**NIRMA UNIVERSITY**

**INSTITUE OF TECHNOLOGY**

**MECHANICAL ENGINEERING**

**Programmable logic controller (PLC)**

**LAB – 4**

**19BME134**

**Shrey Shah**

**Aim**

Programming of TwinCAT PLC using relay based ladder language.

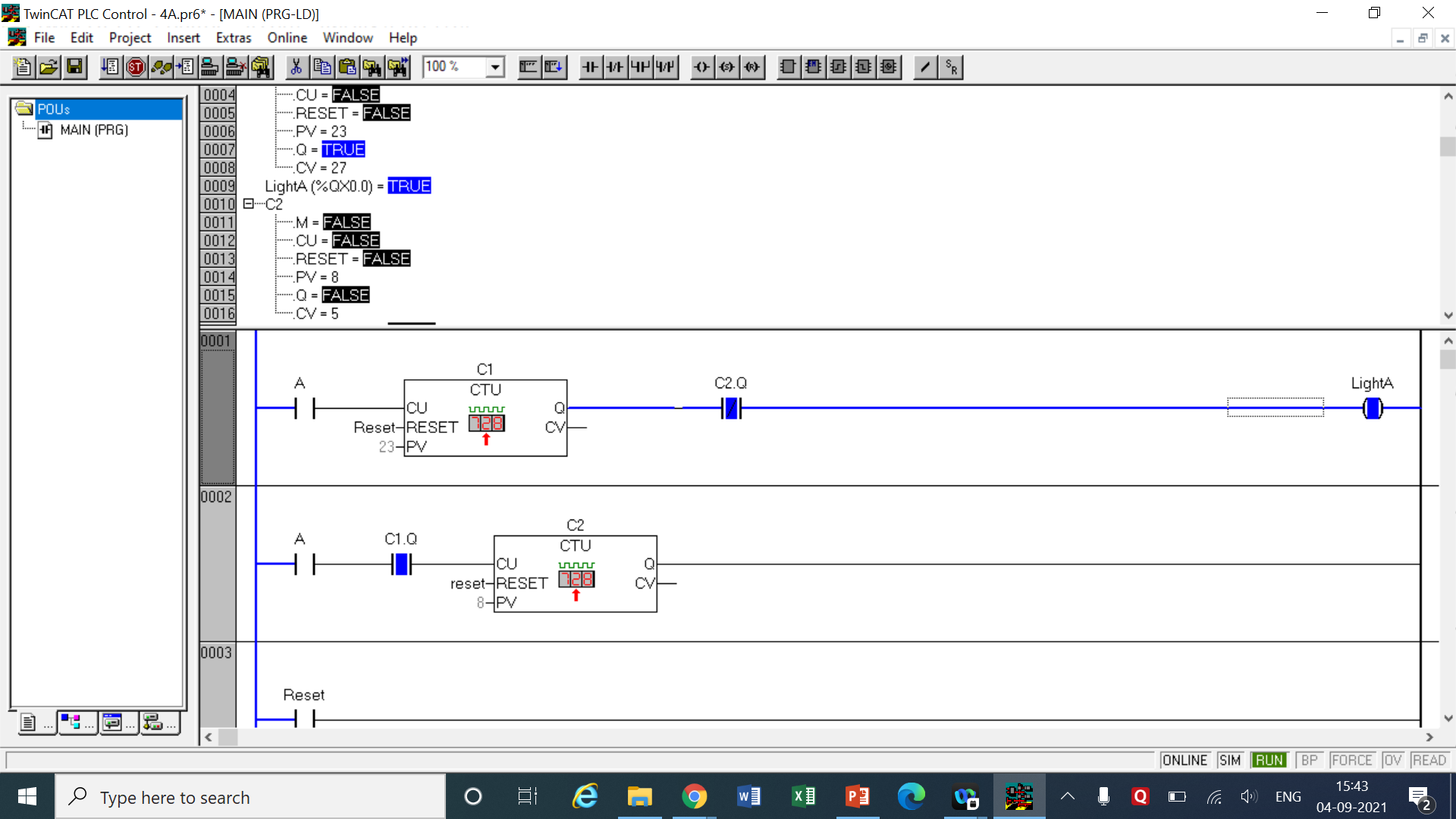
**Theory**

The programmable logic controller (PLC) is a solid state electronic device designed to replace electromechanical relays, timers, counters and sequences, by using a programmable memory for the internal storage of user oriented instructions for implementing specific functions such as logic sequencing timing, country and arithmetic control through digital or analog inputs and outputs, various types of machines or processes.

Major advantages of using PLC are as follows:

1. The PLC is a hardened industrial computer designed to withstand the harsh factory environment.
2. PLCs are reusable they contain a changeable program that eliminators extensive and component changes and that makes them cost effective
3. PLCs offer easy troubleshooting
4. PLCs feature easy installation and small size.
5. Increase productivity.
6. Ease of programming.
7. Ability to communicate with computer.

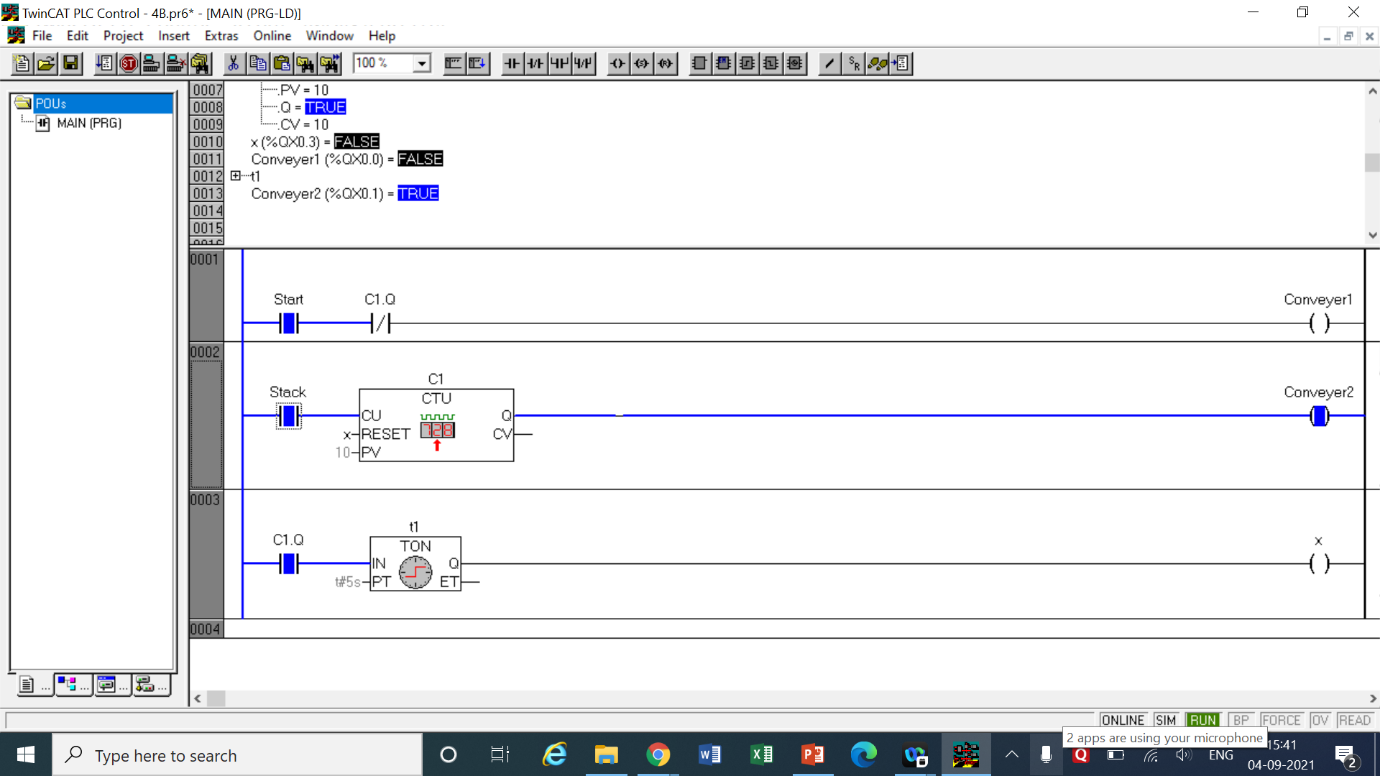
**Question 1**

1. An indicating light is to go on when a count reaches 23. The light is then to go off when count of 31 is reached. Assume: count pulse is coming from single external source and it is connected to single input terminal.

LOGIC -

* We have to make the lights go one for counts 24 to 31 only rest of the counts should be off.
* 2 Up Counters are taken named C1 and C2. We give the input pulse from the A switch to both the counters in separate networks as shown. In the 1st network the pre-set value is set as 23 which means that the circuit will not be complete until 23 pulses are taken.
* In 2nd network, after the switch A output of C1 is taken as normally open switch which means until 23 counts are reached the second network won’t be complete. When 23 counts are completed the second circuit closes and the counter 2 starts counting.
* We keep the pre-set value of C2 as (31-23) = 8 counts. In the 1st network the output of C2 taken as normally closed switch is connected to a light source after the counter connection.
* It means that while the 8 counts are happening the network one will be closed after 8 counts of C2 are done C2.Q opens breaking off the crcuit thus resetting the light (Off mode).
* A reset switch is taken in the 3rd network to reset the counts of both the counters after being pressed and released.

|  |  |  |  |
| --- | --- | --- | --- |
| INPUT | | OUTPUT | |
| Switch A | %ix0.0 | light | %qx0.0 |
| UP Counter C1 | Preset value = 23  RESET = Reset | C1.q | Output of C1 |
| Up Counter C1 | Preset value = 8  RESET = Reset | C2.q | Output of C2 |
| Reset | %ix0.1 | - | - |

1. When the start push button is pressed, conveyer 1 begins operating. After 10 plates have been stacked, conveyer 2 starts and conveyer 1 is stopped. After conveyer 2 has been operating for 5 seconds, it stops. The output coil for the timer resets timer and also provides a momentary pulse to automatically restart conveyer 1.

LOGIC

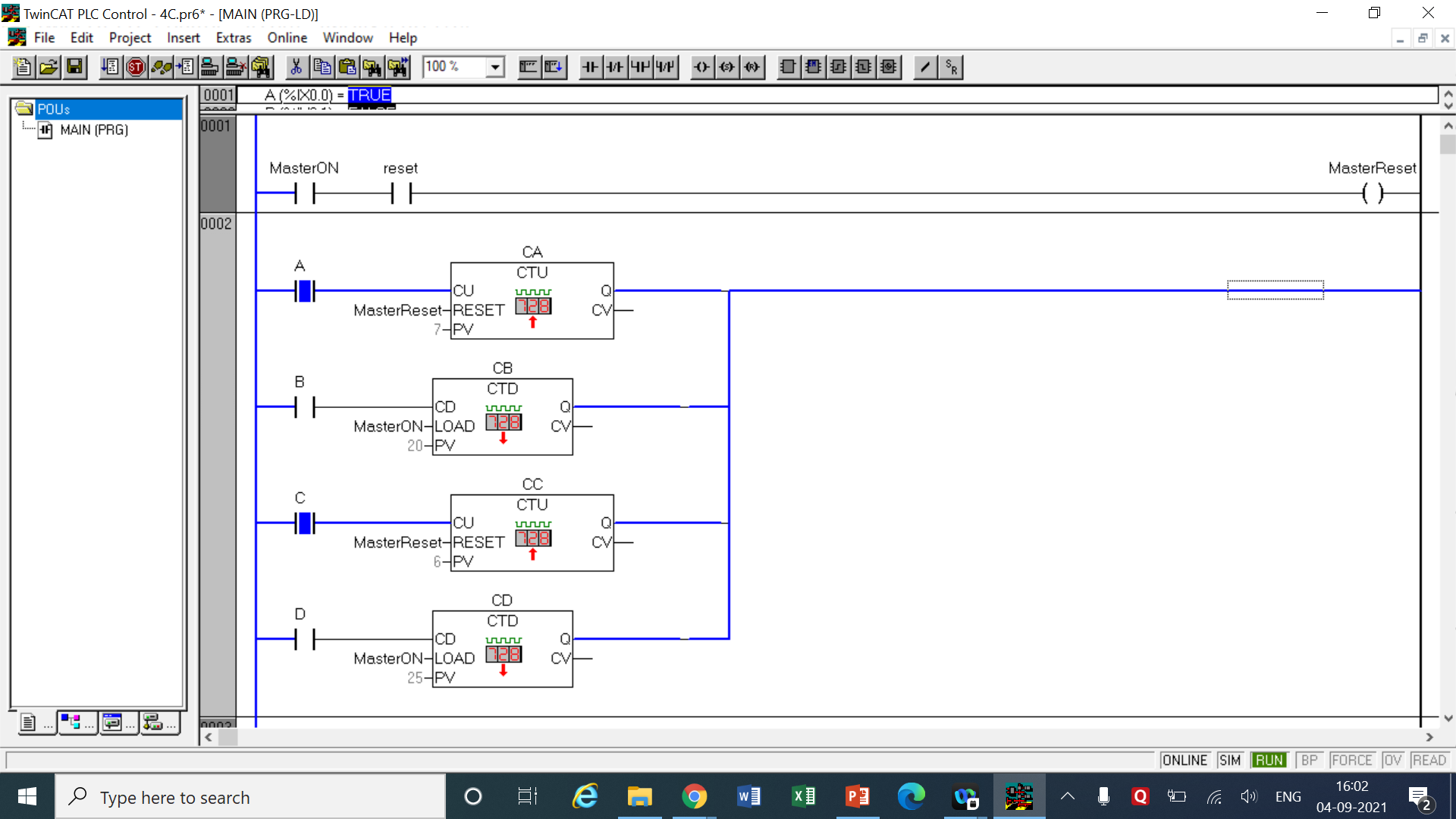
* 2 conveyers are needed to be operated – one based on counts and other based on time value.
* In the 1st network we take a start switch for the entire setup and a light indicating that conveyer 1 is working now.
* In the 2nd network we take a switch named ‘stack’ as input which will be pushed for every count of stacks. Following the switch is the UP Counter C1 with a present value of 10 counts. A light indicating that conveyer 2 is on is connected after this counter.
* When the 10 stacks are complete the circuit will be closed and the conveyer 2 will be on. To turn on conveyer 1 we add a normally closed switch of output of C1 after start switch in 1st network.
* When C1.Q is on conveyer 1 will be off and conveyer 2 will be on.
* In the third network we take the output of C1 and connect it with a On timer t1 of delay 5 seconds.
* When 5 seconds are over t1 will complete the circuit. We attach a output (light) ‘x’ in 3rd network which when turn on will reset the counter, hence also the timer.
* This will result in conveyer 2 being off and conveyer 1 being turned on again. This will continue endless times until the start switch is turned off

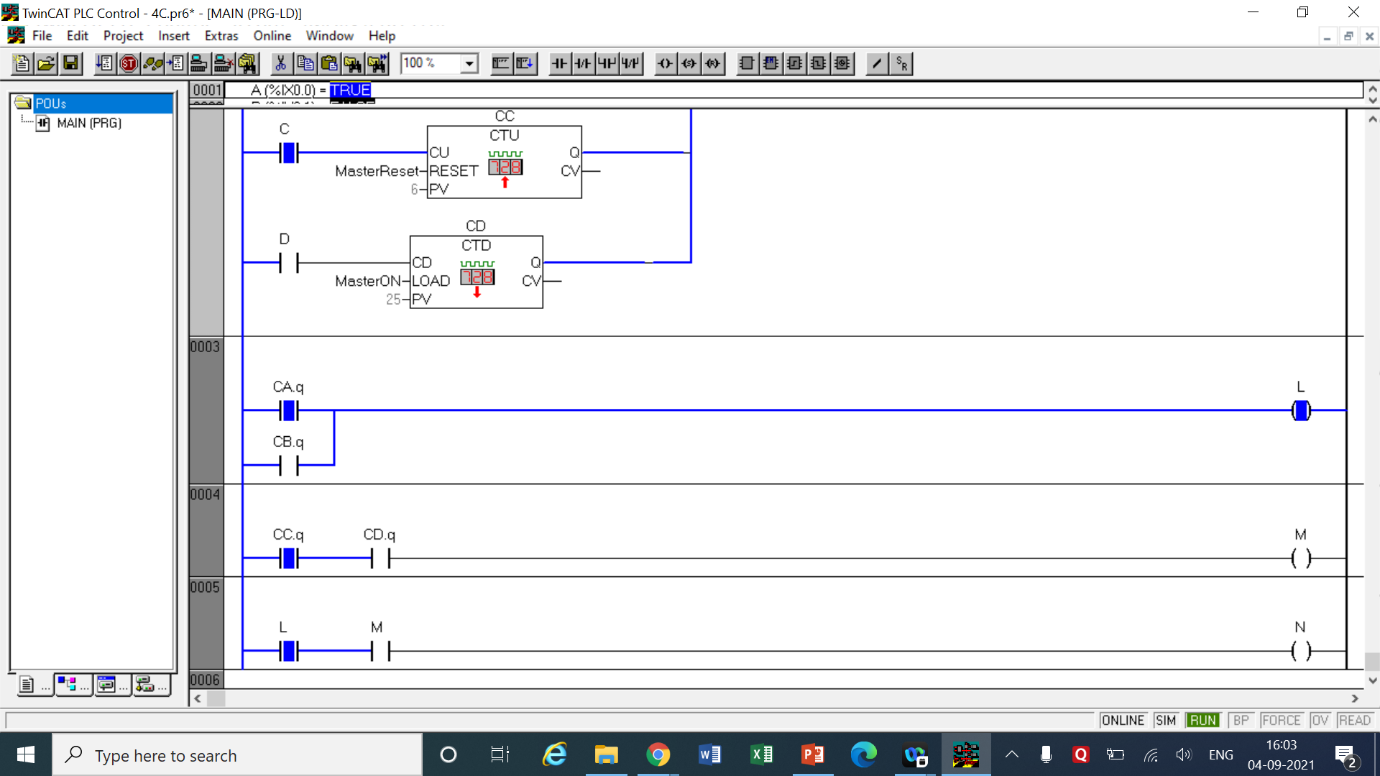
|  |  |  |  |
| --- | --- | --- | --- |
| INPUT | | OUTPUT | |
| Start | %ix0.0 | light | %qx0.0 |
| Stack | %ix0.1 | x | %qx0.3 |
| Up Counter C1 | Preset value = 10  RESET = x | C1.q | Output of C1 |
| On Timer t1 | Delay = 5 sec | t1.q | Output of t1 |

1. A process has four inputs which are connected to four PLC counters:

* IN021 Up counter
* IN022 Down counter (Preset 20)
* IN023 UP counter
* IN024 Down counter (Preset 25)

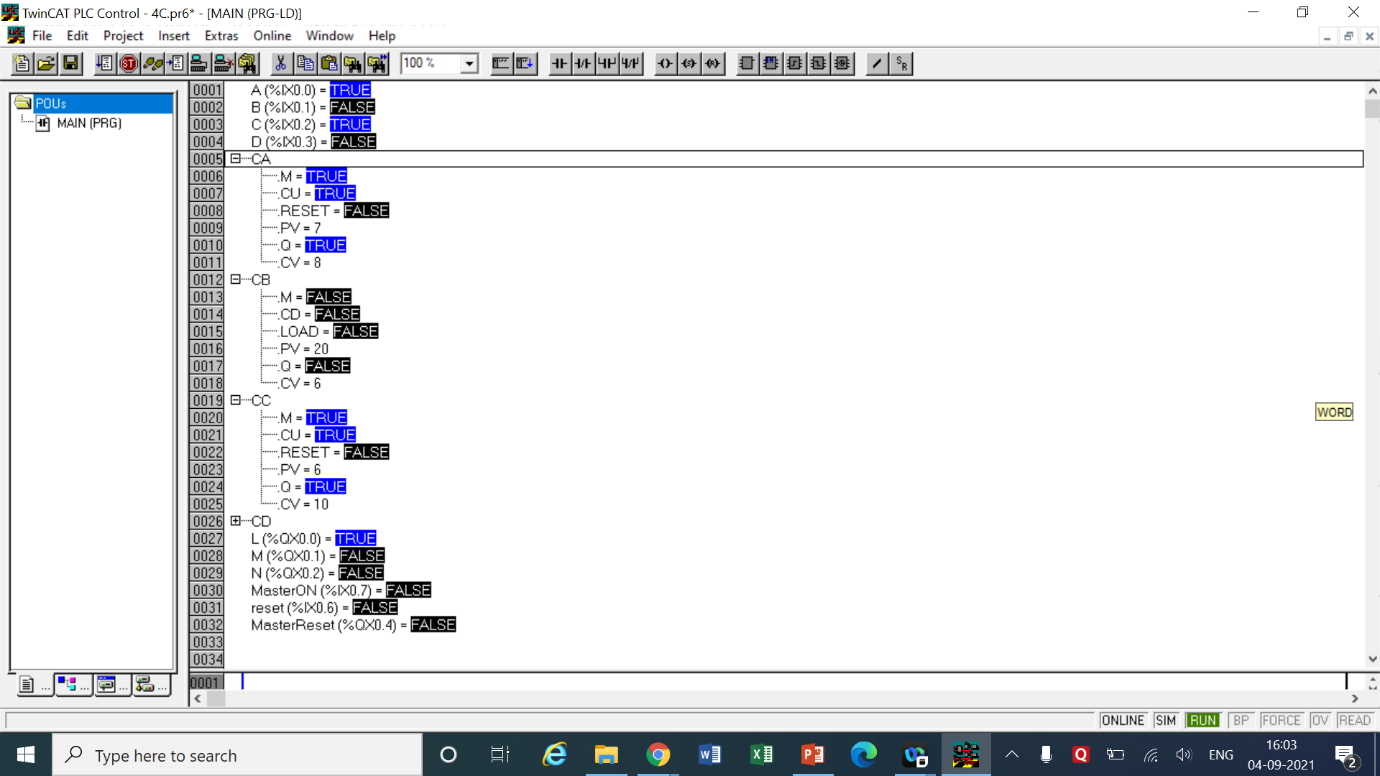
When A reaches 7 or B reaches 0, output L is to go on. When C reaches 6 and D reaches 0, output M is to go on. When both L and M are on, output N, should be turned on.



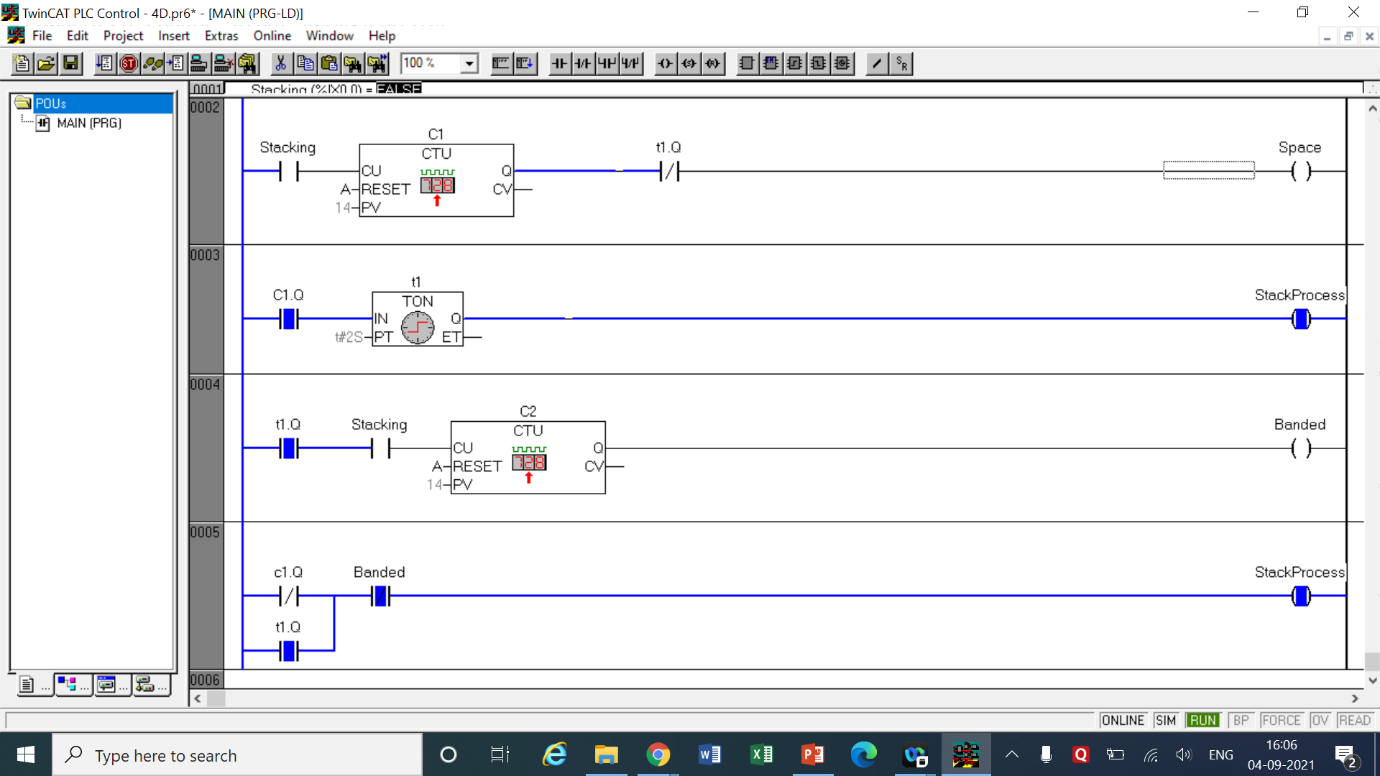


LOGIC

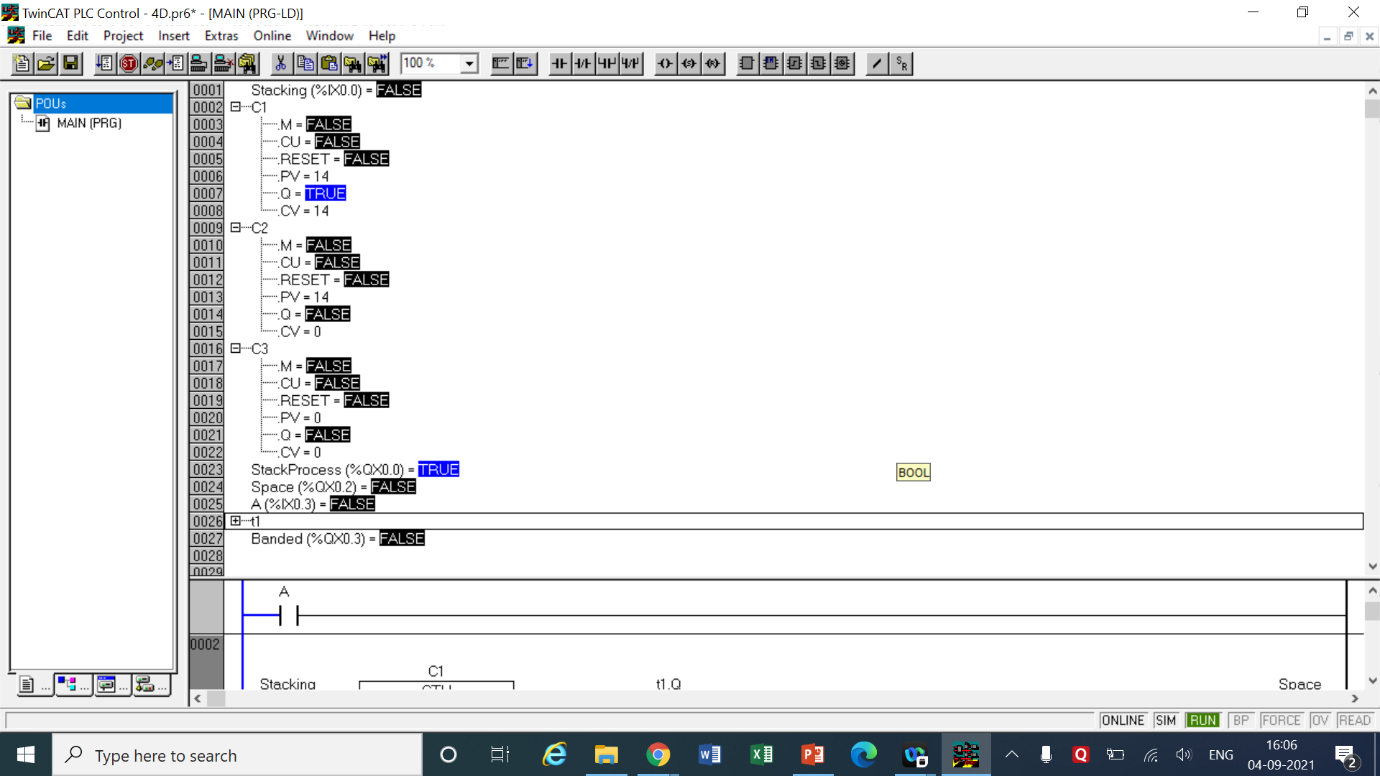
* In accordance with the question 4 counters are taken. Out of those 4 counters CA and CC are UP counters with preset values as 6 and 7 respectively.
* The counters CB and CD are DOWN Counters with preset values as 20 and 25 respectively. All counters are operated by their respective switches A, B, C and D all in the same 2nd network.
* The light L should be turned on when either of the counters CA or CB are done so the output of those counters are kept in parallel in network 3.
* The light M should be turned on when both the counters CC and CD are done so the output of those are kept in series in 4th network.
* When both L and M are turned on the light N should be turned on, hence L and M are placed in series as switches in 5th network connected to N.
* Now in the 1st network we take 2 switches, a MasterON switch is used which is also used to load the DOWN counters. A reset switch is used in series.
* When both MasterON and reset are pressed the MasterReset light turns on resetting the UP counters too along with reloading the DOWN counters, henceforth resetting the entire circuit in the process.



|  |  |  |  |
| --- | --- | --- | --- |
| INPUT | | OUTPUT | |
| Switch A | %ix0.0 | Light ‘L’ | %qx0.0 |
| Switch B | %ix0.1 | Light ‘M’ | %qx0.1 |
| Switch C | %ix0.2 | Light ‘N’ | %qx0.2 |
| Switch D | %ix0.3 | MasterReset | %qx0.4 |
| Up Counter CA | Preset value = 7  RESET = MasterReset | CA.q | Output of CA |
| DOWN Counter CB | Preset value = 20  LOAD = MasterOn | CB.q | Output of CB |
| Up Counter CC | Preset value = 6  RESET = MasterReset | CC.q | Output of CC |
| DOWN Counter CD | Preset value = 25  LOAD = MasterOn | CD.q | Output of CD |
| reset | %ix0.6 | - | - |
| MasterOn | %ix0.7 | - | - |

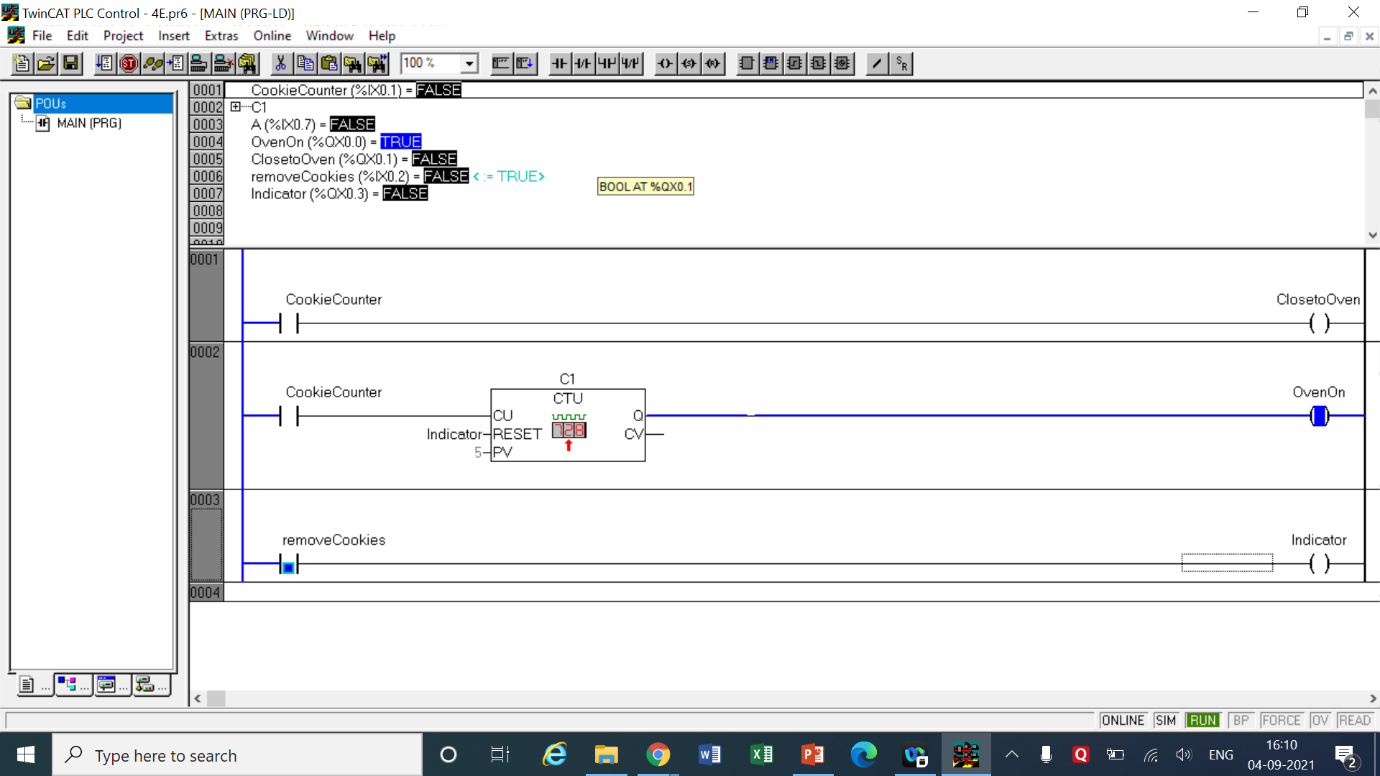
1. A stacking and banding system (S) requires a space to be inserted (I) in a stack of panels after 14 sheets are stacked. After 14 more (28 total), the stack is to be banded (B), Add sensors and assumed output devices as required.

LOGIC

* In this question we need stacks of 14 with a space inserted between them. For the space I used a space of time 2 seconds which means that there will be no stacking for 2 seconds of space interval.
* As always a stacking switch is connected to the UP Counter C1 in the 2nd network with a preset value of 14 which counts the amount stacked. The end of this network has the ‘space’ light to indicate a space is being inserted.
* In the 3rd network we take the output of C1 and connect it with an On Timer t1 of 2 seconds which is connected to the StackProcess Light. The output of t1 is connected after C1 as a normally closed switch meaning when 2 seconds are over, the circuit in 2nd network is cut off turning off the space light.
* In the 4th network the output of t1 is taken as a NO switch in series with the Stacking switch. The Stacking switch is in turn connected with a UP counter C2 with preset value 14 again. After the 14 more stacks are done it is then banded and hence the banded light turns on.
* In the last and 5th network we take output of C1 as NC and output of t1 as NO in parallel which is connected in series with banded light as a switch which is NC. The output of this network is the StackingProcess light.
* This means that the stacking process will be on all the time except when C1 is done and space is being inserted which breaks the circuit turning off the StackingProcess light and inserting space.
* ****A switch A is placed in the 1st network to reset the banding process along with the counters which will reset and start stacking again.

|  |  |  |  |
| --- | --- | --- | --- |
| INPUT | | OUTPUT | |
| A | %ix0.3 | Space | %qx0.2 |
| Stacking | %ix0.0 | StackingProcess | %qx0.0 |
| Up Counter C1 | Preset value = 14  RESET = A | C1.q | Output of C1 |
| On Timer t1 | Delay = 2 sec | t1.q | Output of t1 |
| Up Counter C2 | Preset value = 14  RESET = A | C2.q | Output of C2 |
| - | - | Banded | %qx0.3 |

1. Place five trays of cookies in Oven O2 and switch it ON. When a tray is brought close to the oven indicator I1 should switch ON and OFF. When the cookies are taken out of the oven, indicator I2 should switch ON. Reset the counter when I2 is switched ON. Write the ladder logic for this program.

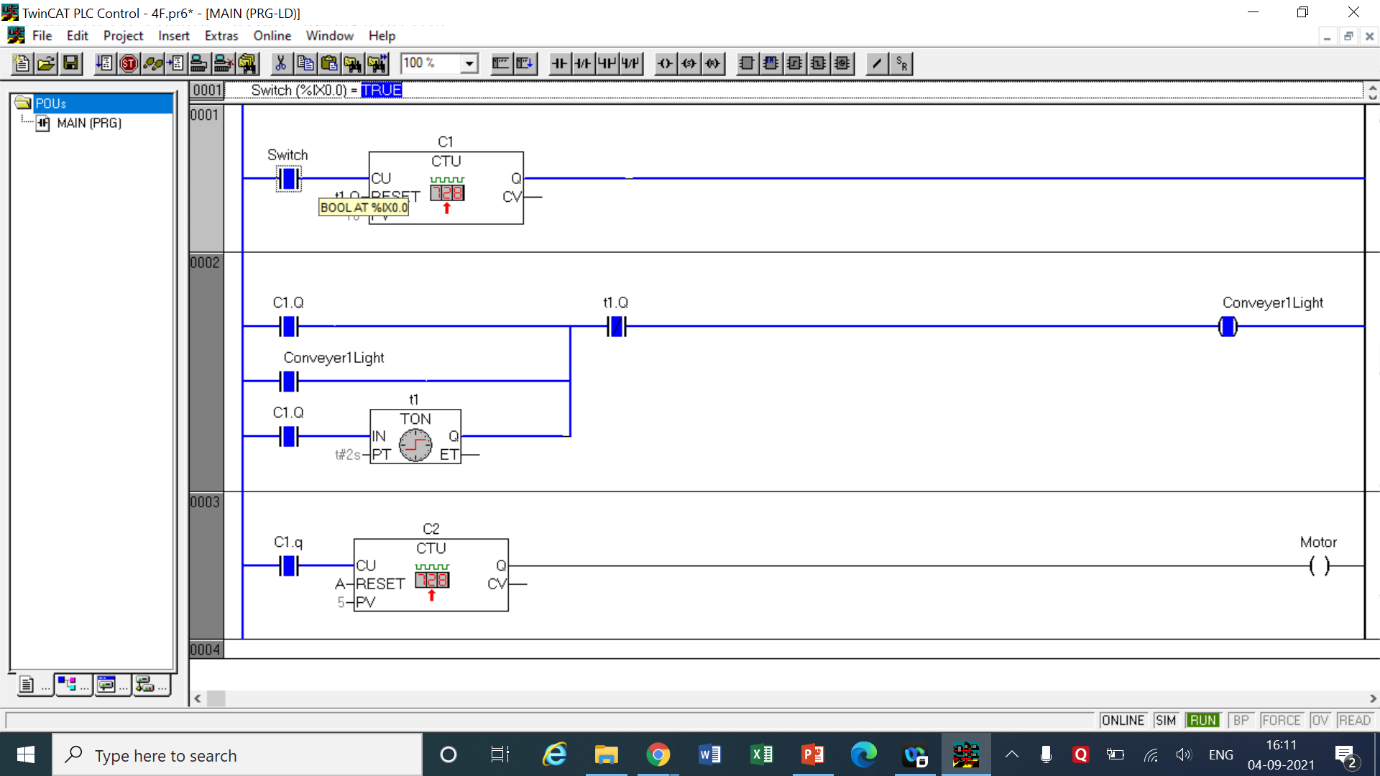
****

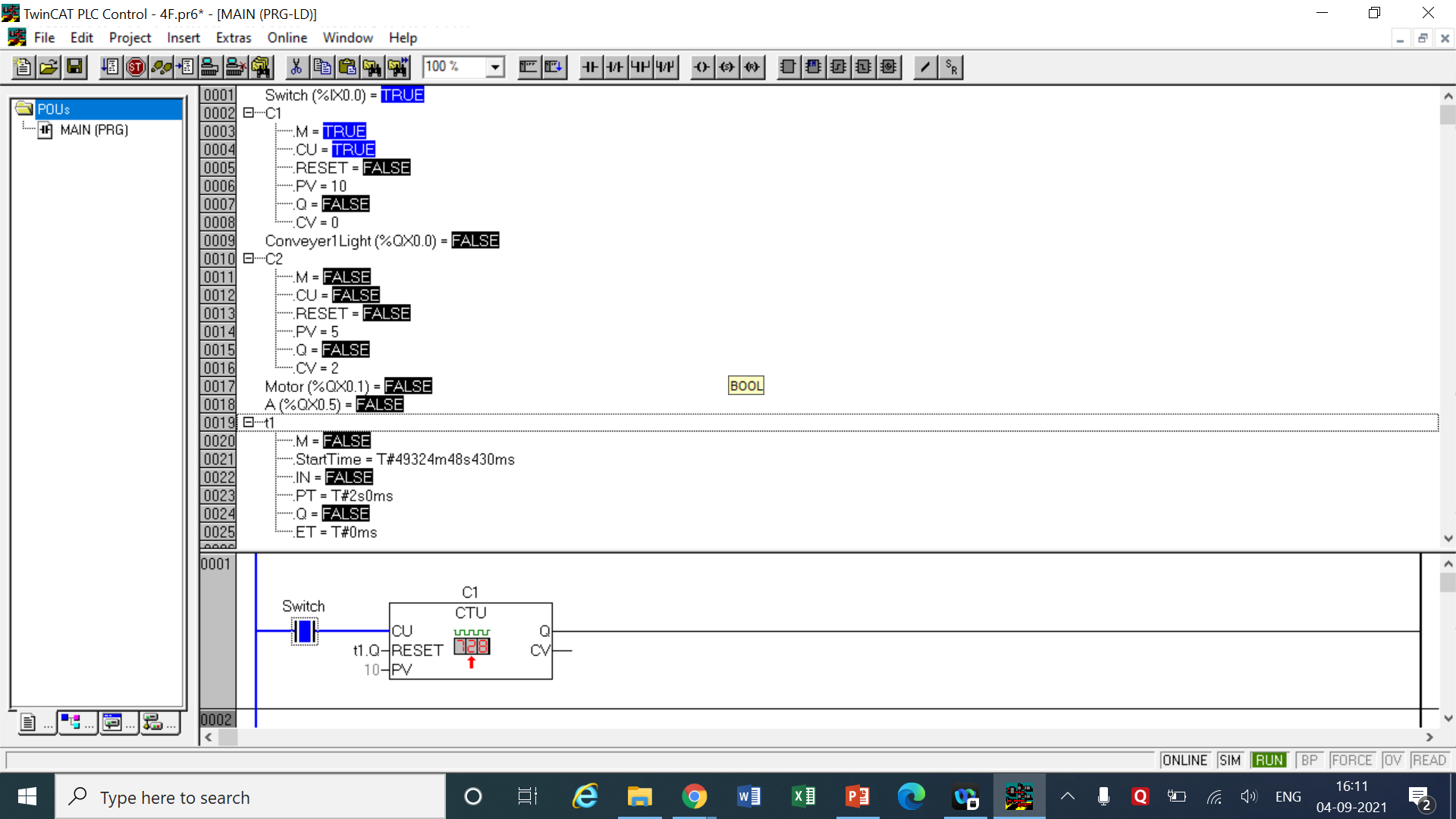
LOGIC

* This logic is pretty simple. We have to place 5 cookie trays before turning the over on so we take a switch ‘CookieCunter’ to take input of number of trays and a UP Counter C1 to count the number of trays kept in the oven.
* C1 has a preset value of 5 counts. To remove the cookies we use ‘removeCookies’ switch which turns the indicator light on showing that at this time the cookies are being removed.
* The indicator light is taken as a reset input of the counter C1 so when cookies are being removed the oven will automatically turn off.
* The question requires to turn on a light whenever the cookies are brought close to the oven so in the first network we again place the CookieCounter switch which is connected to the ‘ClosetoOven’ light. Whenever each tray is placed the close to over light will be on indicating the tray is near to the oven.

|  |  |  |  |
| --- | --- | --- | --- |
| INPUT | | OUTPUT | |
| CookieCounter | %ix0.1 | ClosetoCookie | %qx0.1 |
| removeCookies | %ix0.2 | OvenOn | %qx0.2 |
| UP Counter C1 | Preset = 5  RESET = Indicator | Indicator | %qx0.3 |

1. Write a program that switches ON a light L1 if conveyor C1 runs 10 times. If L1 is switched ON five times, Motor M2 should switch ON.

****

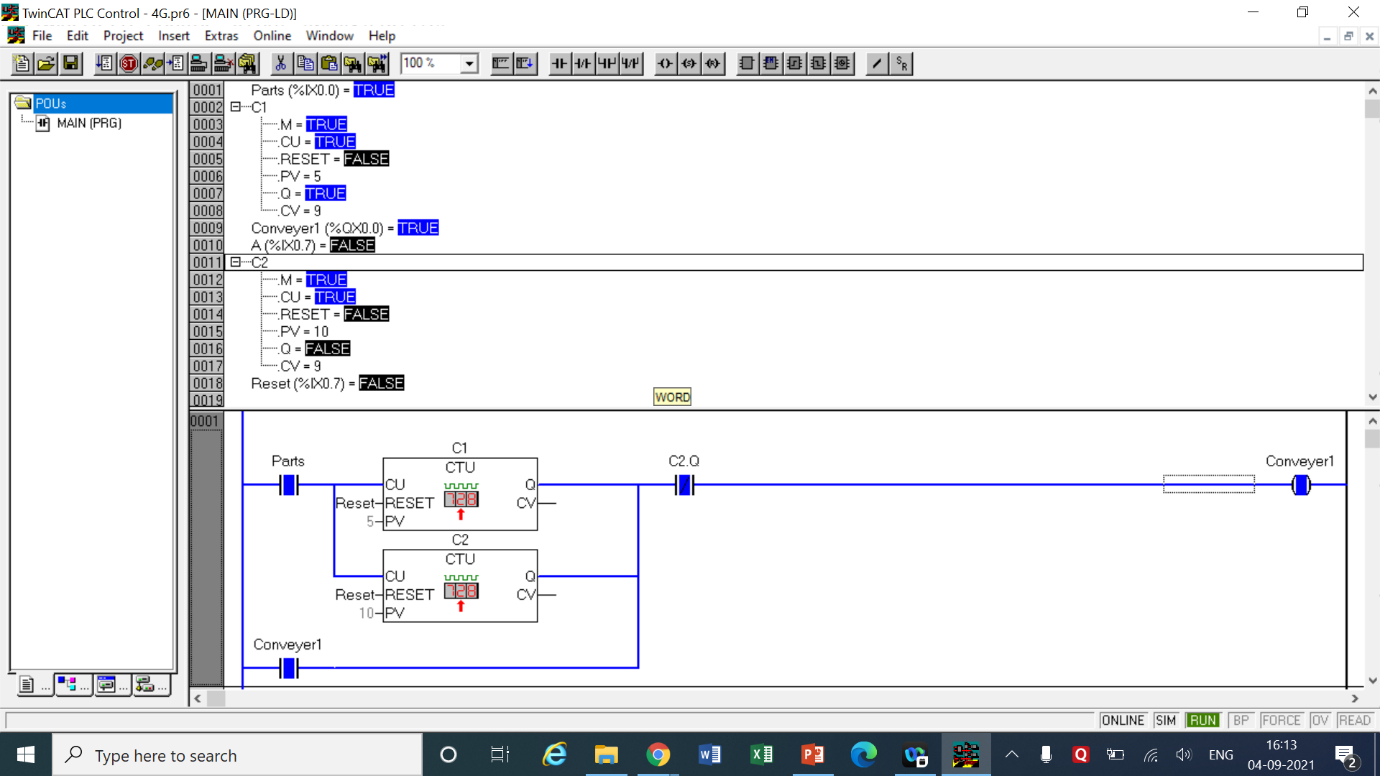
****

LOGIC

* This logic is also straight forward. We simply take 2 UP counters C1 and C2 of Preset value 10 and 5 respectively.
* Every time the conveyer runs 10 times then the conveyer light will be turned on.
* When the conveyer light is turned on 5 times then the motor will start running.
* The circuit in the 2nd network is just to indicate that the conveyer 1 has been run 10 tines. It uses a combination of simple NO and NC switches of outputs of C1 and C2 using a On timer t1 with a delay time of 2 seconds.
* It means that when C1 has finished counting 10 times then the counts of C1 will be reset and the conveyer light will turn on for 2 seconds. After the countdown of 2 seconds are over the output of t1 will break the circuit turning off the conveyer light.
* The output of C1 is also connected to the counter C2 with preset value as 5 counts so that when 10 counts are over 5 timer the motor which is connected to it will turn on too.

|  |  |  |  |
| --- | --- | --- | --- |
| INPUT | | OUTPUT | |
| Switch | %ix0.0 | Conveyer1Light | %qx0.0 |
| UP Counter C1 | Preset = 10  RESET = t1.Q | C1.Q | Output of C1 |
| UP Counter C2 | Preset = 5  RESET = Switch | C2.Q | Output of C2 |
| On timer t1 | Delay = 2 sec | t1.Q | Output of t1 |
| - | - | Motor | %qx0.1 |

1. Write a program that runs the conveyor C2 when five parts are placed on it. When 10 parts are placed on it, the conveyor should stop.

****

LOGIC

* This is also a simple program in this 2 UP Counters C1 and C2 with preset values as 5 and 10 are placed in parallel respectively meaning whenever a pulse is received both of them will receive at the same time and the same pulse.
* A switch Parts has been used to make the counts for the counters. According to the arrangement when C1 finishes its counting then the circuit will be complete and the conveyer1 will be turned on.
* The conveyer light is taken as a switch from input again so that it remains on until the circuit is broken.
* Now C1 becomes pretty much useless but the count is still going on in C2 so when the count reaches 10 and C2 finishes its countdown the conveyer1 should be turned off.
* For that we place a NC switch of output of C2 after the parallel connection of the counters so that when C2 is done the circuit will be broken and the light will be turned off.
* In the 2nd network a simple reset switch is placed which after pressing will reset C1 and C2 and the process can be repeated again and again.

|  |  |  |  |
| --- | --- | --- | --- |
| INPUT | | OUTPUT | |
| Parts | %ix0.0 | Conveyer1Light | %qx0.0 |
| UP Counter C1 | Preset = 5  RESET = Reset | C1.Q | Output of C1 |
| UP Counter C2 | Preset = 10  RESET = Reset | C2.Q | Output of C2 |
| Reset | %ix0.1 | - | - |

COMMENTS –

* The counters in most of the questions are denoted by the letter ‘C’ followed by the number of the counter or the letter.
* In case of UP and DOWN timers their names have been specified while mentioning the use of counters. Cn.q represents the output of those counters which may be taken as NO or NC switches.
* Each question has a table of inputs and outputs which specifies which I/Os have been taken along with its addresses.
* All the timers are generally denoted by the symbol ‘tn’ where n represents the number of the timer.
* tn.q represents the output of the timer tn which can be both normally open or normally closed depending on how it is used based on the question’s requirements.

**CONCLUSION**

* The experiment thoroughly explained the use of both the UP and DOWN counters in real life. These counters can be used to perform orderly processes which come in steps. The counters can also be coupled with timers to give even wider range of applications. In case both Up and Down counting is required special updown counter (CTUD) is also available in the TwinCat software. PLCs with counters and timers in them have decisive results without much errors.