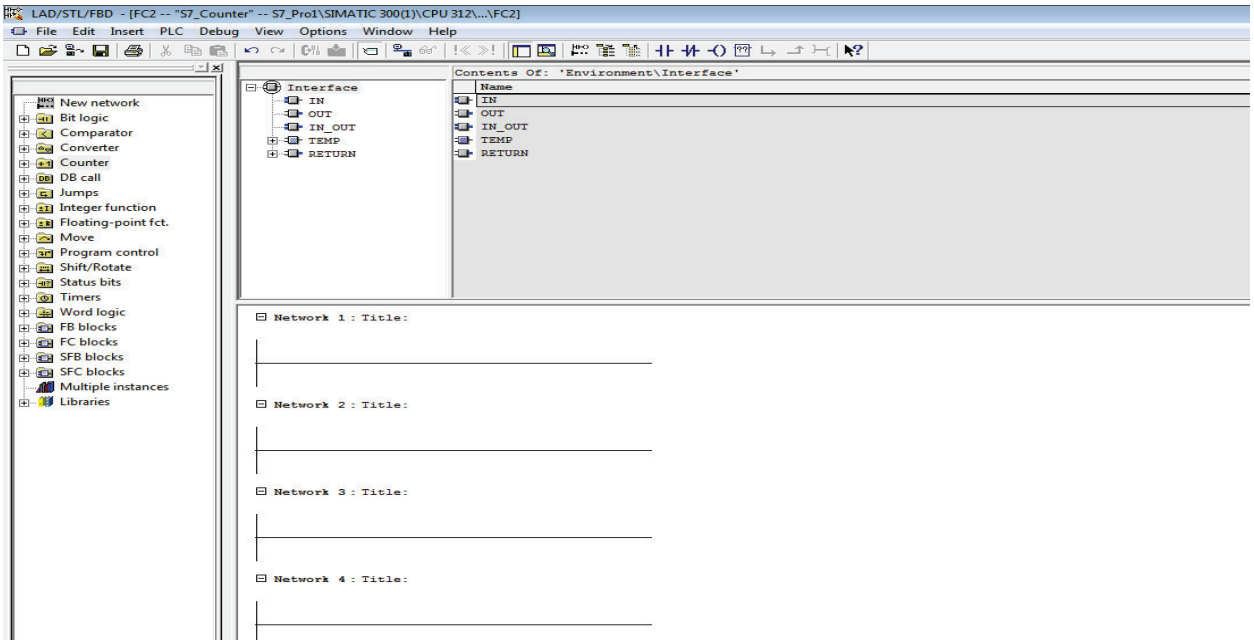
First open Simatic Manager software as shown below diag#1

Daig#1

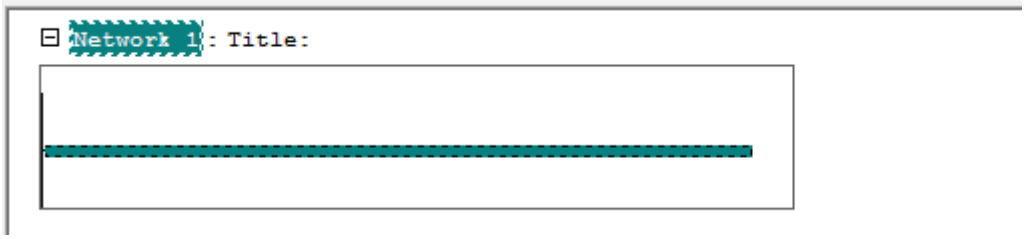


Then double click on FC2 (Function Block) and then following file appears shown in diag#2

Daig#2

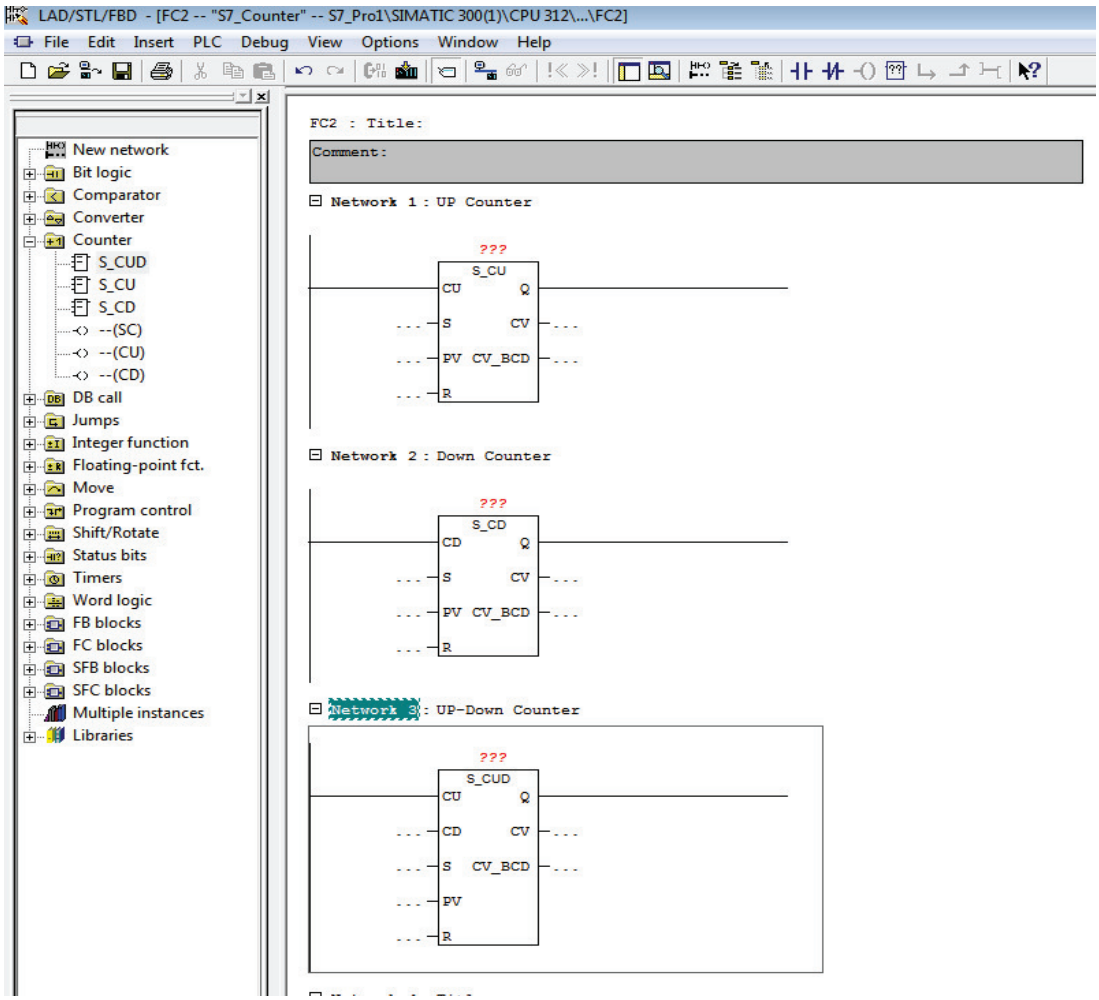


then double click on the line of network1 as shown below



After this goto left side of FC2 where it shows various features or instruction folder such as Bit Logic, Comparator etc. Exactly click on folder counter and it will show number of counter types as shown below and then double click on S_CU (UP Counter), this counter will appear with ??? over it, on network 1and similarly for S_CD (Down Counter) and S_CUD (UP Down Counter) as shown in network 1,2 & 3 each in below diag#3.

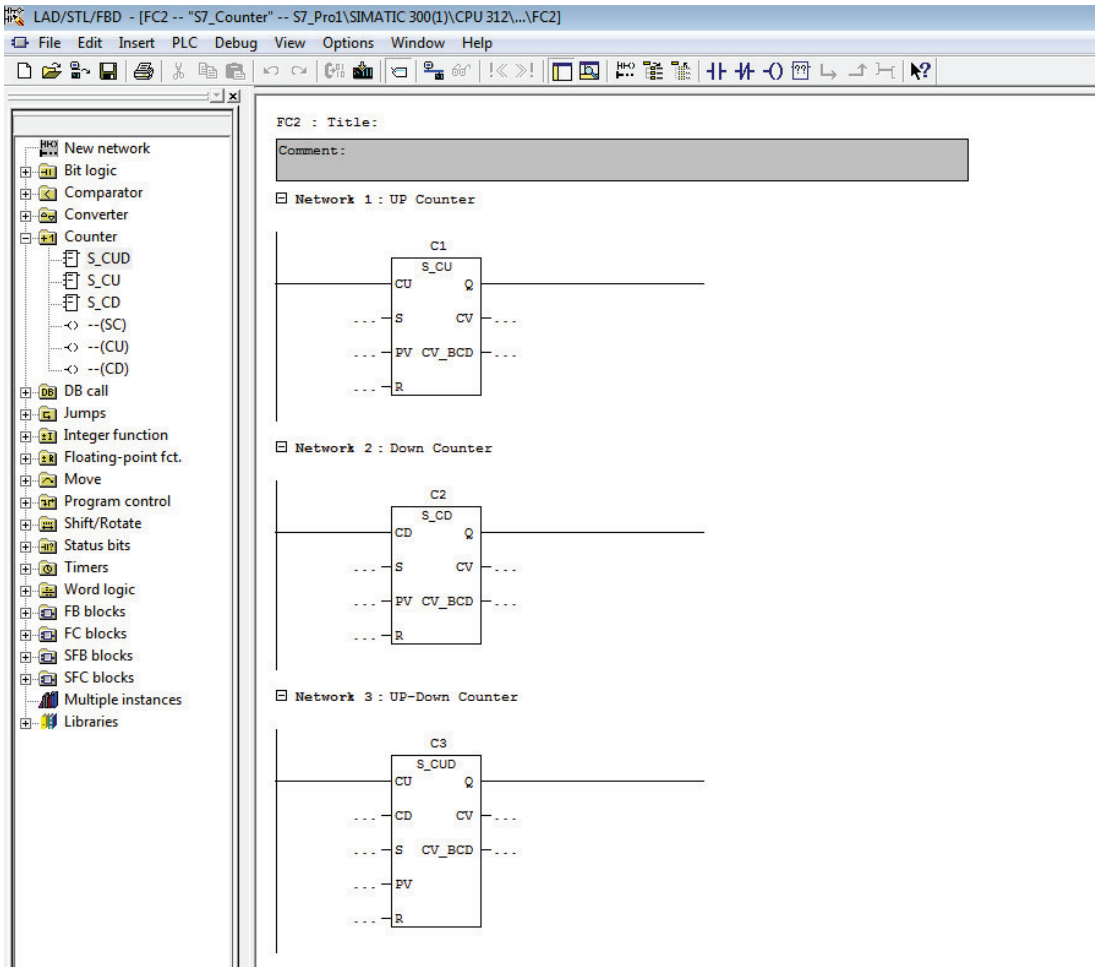
Daig#3



2) How to assign number to Counters?

Then click on ???type C1 for up counter (S_CU), C2 for down counter (S_CD) and C3 for up down counter (S_CUD) every network and then these counters are assigned with names or tag C1, C2 & C3 as shown below diag#4.

Daig#4



UP Counter Operation:

The counter C1 shown above diag#4 performs function of UP counter. Let us learn with example on number of lamps turned ON count.

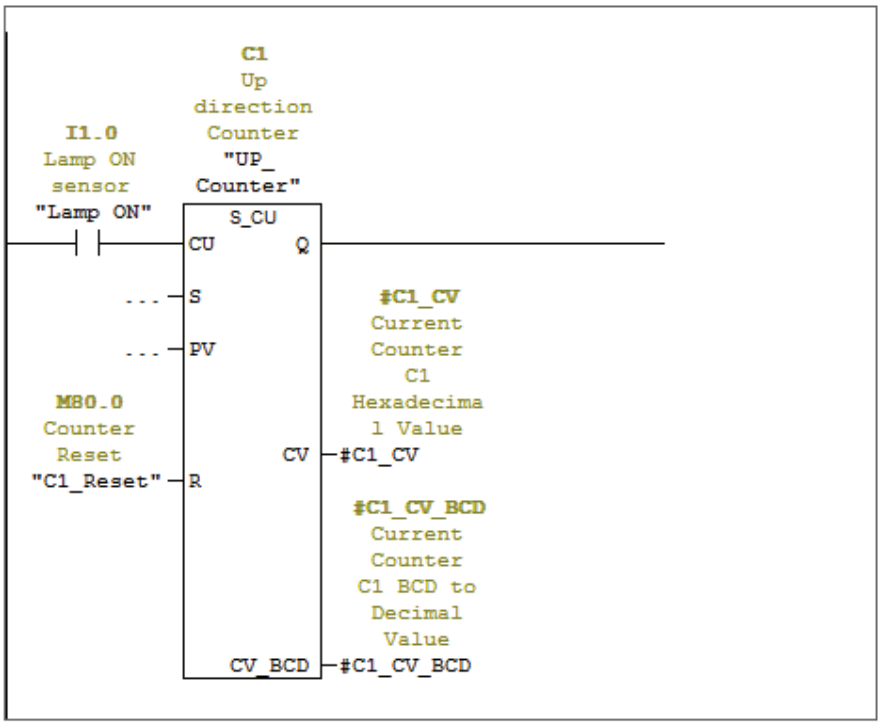
It can be understood from following diagram. The input CU is connected to real world input I1.0 which reads 1when a lamp turns on. Now consider here when 55 lamps are to be turned on and after all 55 lamps are turned on counter is reset and starts counting again. Let us focus on following diag#5 to get understanding of UP counter operation.

Daig#5

FC2 : Title:

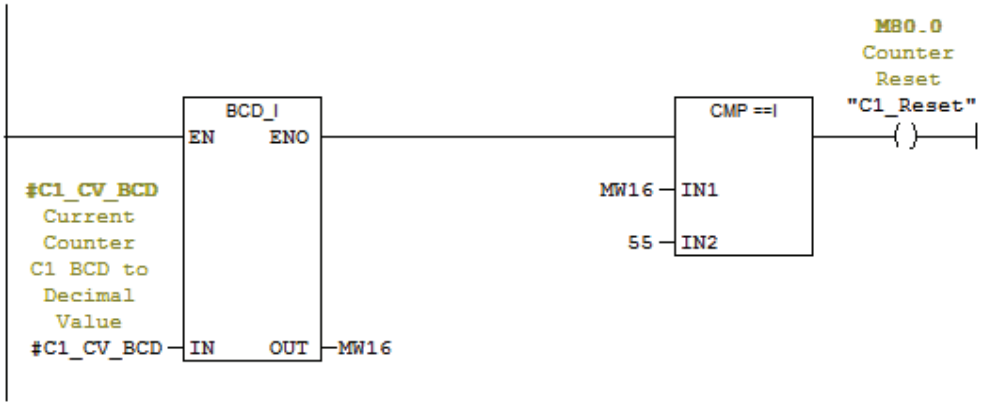
Comment:

Network 1: UP Counter



Network 2 : BCD to Decimal

MW16 has the decimal value and can be tagged into HMI to display counter value.

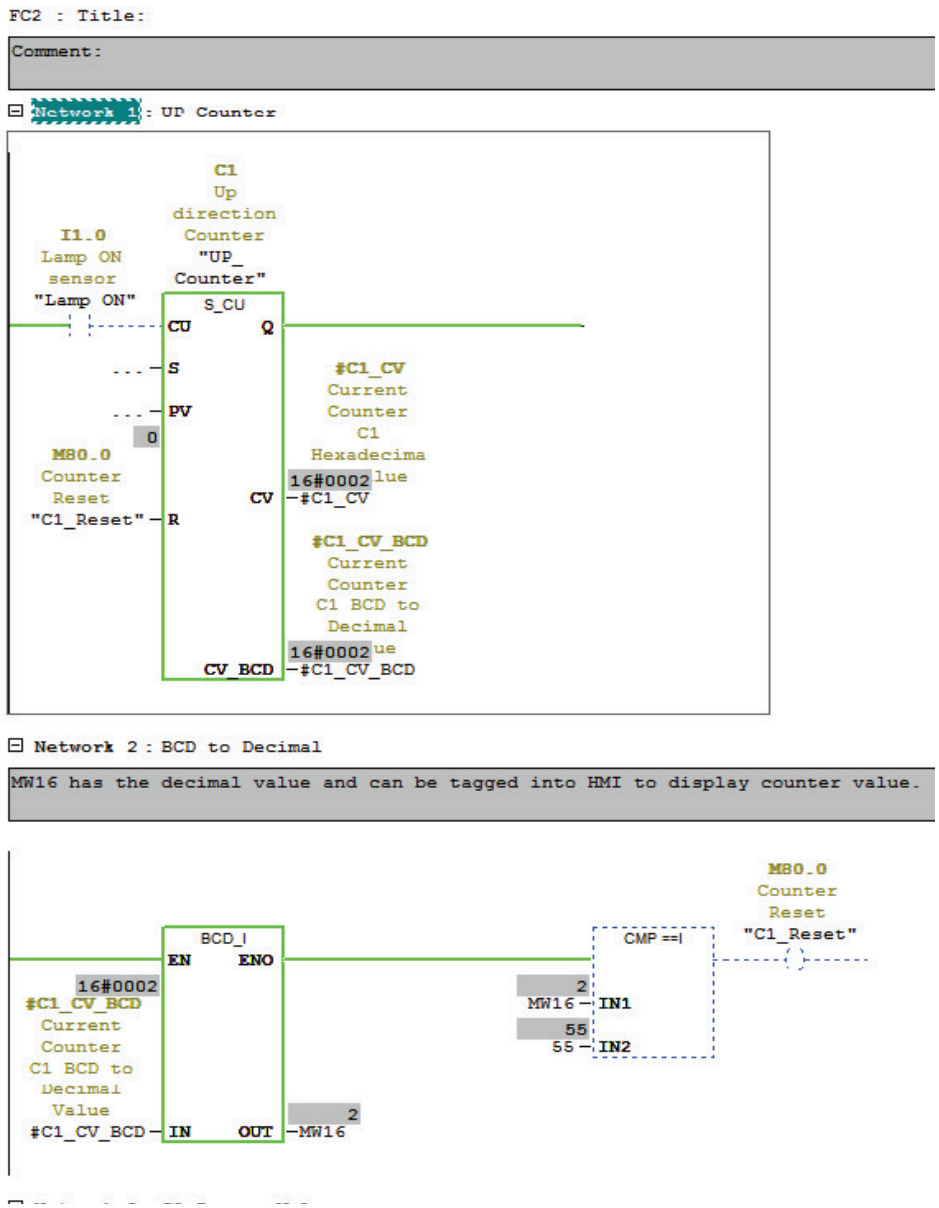


Now here real world input I1.0 is connected to CU and it turns on when it lamp switches are turned on one by one for every lamp respectively. As real world input I1.0 senses any switch

that is turned on CU input increments counter (here count is 2) as shown in below figure diag#6 after two lamps are on. The count BCD value is converted into decimal in more simpler way to understand and stored in internal memory MW16.

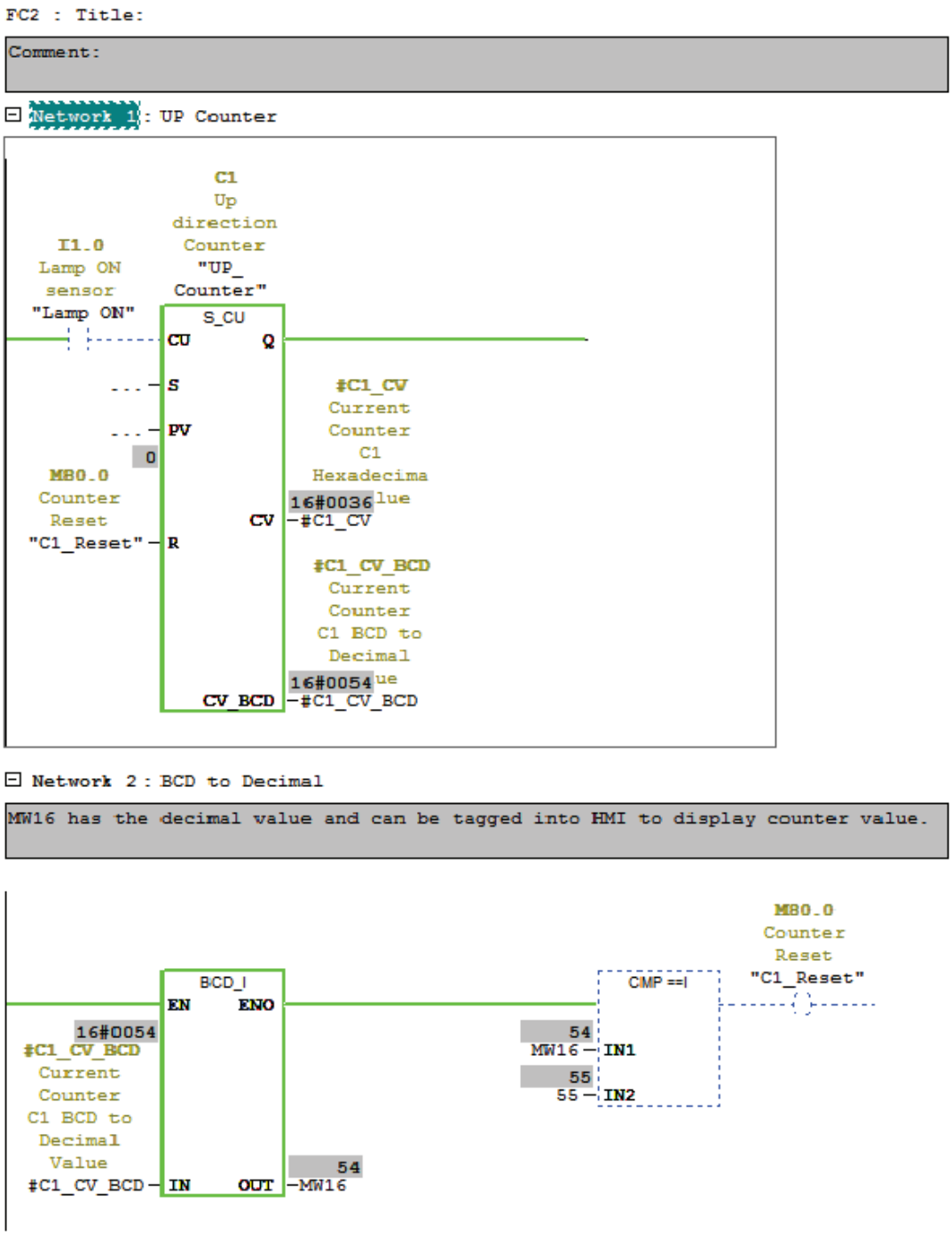
This count in internal memory is compared with the set point 55 counts and when count becomes 55 the counter is reset by internal memory bit M80.0 (C1_Reset).

Daig#6



As shown in below diag#7 the count is 54 as 54th lamp has turned ON.

Daig#7



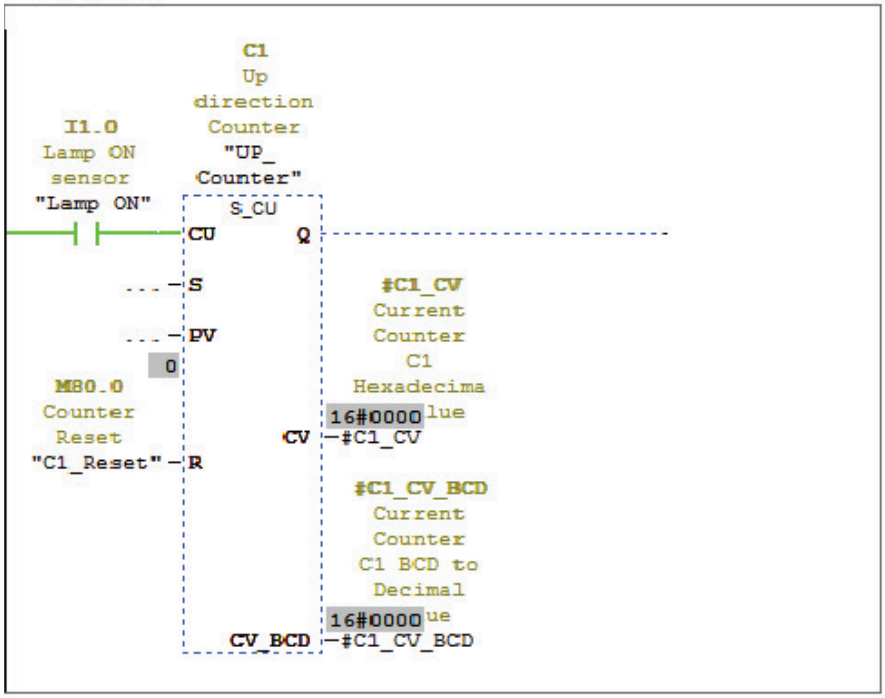
As 55th lamp turns ON the counter is reset to zero as shown below diag#8 by internal memory M80.0 (C1_Reset) and again counter is ready to count for next cycle.

Daig#8

FC2 : Title:

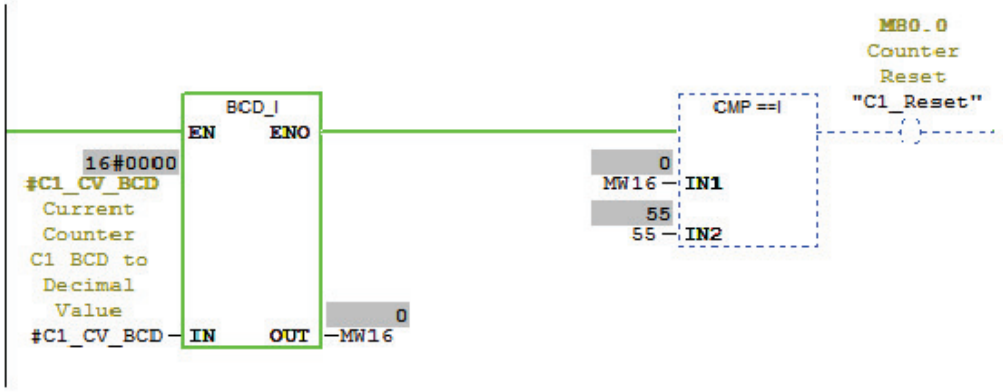
Comment:

Network 1: UP Counter



Network 2 : BCD to Decimal

MW16 has the decimal value and can be tagged into HMI to display counter value.



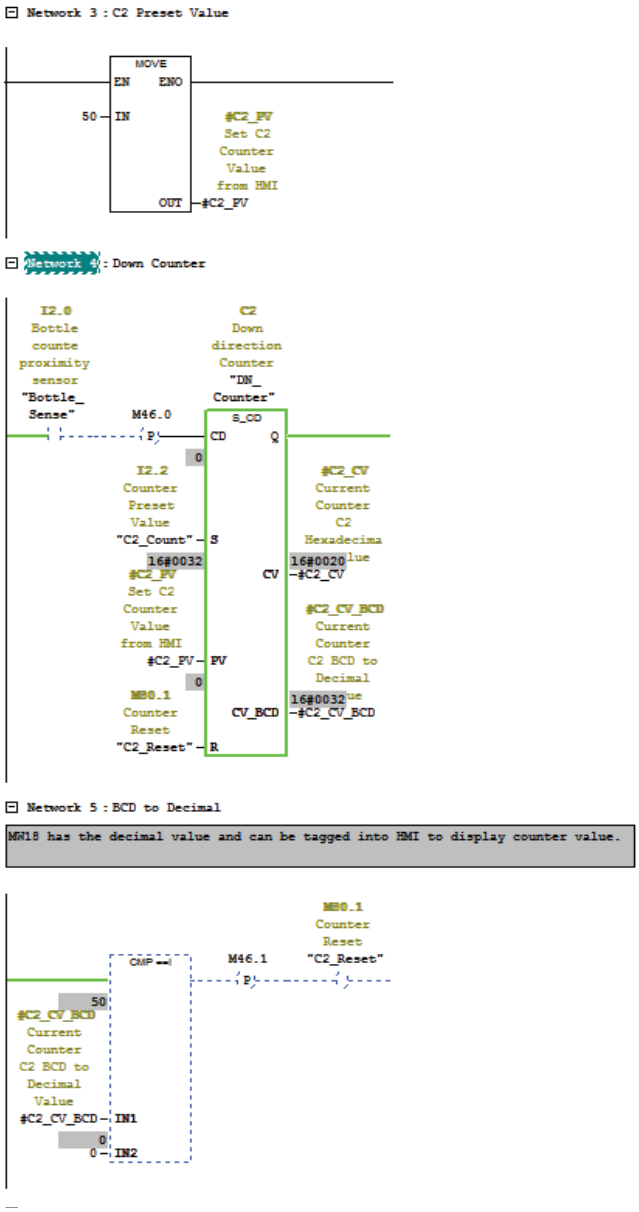
In this way Up Counter operation works!!

Down Counter

Now the drop drag procedure is same as explained for Up counter and you can follow diag#1 to diag#4 explained above where Down counter is shown too named C2.

The function of down is opposite to Up counter, here the counter is decremented by one if the signal state at input CD changes, the counter is reset if there is a 1 at input R. Let us understand using example of Bottle Sense as shown in diag#9 below

Daig#9

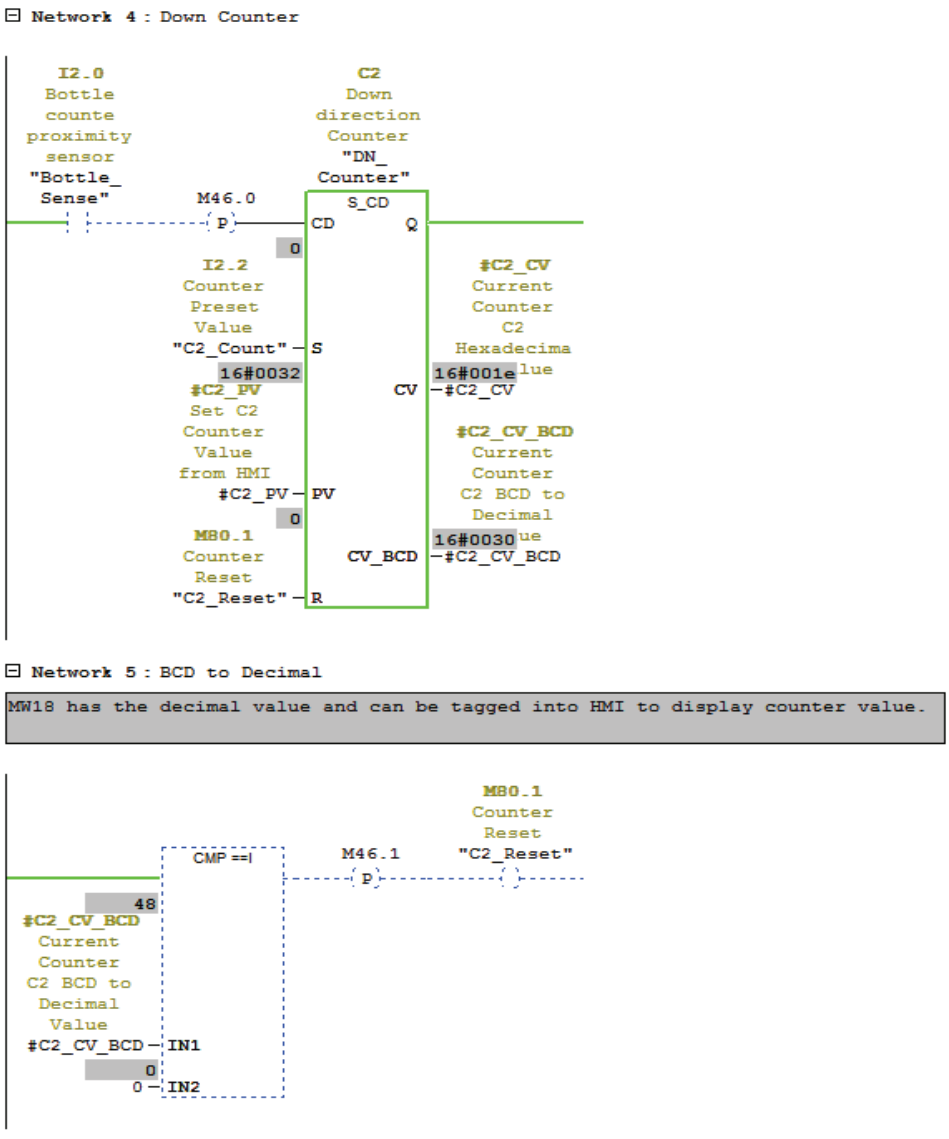


In the above diagram let us say 50 bottle are to be counted by real world input I2.0. When real world input I2.0 sense a passing of bottle by sensor, CD input reads 1 and decrements count by 1 until 50 bottles are sensed. The CD read 1 from internal memory M46.0 called rising edge which reads 1 from real world input I2.0.

The counter has to be preset to 50 and when real world input I2.2 is turned ON counter is loaded with value 50. It comes from local memory C2_PV loaded with 50 as shown in above diag#9 network3.

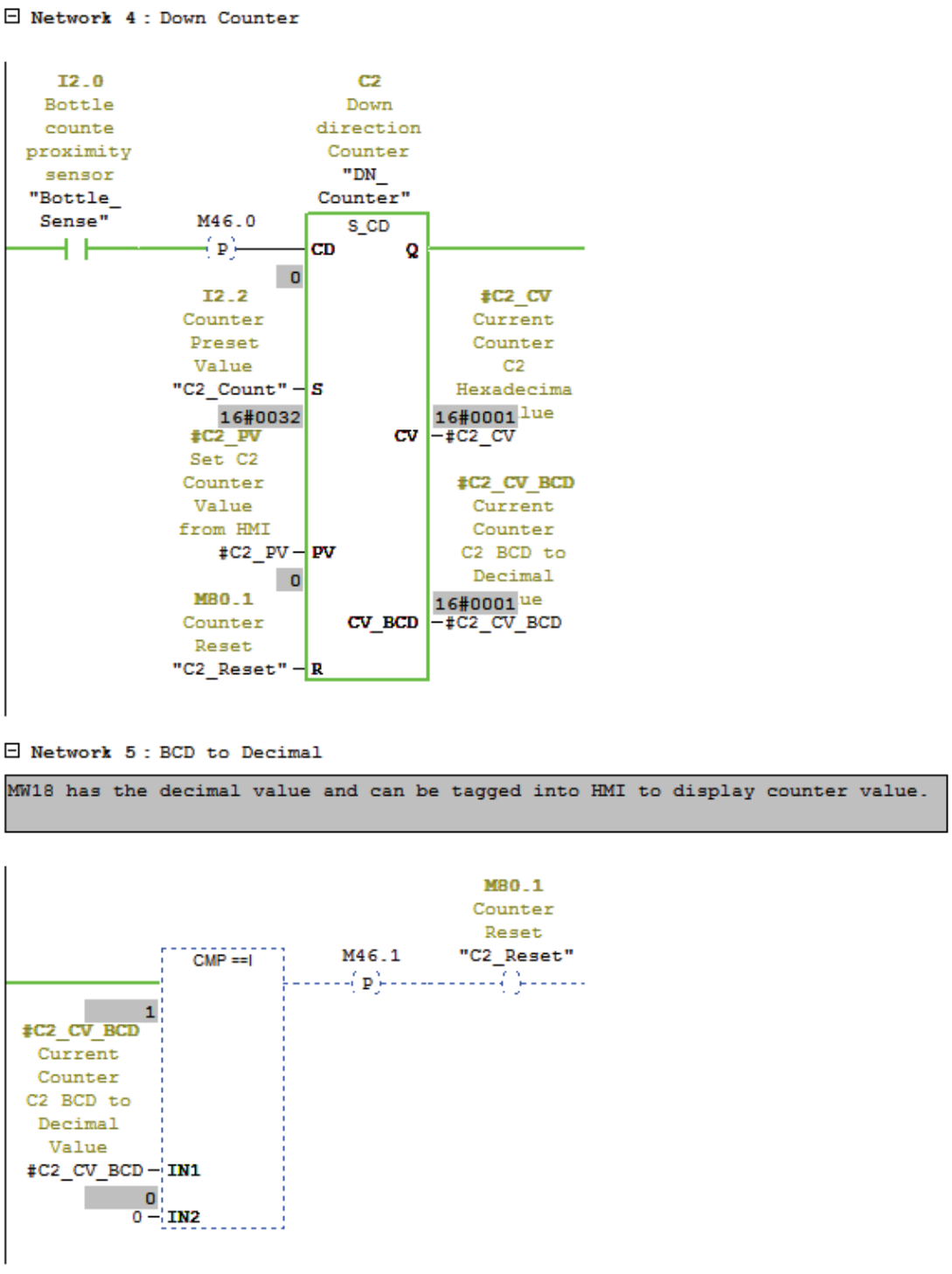
Now for example real world input I2.0 senses two bottles the count is 48 as shown in diag#10.

Daig#10



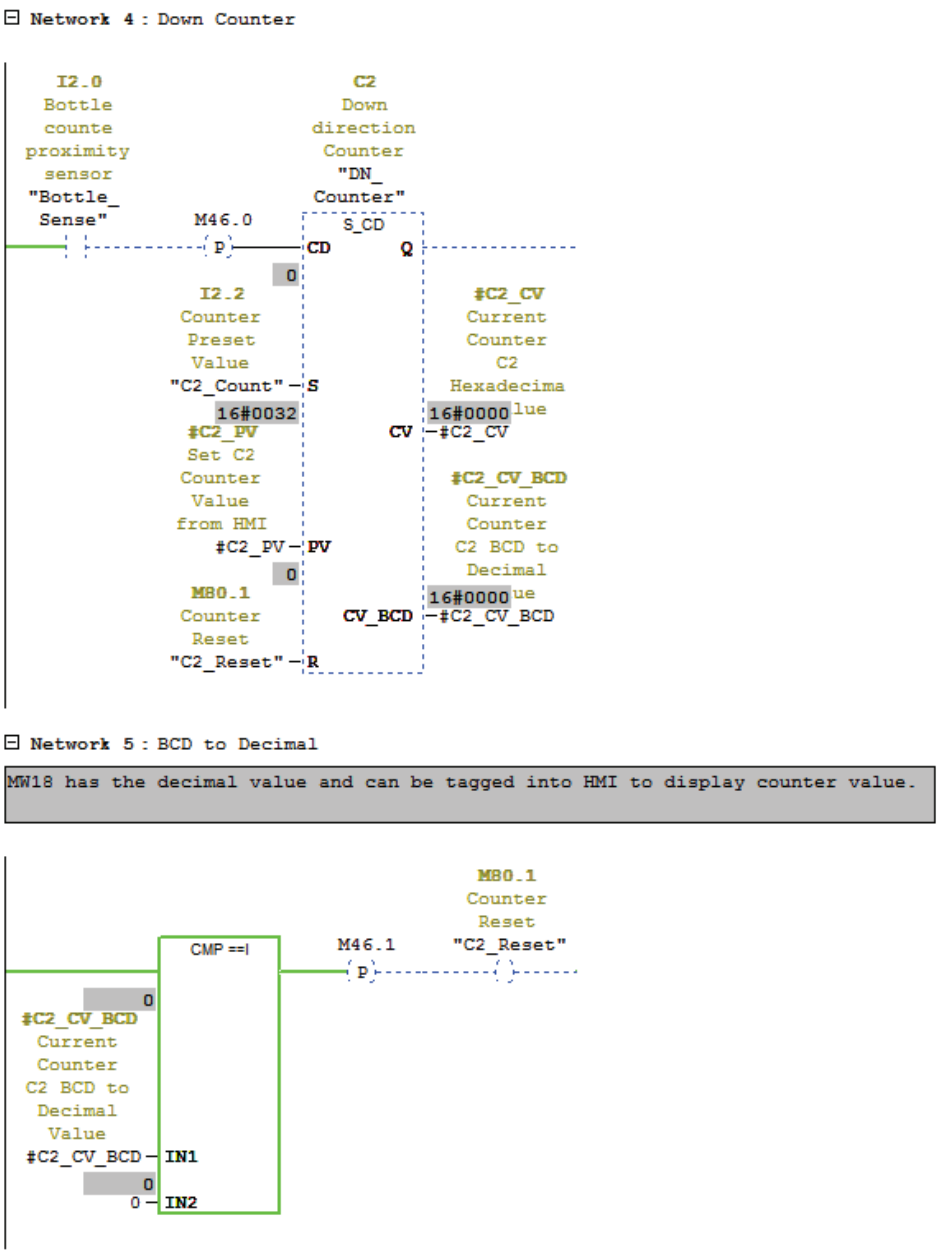
Likewise CD decrements until count becomes zero. Now 49 bottles are counted and count is 1 as shown in diag#11 below.

Daig#11



When last bottle 50 is counted the counter resets to zero by internal memory M80.1 (C2_Reset) as shown below diag#12

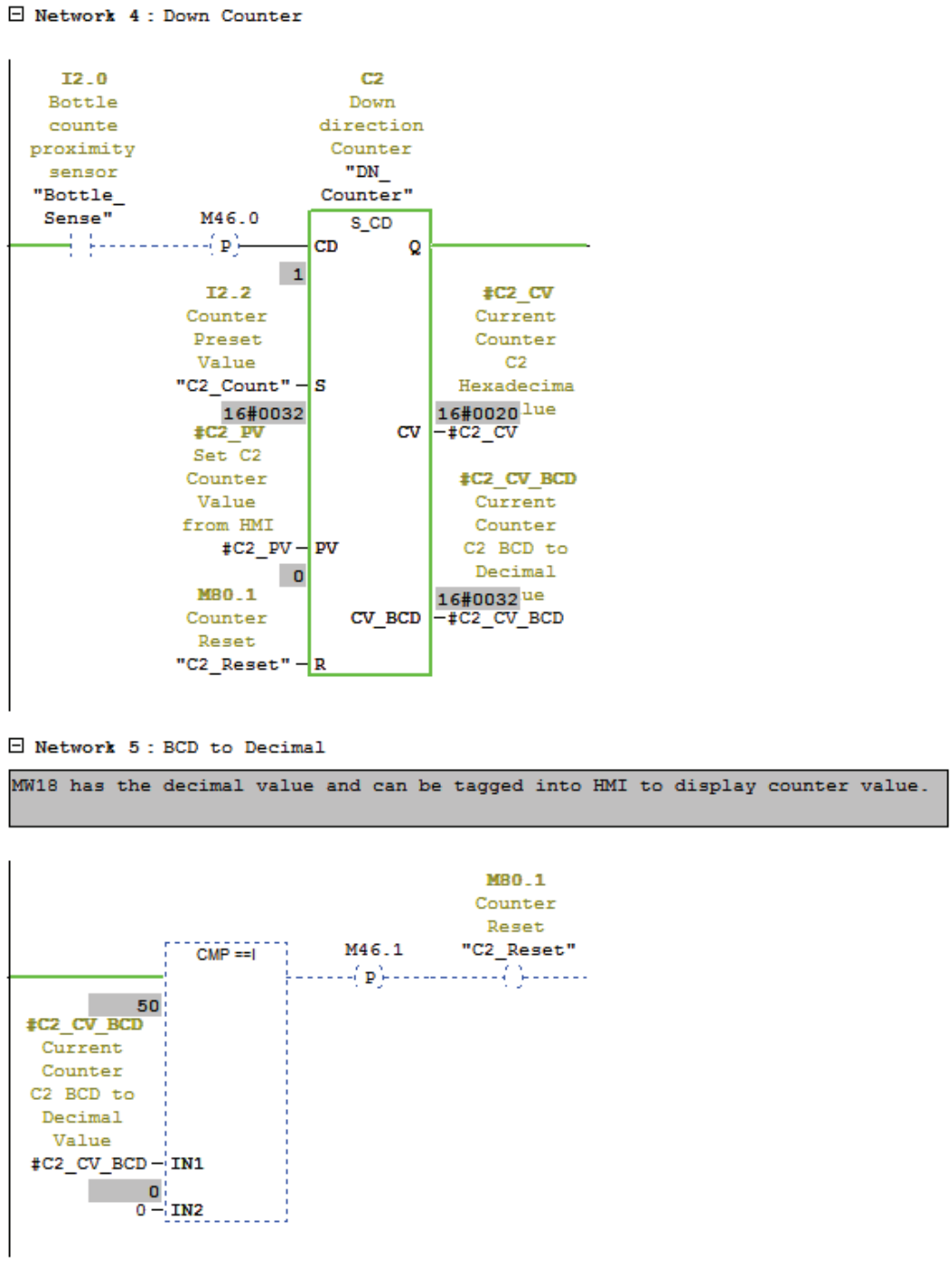
Daig#12



Now again real world input I2.2 needs to ON/OFF (i.e. push button action) so that counter is preset to 50 again and next 50 bottles are count down as shown in diag#13, but in actual project or programming logic you should write such a logic that counter is preset when count becomes

zero, you can use internal memory or whatever is better. Here using is just to understand basics!!

Daig#13



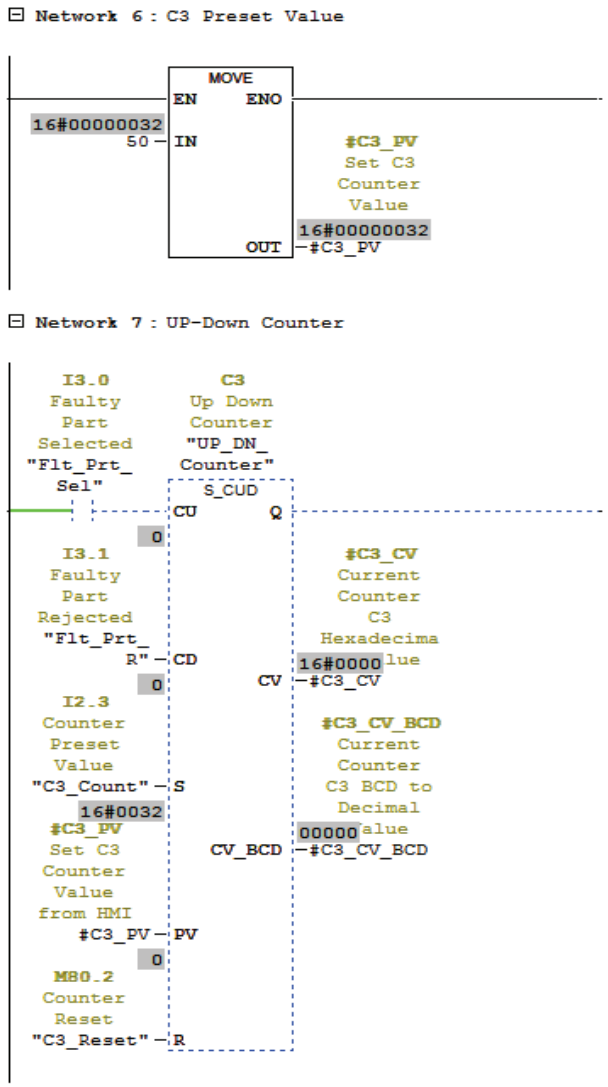
Here ends operation of Down counter!!

UP Down Counter

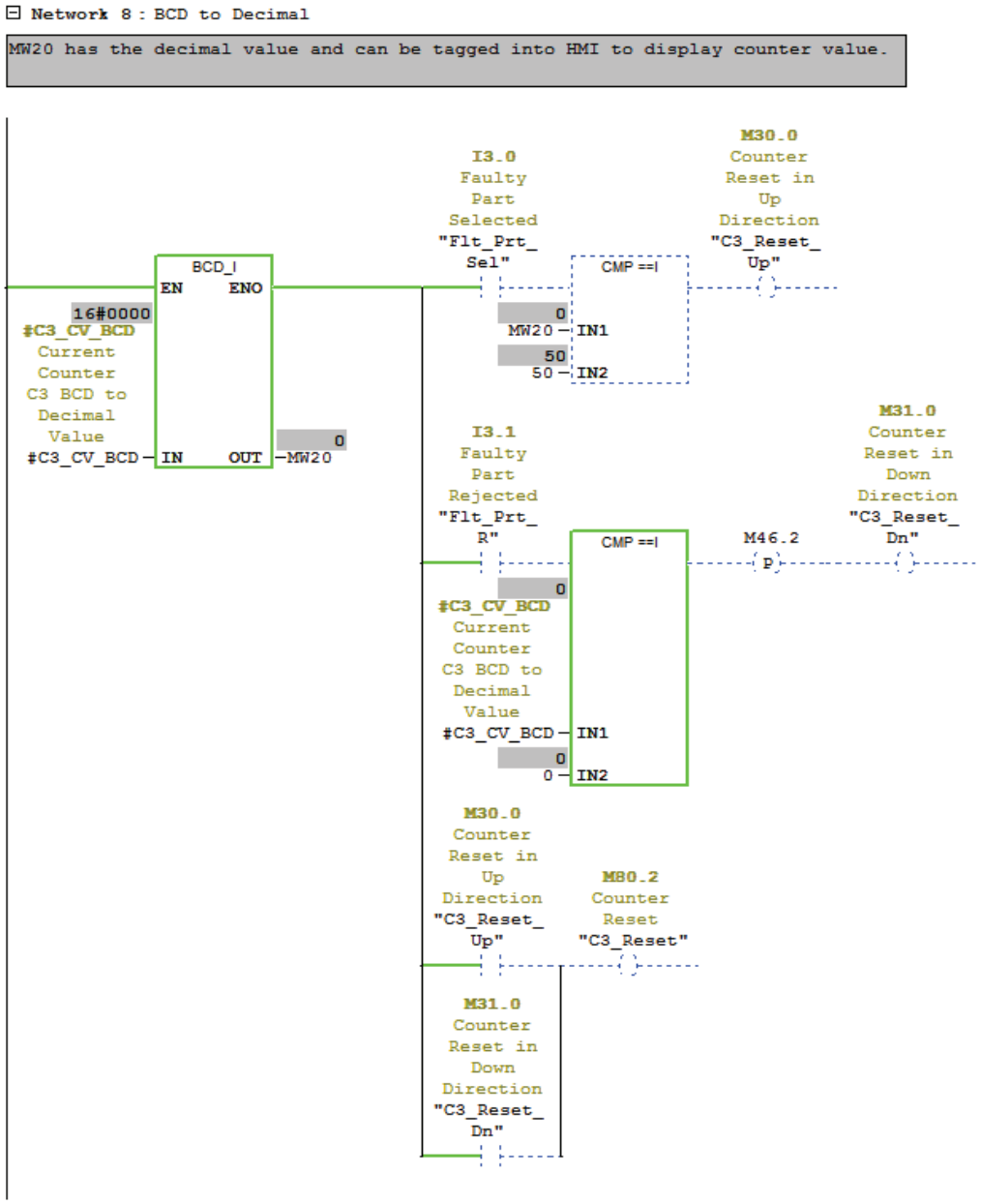
Now the drop drag procedure is same as explained for Up counter and you can follow diag#1 to diag#4 explained above where Up Down counter is shown too named C3.

The counter is incremented by one if the signal state at input CU reads 1. The counter is decremented by one if the signal state at input CD reads 1 and the value of the counter is greater than 0. If there is value 1 read at both count inputs, both instructions are executed and the count value remains unchanged, the counter is reset if there is a 1 at input R. Following diag#14 and diag#15 shows logic of Up Down counter.

Daig#14



Daig#15



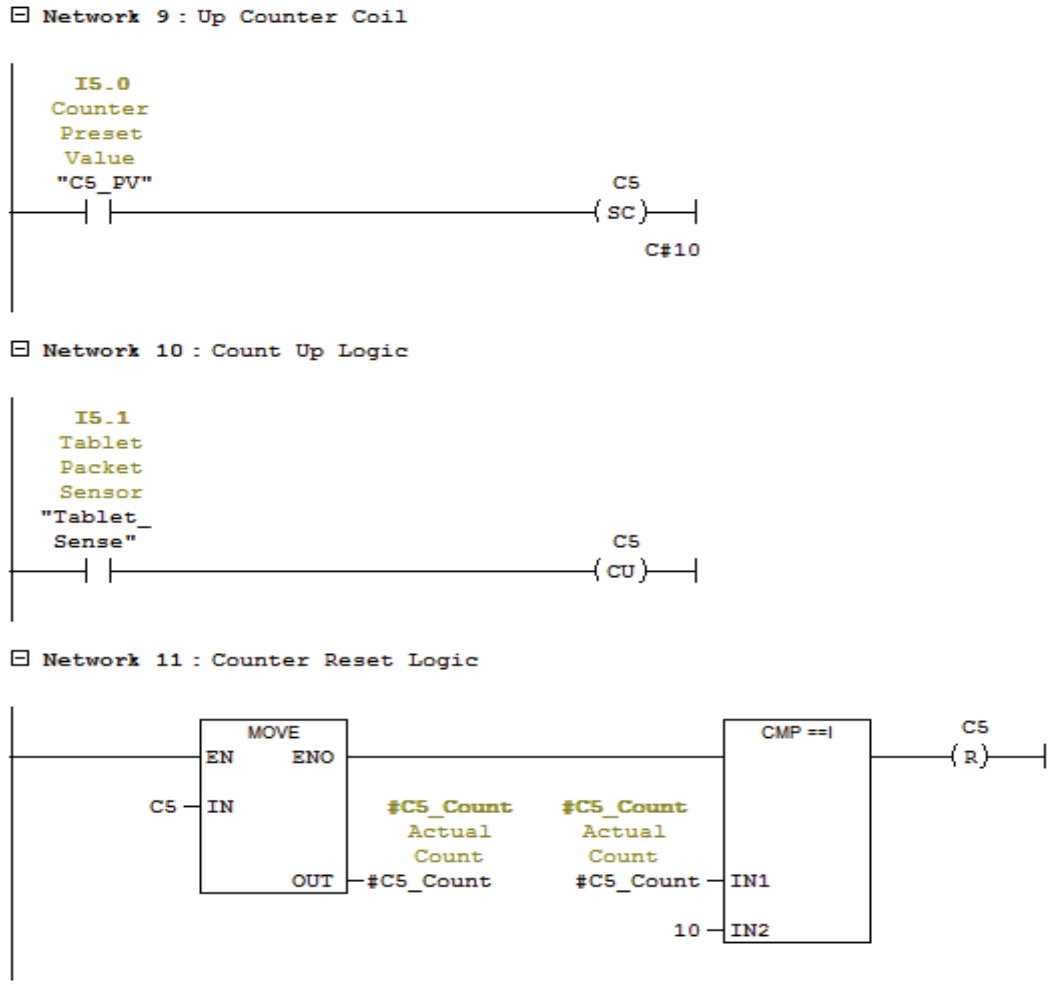
The logic counts Faulty part selected in up direction same in the way as explained in UP Counter and Faulty part rejected counts in down direction as explained with down counter example.

Here when Up count is reached internal memory M80.2 (C3_Reset) resets count to zero by reading 1 value from internal memory M30.0 (Reset_Up) and in case of down operation M80.2 (C3_Reset) reset count to zero by reading 1 value from internal memory M30.1(Reset_Down)!!

UP Counter Coil

This Up counter in coil form as shown below diag#16, it is dragged and dropped similarly as shown for above counters, can refer diag#3 above.

Daig#16



Here as shown in above diag#15 coil **SC** has two parameters

1) Counter number, here it is C5 and

2) Counter preset value, here it is 10, when real world input I5.0 goes high counter C5 is loaded by 10.

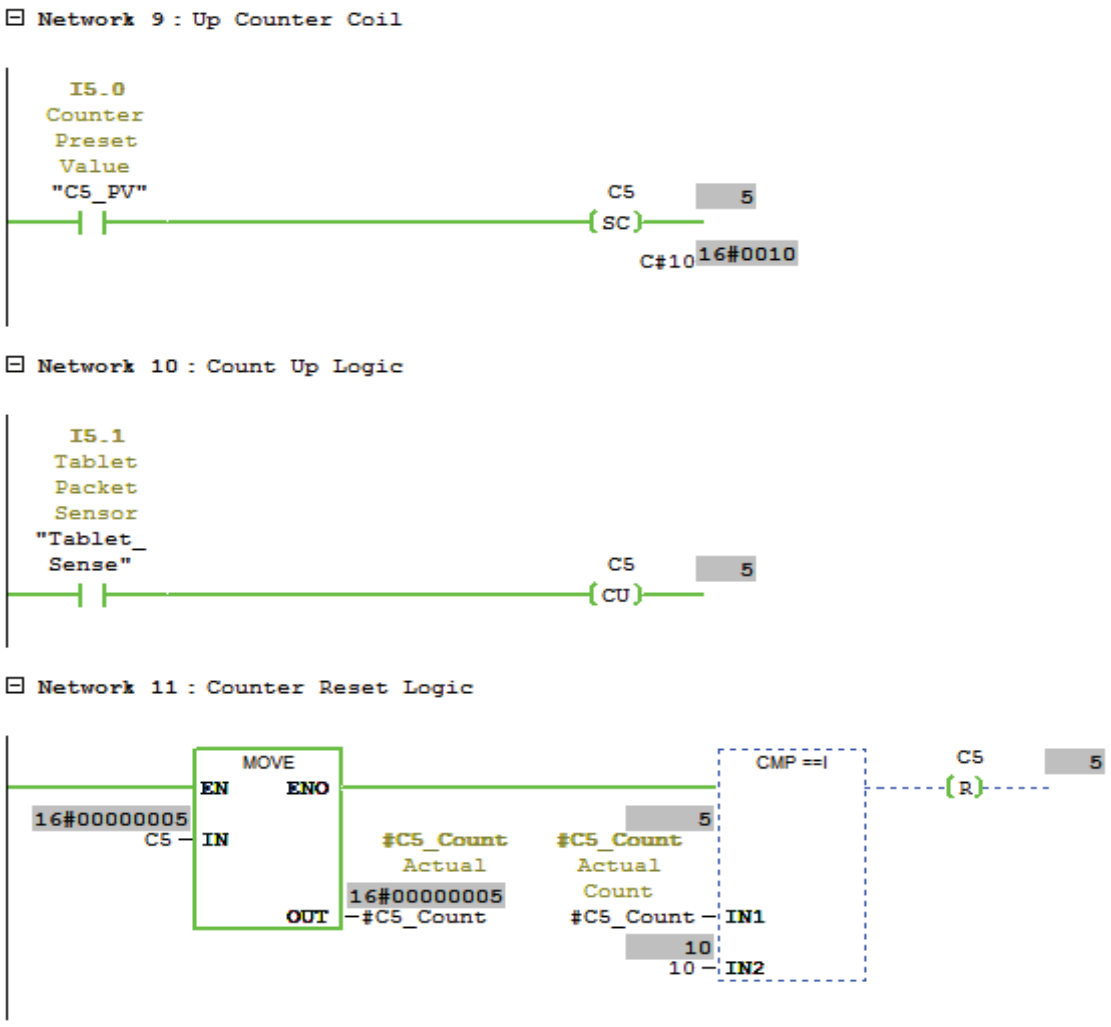
Coil **CU** acts as up counter which increment when real world input I5.1 goes high i.e. senses Tablet packet i.e. any preceding real world input or memory bit before coil exist and senses high state.

The counter is reset here when C5 reaches desired count 10. C5 value is moved into local memory (C5_Count) as C5 is not accepted by comparator block as it needs internal or local memory or direct value. The actual count is in C5.

Here suppose 10 tablet packets are to be counted in each batch then counter has to be reset to zero after count 10 is reached, which means one batch is completed, which fills one box of 10 tablet packets.

Now suppose 5 tablet packets are sensed the C5 has value 5 as shown in below diag#17.

Daig#17

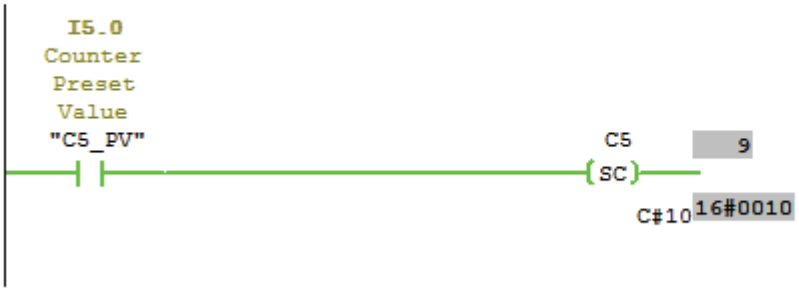


Now it counts till 10 and automatically resets to zero after 10 tablets are sensed.

Now C5 has count 9 as shown in diag#18

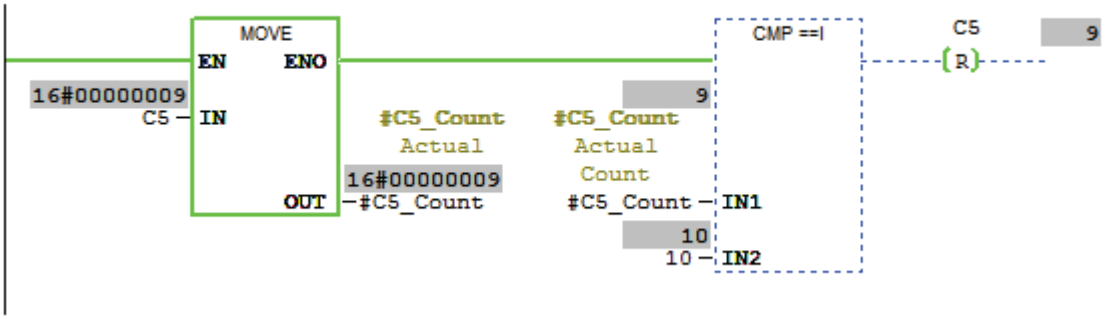
Daig#18

Network 9 : Up Counter Coil



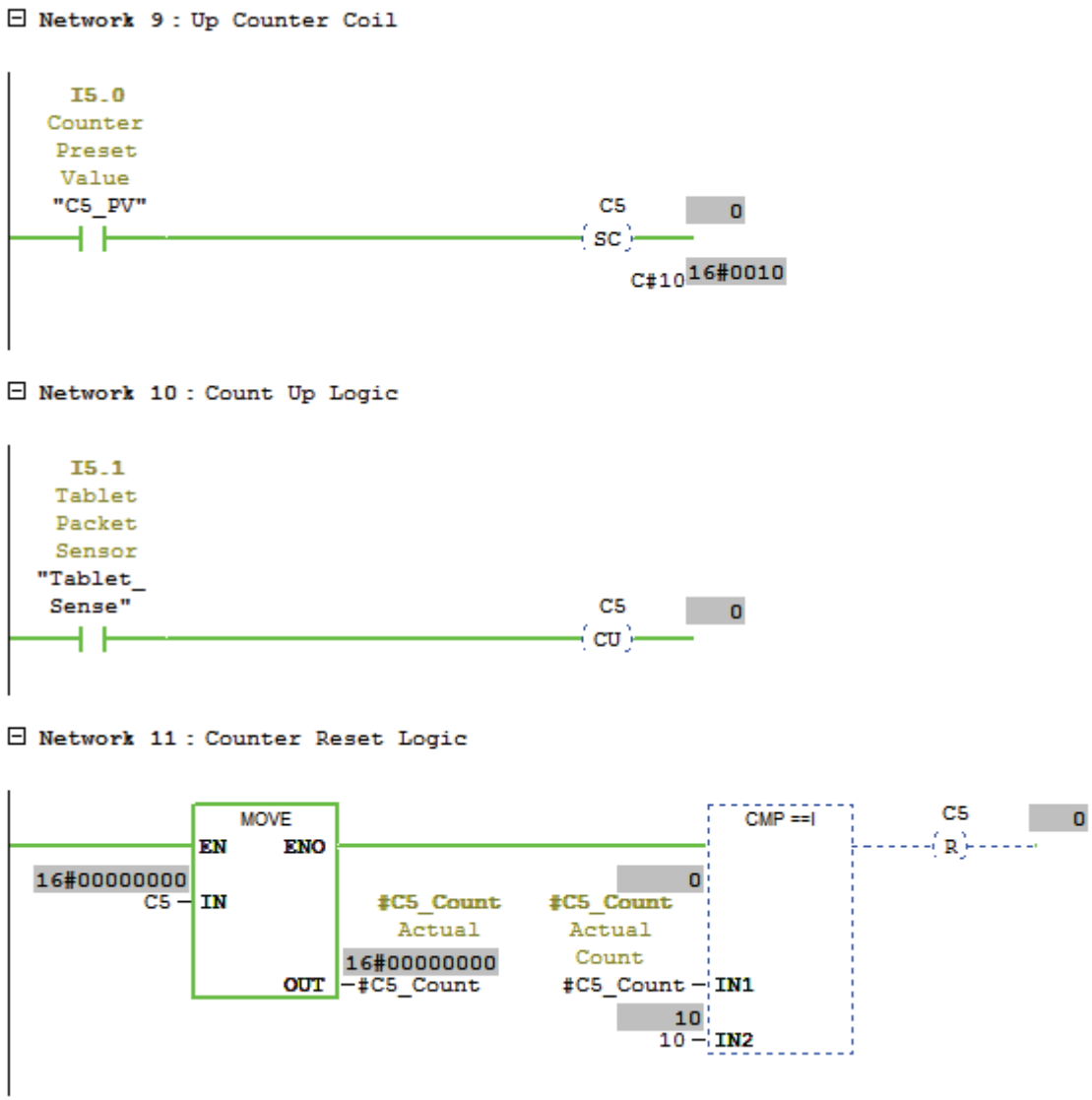
Network 10 : Count Up Logic

Network 11 : Counter Reset Logic



Now as 10th tablet pack is sensed counter C5 is automatically reset to zero as shown in diag#19

Daig#19



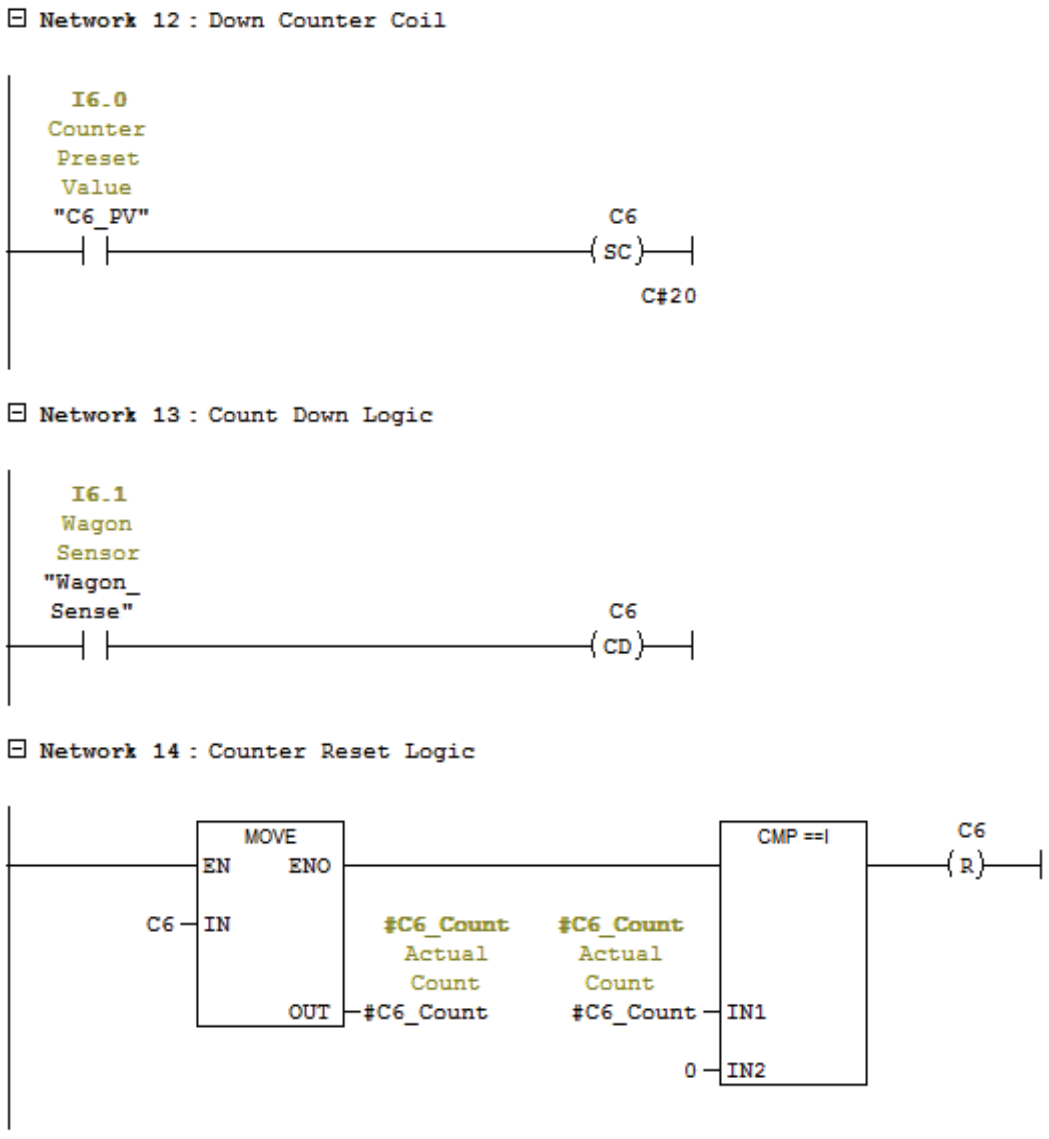
Again the counter is preset with value by turning real world input I5.0 ON, here after count 10, C5 was reloaded by 10 as I5.0 real world input was kept high throughout logic.

Note: In real practice it may happen that preset value has to be loaded manually from HMI after every batch and not kept high throughout logic always or depending upon requirement of process theory/ philosophy. In this way UP counter works in coil form!!

Down Counter Coil

This counter also looks like up counter in coil form as shown below diag#20 except coil CD in place of coil CU, it is dragged and dropped similarly as shown for above counters, can refer diag#16 above.

Daig#20



The operation of Down counter coil is in reverse way of Up counter coil and logical arrangement is same as Up counter coil as shown in diag#20. Here the counter counts in downward direction from its preset value to zero.

Here real world input I6.0 loads counter preset value and real world input I6.1 senses noof wagons in unloading station.

In above logic wagons used for loading coal on conveyor/truck carrying coal to thermal power station are counted. At coal station wagons are counted one by one, here for e.g. the desired count is 20 shown in above. When last wagon i.e #20 is counted counter is automatically resetted, similarly as explained in Up counter operation. In this way Down counter coil works.

Counter Operation in Structured Text

To understand counter operation in another programming language, provided below is a Structured Text example (IEC 61131-3). With structured text PAC programming, the life cycle cost of industrial automation control greatly increases compared to using just ladder logic. As the people maintaining and making minor modifications to control automation for the next 20-30 years are typically not computer programmers and IT people. They are electricians and industrial engineers.

Note below:

"=" is a compare operator

":=" is an assignment operator

"[:=]" is a non-retentive assignment operator

";" semicolon indicates end of current statement

"//" are followed by comment, like rung comment

"Part_Present" and "Operator_Reset_PB" are Boolean (one or zero) Tags (memory storage areas)

"PLCCount_7" is an INT (integer, whole number 1 to 32767) Tag (memory storage area)

Note: Tags are assigned data types in PAC software, so do not need to use "VAR" construct within Structured Text.

As you can see from above one typo or misunderstanding of similar looking operators, ***can cause damage to man or machine****. An issue far less likely to occur using a graphical programming language like ladder logic. Probably the most common error people make with structured text programming language is putting a "=" where they should have used a ":=". Possibly followed by using a semicolon where they should have used a colon.*

.

Below is PAC Structured Text example to make photo eye ('Part_Present') count to 38

```
// increment counter if photo eye sees part
IF Part_Present True THEN
PLCCount_7 [:=] PLCCount_7 +1 ;
// The brackets '[]'around ' := ' cause 'PLCCount_7' to reset to zero on PAC power up.

// if counter reaches 38, set 'counter done bit'
If PLCCount_7 = 38 THEN
PLCCount_7.DN True ;

// if counter reaches 998, set 'counter current value' to zero(0). In other world reset counter as
fail safe so counter never causes math overflow error.
If PLCCount_7 = 998 THEN
PLCCount_7 := 0 ;

// Or if operator presses reset button, reset counter to zero.
If Operator_Reset_PB True THEN
PLCCount_7 := 0 ;
```

**********THE END**********