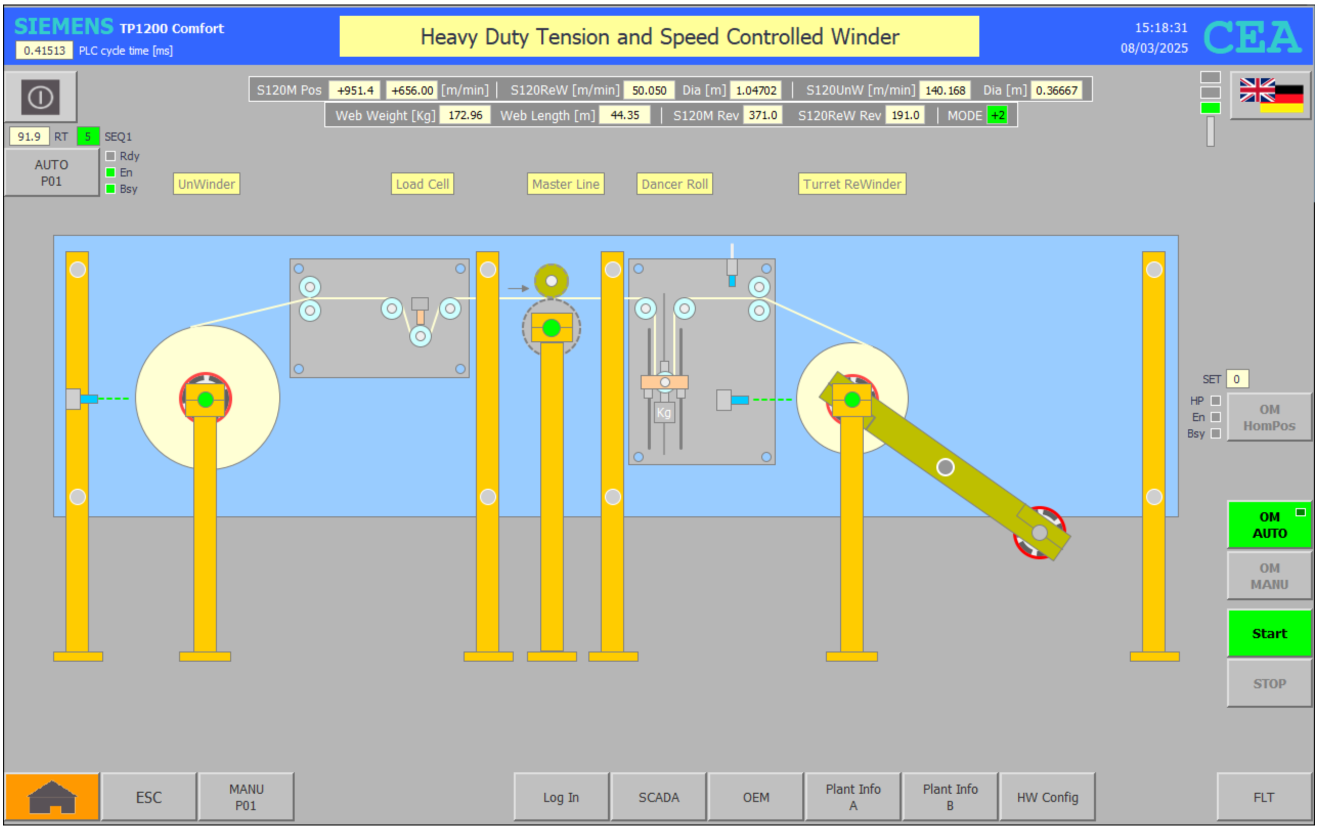
Heavy Duty Winder Speed and Tension Control



SIEMENS TP1200 Comfort

0.41513 PLC cycle time [ms]

Heavy Duty Tension and Speed Controlled Winder

15:18:31
08/03/2025

S120M Pos +951.4 +656.00 [m/min] | S120ReW [m/min] 50.050 Dia [m] 1.04702 | S120UnW [m/min] 140.168 Dia [m] 0.36667

Web Weight [Kg] 172.96 Web Length [m] 44.35 | S120M Rev 371.0 S120ReW Rev 191.0 | MODE +2

91.9 RT 5 SEQ1

□ Rdy
🟩 En
🟩 Bsy

SET 0

HP □
En □
Bsy □

Contents

1. **Revision**

Date:	Name:	Comment:
27.02.2025	mc	Texting basic objectives
28.02.2025	mc	Program view
01.03.2025	mc	Texting winder control calculating mode 0-4
02.03.2025	mc	Texting theory source specification
03.03.2025	mc	Texting PID dancer and load cell control
05.03.2025	mc	Current HMI. S120M ActVelo x 4 on HMI

2. Purpose of this document

Diameters are mainly responsible for the aspired tension and roll speed.
CEA describes in this document its elementary winder application along with the calculation of an Unwinder and a Rewinder roll diameter.
One of the key points is to allow simultaneous changes of the line velocity whilst running process.

3. Main pre-setting

- OEM Setting (Original Equipment Manufacturer)
- SCADA Setting (System Control And Data Acquisition)

4. Main objectives

- Speed calculation driven by the reference line velocity
- 5-modes roll diameter calculation
- Additive torque calculation
- Surface tension control
- Minimising speed oscillations
- Inertia calculation
- Weight calculation
- Rest length calculation
- Target winder diameter or length
- Avoiding wrinkling, telescoping and dished roll

5. Velocity handling

It enables speed adaption during production. The given Master Line velocity can be set to zero, stopped due to threshold detection, Estop, or HMI Stop during any process, and restarted without losing its measuring quality.

6. Unwinder diameter calculation

The Unwinder diameter is permanently calculated by using the same method. The roll diameter is scaled between core and full roll diameter using its rest web length. To keep web tension constant we continually multiply tension set point with diameter ratio of the Unwinder.

7. **Five principle modes of Rewinder diameter calculation**

- 0-Thickness addition
  The winder roll diameter is the result of adding web thickness after each roll revolution. Adding result is smoothed via a PT1-ramp, and speed at roll diameter is the result of Master Line velocity divided by roll perimeter. In order to prevent unnecessary calculation the roll diameter can be just as well calculated after a certain roll revolutions - suitable for thin web materials with low air inclusions.

- 1-Using Perimeter breakpoints
  This method observes the winder revolution and interpolates the slop needed for the roll speed calculation which is then Master Line velocity divide by winder perimeter. The perimeter slop is smoothed via a PT1-ramp – suitable for all web materials.

- 2-Division by roll diameter (linear ramped)
  This method observes the winder revolution and scales the diameter needed to calculate the winder speed. The scaled diameter lies between core and full roll diameter. Master Line velocity is then divided by winder roll perimeter. To prevent unstable slop the resolute of this division is linearly ramped - suitable for thin web materials

- 3-Division by roll diameter (partly linear and partly hyperbolic)
  This method resembles the previous calculation principle. Master Line velocity is divided by winder roll perimeter. To prevent speed oscillation the resolute of this division is then partly linear and partly hyperbolic ramped - suitable for thick web materials

- 4-Open loop
  In this mode the diameter of each roll is permanently measured by using an ultrasound sensor. For both winders the speed at roll diameter is the result of Master Line velocity divided by roll perimeter

8. **General disclaimer**

CEA will in no event be liable for any loss or damage including without limitation, indirect or consequential loss or damage to PLC data, or any damage whatsoever arising from culpable loss of life, bodily injury or damage to health arising out of, or in connection with the use of this documentation or PLC-program.

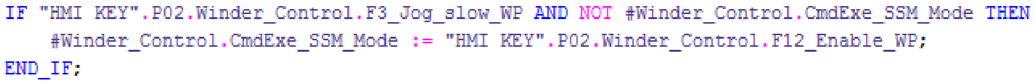
9. **Preamble**

- An unusual change has been taken place in the CEA Manual Mode Logic of this winder application. As a rule, Manual Mode has been conceptualized to operate assembly units during commissioning the plant regardless of any faults, unready units or of any process modules. Furthermore, it can be called during the Automatic Mode for any necessary correction and then back to continue the automatic process. In this case, Automatic Mode is being immediately stopped for all process modules.

- Never the less, CEA has decided to beak the rule in this Application, and let winders start the SSM process also from the Manual Mode without any concept limitations - the sequencer runtime monitoring is in this case disabled.

FB MAN P01 Manual Mode of process module number 1.

Network 1: Winder Control start stop



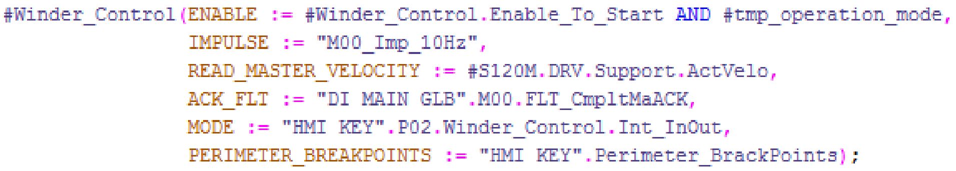
```
IF "HMI KEY".P02.Winder_Control.F3_Jog_slow_WP AND NOT #Winder_Control.CmdExe_SSM_Mode THEN
    #Winder_Control.CmdExe_SSM_Mode := "HMI KEY".P02.Winder_Control.F12_Enable_WP;
END_IF;
```

Setting #Winder_Control.CmdExe_SSM_Mode to TRUE affects the Step Switching Mechanism of the winder in FB SEQ1 P01

10. **The Way to Call the Winder Control**

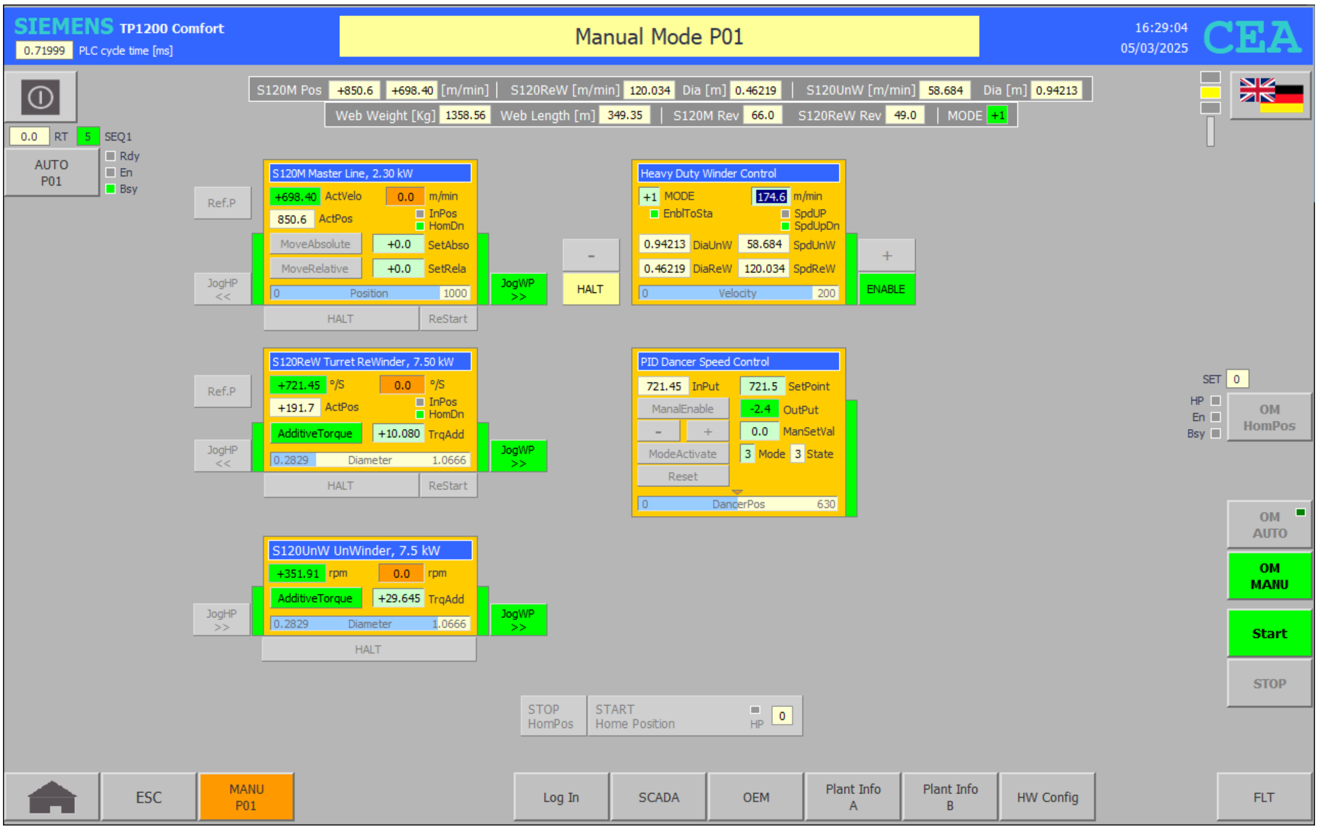
The winder main objectives are realized in one function block (FB) called Winder_Control. You can call this FB in OB30 or just as well in OB1. If you call the FB in OB30, you must set formal operand IMPULSE := TRUE.
To call the FB in OB1 assign IMPULSE := "M00_Imp_10Hz"

Network 11: Winder Control in cyclic program OB1



```
#Winder_Control(ENABLE := #Winder_Control.Enable_To_Start AND #tmp_operation_mode,
                IMPULSE := "M00_Imp_10Hz",
                READ_MASTER_VELOCITY := #S120M.DRV.Support.ActVelo,
                ACK_FLT := "DI MAIN GLB".M00.FLT_CmpltMaACK,
                MODE := "HMI KEY".P02.Winder_Control.Int_InOut,
                PERIMETER_BREAKPOINTS := "HMI KEY".Perimeter_BrackPoints);
```

A screenshot shows SSM in Manual Mode - here at step 3 speeding up.



SIEMENS TP1200 Comfort

0.71999 PLC cycle time [ms]

Manual Mode P01

16:29:04
05/03/2025

S120M Pos +850.6 +698.40 [m/min] | S120ReW [m/min] 120.034 Dia [m] 0.46219 | S120UnW [m/min] 58.684 Dia [m] 0.94213

Web Weight [Kg] 1358.56 Web Length [m] 349.35 | S120M Rev 66.0 S120ReW Rev 49.0 | MODE +1

0.0 RT 5 SEQ1

□ Rdy
□ En
■ Bsy

S120M Master Line, 2.30 kW

+698.40 ActVelo 0.0 m/min

850.6 ActPos □ InPos ■ HomDn

+0.0 SetAbso

+0.0 SetRela

0 Position 1000

Heavy Duty Winder Control

+1 MODE 174.6 m/min

■ EnblToSta □ SpdUP ■ SpdUpDn

0.94213 DiaUnW 58.684 SpdUnW

0.46219 DiaReW 120.034 SpdReW

0 Velocity 200

S120ReW Turret ReWinder, 7.50 kW

+721.45 °/S 0.0 °/S

+191.7 ActPos □ InPos ■ HomDn

+10.080 TrqAdd

0.2829 Diameter 1.0666

PID Dancer Speed Control

721.45 InPut 721.5 SetPoint

-2.4 OutPut

0.0 ManSetVal

3 Mode 3 State

0 DancerPos 630

SET 0

□ HP
□ En
□ Bsy

S120UnW UnWinder, 7.5 kW

+351.91 rpm 0.0 rpm

+29.645 TrqAdd

0.2829 Diameter 1.0666

□ HP 0

11. Programming

CEA Winder_Control includes all objectives we mention before.
For the actual version we have decided to call the winder FB in OB1 where the call distributor takes place.

```
"DI OUT P01"();  // Execute output stage
```

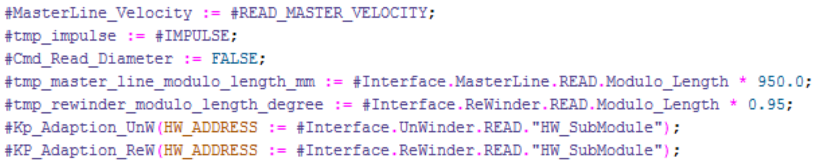
12. Winder Control

The FB WINDER CONTROL consists of 8 Networks and 7 sub functions such as ramping, scaling or interpolation functions. The FB includes a communication Interface from which reading extern signals and writing to extern signals is possible. In order to configure your winder click SCADA and write in the input fields (green background) the date you get from your machine designer. Secondly, click OEM. For error-free operation do not ignore parametrisation needed for the winder or drives. To do so, open FB OUT P01 and add the default values necessary, e.g.: Acce_Dece_TON time, or in winder sub functions

Seed_RampUp_Time_IN, etc. After PLC start-up there should be no zero values presented on your OEM.

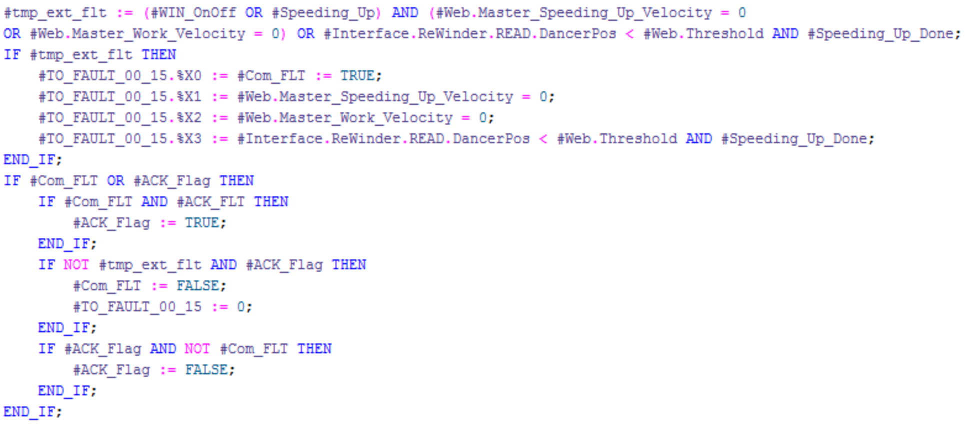
13. CEA Winder Control - PLC code with Siemens TIA V19

Network 1: Pre-Setting



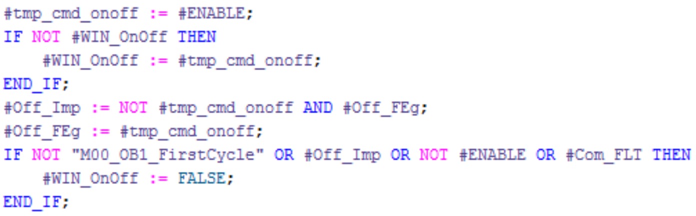
```
#MasterLine_Velocity := #READ_MASTER_VELOCITY;
#tmp_impulse := #IMPULSE;
#Cmd_Read_Diameter := FALSE;
#tmp_master_line_modulo_length_mm := #Interface.MasterLine.READ.Modulo_Length * 950.0;
#tmp_rewinder_modulo_length_degree := #Interface.ReWinder.READ.Modulo_Length * 0.95;
#Kp_Adaption_UnW(HW_ADDRESS := #Interface.UnWinder.READ."HW_SubModule");
#KP_Adaption_ReW(HW_ADDRESS := #Interface.ReWinder.READ."HW_SubModule");
```

Network 2: Fault detection



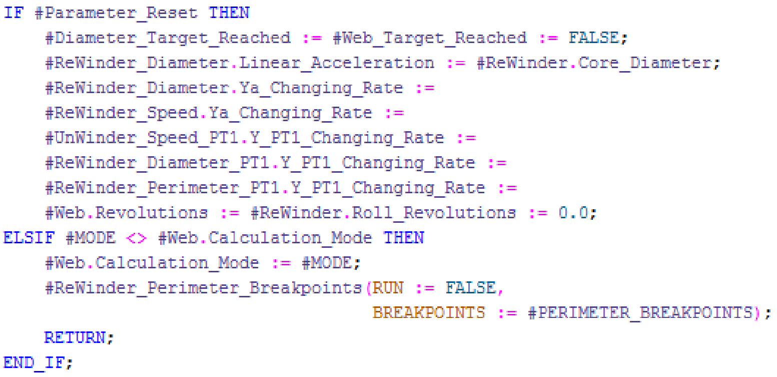
```
#tmp_ext_flt := (#WIN_OnOff OR #Speeding_Up) AND (#Web.Master_Speeding_Up_Velocity = 0
OR #Web.Master_Work_Velocity = 0) OR #Interface.ReWinder.READ.DancerPos < #Web.Threshold AND #Speeding_Up_Done;
IF #tmp_ext_flt THEN
    #TO_FAULT_00_15.%X0 := #Com_FLT := TRUE;
    #TO_FAULT_00_15.%X1 := #Web.Master_Speeding_Up_Velocity = 0;
    #TO_FAULT_00_15.%X2 := #Web.Master_Work_Velocity = 0;
    #TO_FAULT_00_15.%X3 := #Interface.ReWinder.READ.DancerPos < #Web.Threshold AND #Speeding_Up_Done;
END_IF;
IF #Com_FLT OR #ACK_Flag THEN
    IF #Com_FLT AND #ACK_FLT THEN
        #ACK_Flag := TRUE;
    END_IF;
    IF NOT #tmp_ext_flt AND #ACK_Flag THEN
        #Com_FLT := FALSE;
        #TO_FAULT_00_15 := 0;
    END_IF;
    IF #ACK_Flag AND NOT #Com_FLT THEN
        #ACK_Flag := FALSE;
    END_IF;
END_IF;
```

Network 3: Enable



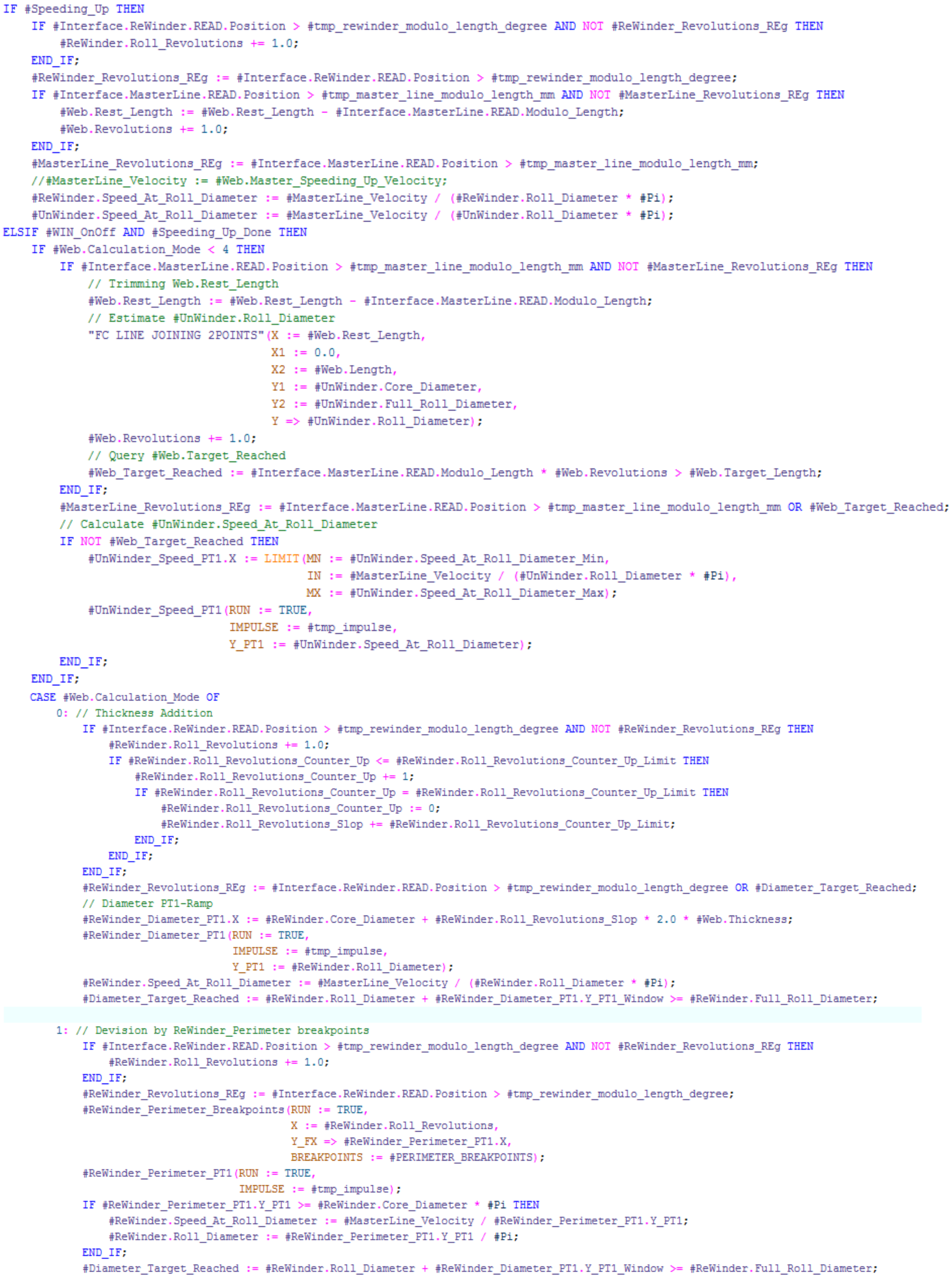
```
#tmp_cmd_onoff := #ENABLE;
IF NOT #WIN_OnOff THEN
    #WIN_OnOff := #tmp_cmd_onoff;
END_IF;
#Off_Imp := NOT #tmp_cmd_onoff AND #Off_FEg;
#Off_FEg := #tmp_cmd_onoff;
IF NOT "M00_OB1_FirstCycle" OR #Off_Imp OR NOT #ENABLE OR #Com_FLT THEN
    #WIN_OnOff := FALSE;
END_IF;
```

Network 4: Web pre-set



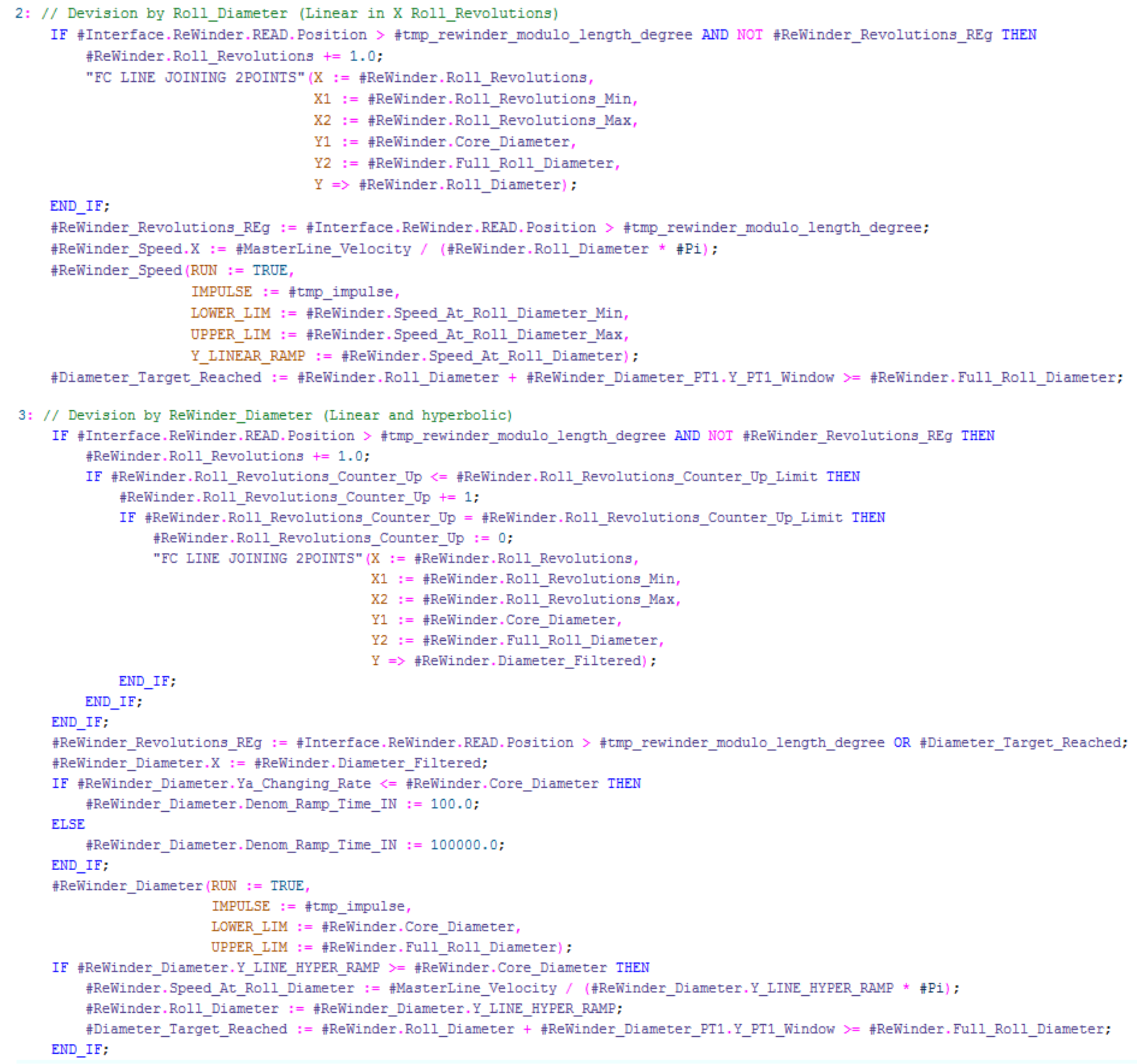
```
IF #Parameter_Reset THEN
    #Diameter_Target_Reached := #Web_Target_Reached := FALSE;
    #ReWinder_Diameter.Linear_Acceleration := #ReWinder.Core_Diameter;
    #ReWinder_Diameter.Ya_Changing_Rate :=
    #ReWinder_Speed.Ya_Changing_Rate :=
    #UnWinder_Speed_PT1.Y_PT1_Changing_Rate :=
    #ReWinder_Diameter_PT1.Y_PT1_Changing_Rate :=
    #ReWinder_Perimeter_PT1.Y_PT1_Changing_Rate :=
    #Web.Revolutions := #ReWinder.Roll_Revolutions := 0.0;
ELSIF #MODE <> #Web.Calculation_Mode THEN
    #Web.Calculation_Mode := #MODE;
    #ReWinder_Perimeter_Breakpoints(RUN := FALSE,
                                    BREAKPOINTS := #PERIMETER_BREAKPOINTS);
    RETURN;
END_IF;
```

Network 5: Diameter Calculation Mode



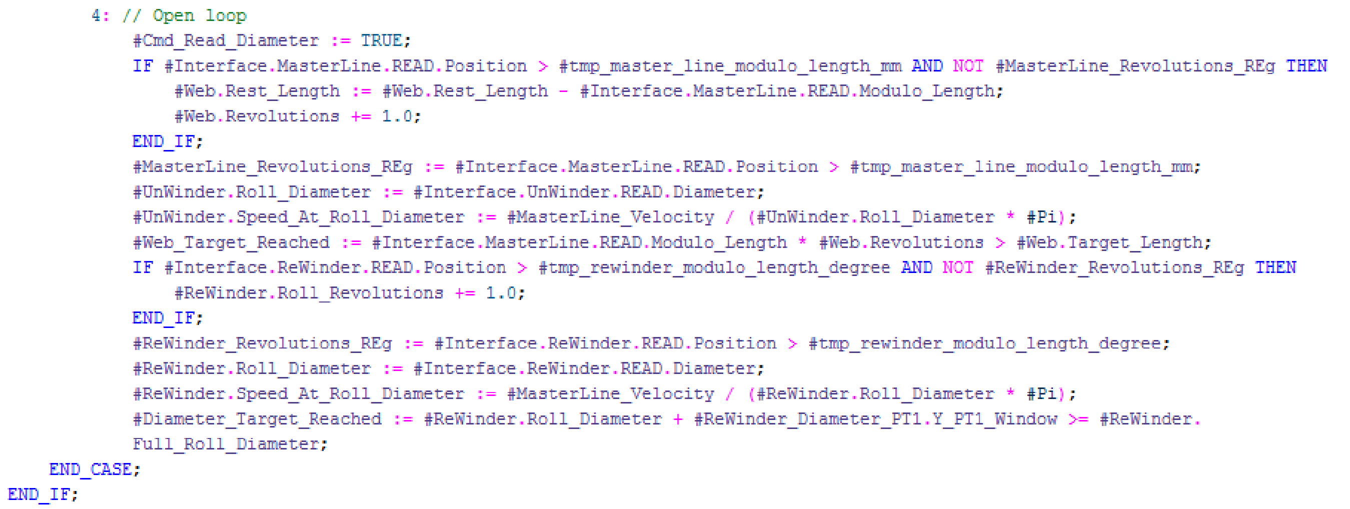
```
IF #Speeding_Up THEN
    IF #Interface.ReWinder.READ.Position > #tmp_rewinder_modulo_length_degree AND NOT #ReWinder_Revolutions_REg THEN
        #ReWinder.Roll_Revolutions += 1.0;
    END_IF;
    #ReWinder_Revolutions_REg := #Interface.ReWinder.READ.Position > #tmp_rewinder_modulo_length_degree;
    IF #Interface.MasterLine.READ.Position > #tmp_master_line_modulo_length_mm AND NOT #MasterLine_Revolutions_REg THEN
        #Web.Rest_Length := #Web.Rest_Length - #Interface.MasterLine.READ.Modulo_Length;
        #Web.Revolutions += 1.0;
    END_IF;
    #MasterLine_Revolutions_REg := #Interface.MasterLine.READ.Position > #tmp_master_line_modulo_length_mm;
    //#MasterLine_Velocity := #Web.Master_Speeding_Up_Velocity;
    #ReWinder.Speed_At_Roll_Diameter := #MasterLine_Velocity / (#ReWinder.Roll_Diameter * #Pi);
    #UnWinder.Speed_At_Roll_Diameter := #MasterLine_Velocity / (#UnWinder.Roll_Diameter * #Pi);
ELSIF #WIN_OnOff AND #Speeding_Up_Done THEN
    IF #Web.Calculation_Mode < 4 THEN
        IF #Interface.MasterLine.READ.Position > #tmp_master_line_modulo_length_mm AND NOT #MasterLine_Revolutions_REg THEN
            // Trimming Web.Rest_Length
            #Web.Rest_Length := #Web.Rest_Length - #Interface.MasterLine.READ.Modulo_Length;
            // Estimate #UnWinder.Roll_Diameter
            "FC LINE JOINING 2POINTS"(X := #Web.Rest_Length,
                                      X1 := 0.0,
                                      X2 := #Web.Length,
                                      Y1 := #UnWinder.Core_Diameter,
                                      Y2 := #UnWinder.Full_Roll_Diameter,
                                      Y => #UnWinder.Roll_Diameter);
            #Web.Revolutions += 1.0;
            // Query #Web.Target_Reached
            #Web_Target_Reached := #Interface.MasterLine.READ.Modulo_Length * #Web.Revolutions > #Web.Target_Length;
        END_IF;
        #MasterLine_Revolutions_REg := #Interface.MasterLine.READ.Position > #tmp_master_line_modulo_length_mm OR #Web_Target_Reached;
        // Calculate #UnWinder.Speed_At_Roll_Diameter
        IF NOT #Web_Target_Reached THEN
            #UnWinder_Speed_PT1.X := LIMIT(MN := #UnWinder.Speed_At_Roll_Diameter_Min,
                                           IN := #MasterLine_Velocity / (#UnWinder.Roll_Diameter * #Pi),
                                           MX := #UnWinder.Speed_At_Roll_Diameter_Max);
            #UnWinder_Speed_PT1(RUN := TRUE,
                                IMPULSE := #tmp_impulse,
                                Y_PT1 := #UnWinder.Speed_At_Roll_Diameter);
        END_IF;
    END_IF;
    CASE #Web.Calculation_Mode OF
        0: // Thickness Addition
            IF #Interface.ReWinder.READ.Position > #tmp_rewinder_modulo_length_degree AND NOT #ReWinder_Revolutions_REg THEN
                #ReWinder.Roll_Revolutions += 1.0;
                IF #ReWinder.Roll_Revolutions_Counter_Up <= #ReWinder.Roll_Revolutions_Counter_Up_Limit THEN
                    #ReWinder.Roll_Revolutions_Counter_Up += 1;
                    IF #ReWinder.Roll_Revolutions_Counter_Up = #ReWinder.Roll_Revolutions_Counter_Up_Limit THEN
                        #ReWinder.Roll_Revolutions_Counter_Up := 0;
                        #ReWinder.Roll_Revolutions_Slop += #ReWinder.Roll_Revolutions_Counter_Up_Limit;
                    END_IF;
                END_IF;
            END_IF;
            #ReWinder_Revolutions_REg := #Interface.ReWinder.READ.Position > #tmp_rewinder_modulo_length_degree OR #Diameter_Target_Reached;
            // Diameter PT1-Ramp
            #ReWinder_Diameter_PT1.X := #ReWinder.Core_Diameter + #ReWinder.Roll_Revolutions_Slop * 2.0 * #Web.Thickness;
            #ReWinder_Diameter_PT1(RUN := TRUE,
                                   IMPULSE := #tmp_impulse,
                                   Y_PT1 := #ReWinder.Roll_Diameter);
            #ReWinder.Speed_At_Roll_Diameter := #MasterLine_Velocity / (#ReWinder.Roll_Diameter * #Pi);
            #Diameter_Target_Reached := #ReWinder.Roll_Diameter + #ReWinder_Diameter_PT1.Y_PT1_Window >= #ReWinder.Full_Roll_Diameter;

        1: // Devision by ReWinder_Perimeter breakpoints
            IF #Interface.ReWinder.READ.Position > #tmp_rewinder_modulo_length_degree AND NOT #ReWinder_Revolutions_REg THEN
                #ReWinder.Roll_Revolutions += 1.0;
            END_IF;
            #ReWinder_Revolutions_REg := #Interface.ReWinder.READ.Position > #tmp_rewinder_modulo_length_degree;
            #ReWinder_Perimeter_Breakpoints(RUN := TRUE,
                                            X := #ReWinder.Roll_Revolutions,
                                            Y_FX => #ReWinder_Perimeter_PT1.X,
                                            BREAKPOINTS := #PERIMETER_BREAKPOINTS);
            #ReWinder_Perimeter_PT1(RUN := TRUE,
                                    IMPULSE := #tmp_impulse);
            IF #ReWinder_Perimeter_PT1.Y_PT1 >= #ReWinder.Core_Diameter * #Pi THEN
                #ReWinder.Speed_At_Roll_Diameter := #MasterLine_Velocity / #ReWinder_Perimeter_PT1.Y_PT1;
                #ReWinder.Roll_Diameter := #ReWinder_Perimeter_PT1.Y_PT1 / #Pi;
            END_IF;
            #Diameter_Target_Reached := #ReWinder.Roll_Diameter + #ReWinder_Diameter_PT1.Y_PT1_Window >= #ReWinder.Full_Roll_Diameter;
```



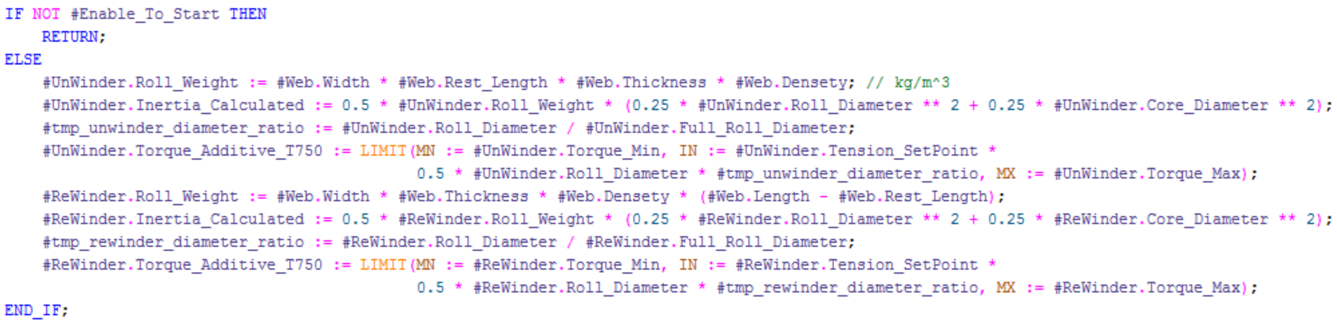
```
2: // Devision by Roll_Diameter (Linear in X Roll_Revolutions)
    IF #Interface.ReWinder.READ.Position > #tmp_rewinder_modulo_length_degree AND NOT #ReWinder_Revolutions_REg THEN
        #ReWinder.Roll_Revolutions += 1.0;
        "FC LINE JOINING 2POINTS"(X := #ReWinder.Roll_Revolutions,
                                  X1 := #ReWinder.Roll_Revolutions_Min,
                                  X2 := #ReWinder.Roll_Revolutions_Max,
                                  Y1 := #ReWinder.Core_Diameter,
                                  Y2 := #ReWinder.Full_Roll_Diameter,
                                  Y => #ReWinder.Roll_Diameter);
    END_IF;
    #ReWinder_Revolutions_REg := #Interface.ReWinder.READ.Position > #tmp_rewinder_modulo_length_degree;
    #ReWinder_Speed.X := #MasterLine_Velocity / (#ReWinder.Roll_Diameter * #Pi);
    #ReWinder_Speed(RUN := TRUE,
                    IMPULSE := #tmp_impulse,
                    LOWER_LIM := #ReWinder.Speed_At_Roll_Diameter_Min,
                    UPPER_LIM := #ReWinder.Speed_At_Roll_Diameter_Max,
                    Y_LINEAR_RAMP := #ReWinder.Speed_At_Roll_Diameter);
    #Diameter_Target_Reached := #ReWinder.Roll_Diameter + #ReWinder_Diameter_PT1.Y_PT1_Window >= #ReWinder.Full_Roll_Diameter;

3: // Devision by ReWinder_Diameter (Linear and hyperbolic)
    IF #Interface.ReWinder.READ.Position > #tmp_rewinder_modulo_length_degree AND NOT #ReWinder_Revolutions_REg THEN
        #ReWinder.Roll_Revolutions += 1.0;
        IF #ReWinder.Roll_Revolutions_Counter_Up <= #ReWinder.Roll_Revolutions_Counter_Up_Limit THEN
            #ReWinder.Roll_Revolutions_Counter_Up += 1;
            IF #ReWinder.Roll_Revolutions_Counter_Up = #ReWinder.Roll_Revolutions_Counter_Up_Limit THEN
                #ReWinder.Roll_Revolutions_Counter_Up := 0;
                "FC LINE JOINING 2POINTS"(X := #ReWinder.Roll_Revolutions,
                                          X1 := #ReWinder.Roll_Revolutions_Min,
                                          X2 := #ReWinder.Roll_Revolutions_Max,
                                          Y1 := #ReWinder.Core_Diameter,
                                          Y2 := #ReWinder.Full_Roll_Diameter,
                                          Y => #ReWinder.Diameter_Filtered);
            END_IF;
        END_IF;
    END_IF;
    #ReWinder_Revolutions_REg := #Interface.ReWinder.READ.Position > #tmp_rewinder_modulo_length_degree OR #Diameter_Target_Reached;
    #ReWinder_Diameter.X := #ReWinder.Diameter_Filtered;
    IF #ReWinder_Diameter.Ya_Changing_Rate <= #ReWinder.Core_Diameter THEN
        #ReWinder_Diameter.Denom_Ramp_Time_IN := 100.0;
    ELSE
        #ReWinder_Diameter.Denom_Ramp_Time_IN := 100000.0;
    END_IF;
    #ReWinder_Diameter(RUN := TRUE,
                       IMPULSE := #tmp_impulse,
                       LOWER_LIM := #ReWinder.Core_Diameter,
                       UPPER_LIM := #ReWinder.Full_Roll_Diameter);
    IF #ReWinder_Diameter.Y_LINE_HYPER_RAMP >= #ReWinder.Core_Diameter THEN
        #ReWinder.Speed_At_Roll_Diameter := #MasterLine_Velocity / (#ReWinder_Diameter.Y_LINE_HYPER_RAMP * #Pi);
        #ReWinder.Roll_Diameter := #ReWinder_Diameter.Y_LINE_HYPER_RAMP;
        #Diameter_Target_Reached := #ReWinder.Roll_Diameter + #ReWinder_Diameter_PT1.Y_PT1_Window >= #ReWinder.Full_Roll_Diameter;
    END_IF;
```



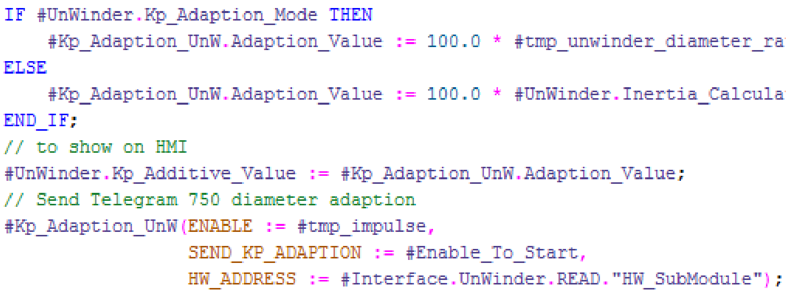
```
        4: // Open loop
            #Cmd_Read_Diameter := TRUE;
            IF #Interface.MasterLine.READ.Position > #tmp_master_line_modulo_length_mm AND NOT #MasterLine_Revolutions_REg THEN
                #Web.Rest_Length := #Web.Rest_Length - #Interface.MasterLine.READ.Modulo_Length;
                #Web.Revolutions += 1.0;
            END_IF;
            #MasterLine_Revolutions_REg := #Interface.MasterLine.READ.Position > #tmp_master_line_modulo_length_mm;
            #UnWinder.Roll_Diameter := #Interface.UnWinder.READ.Diameter;
            #UnWinder.Speed_At_Roll_Diameter := #MasterLine_Velocity / (#UnWinder.Roll_Diameter * #Pi);
            #Web_Target_Reached := #Interface.MasterLine.READ.Modulo_Length * #Web.Revolutions > #Web.Target_Length;
            IF #Interface.ReWinder.READ.Position > #tmp_rewinder_modulo_length_degree AND NOT #ReWinder_Revolutions_REg THEN
                #ReWinder.Roll_Revolutions += 1.0;
            END_IF;
            #ReWinder_Revolutions_REg := #Interface.ReWinder.READ.Position > #tmp_rewinder_modulo_length_degree;
            #ReWinder.Roll_Diameter := #Interface.ReWinder.READ.Diameter;
            #ReWinder.Speed_At_Roll_Diameter := #MasterLine_Velocity / (#ReWinder.Roll_Diameter * #Pi);
            #Diameter_Target_Reached := #ReWinder.Roll_Diameter + #ReWinder_Diameter_PT1.Y_PT1_Window >= #ReWinder.
            Full_Roll_Diameter;
    END_CASE;
END_IF;
```

Network 6: Inertia and Torque_T750 calculation



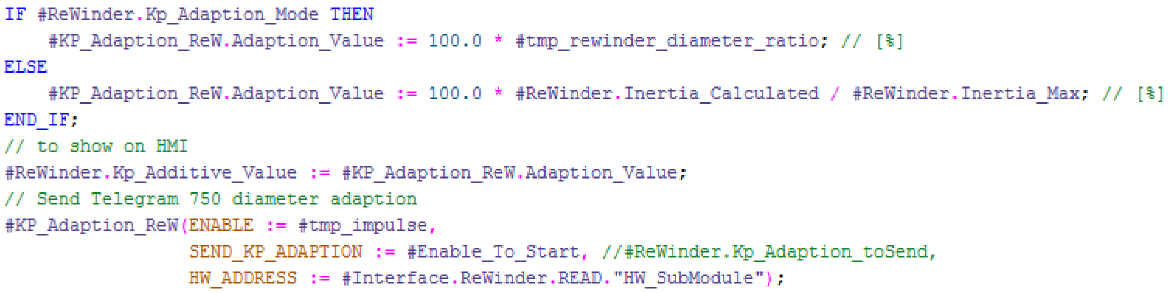
```
IF NOT #Enable_To_Start THEN
    RETURN;
ELSE
    #UnWinder.Roll_Weight := #Web.Width * #Web.Rest_Length * #Web.Thickness * #Web.Densety; // kg/m^3
    #UnWinder.Inertia_Calculated := 0.5 * #UnWinder.Roll_Weight * (0.25 * #UnWinder.Roll_Diameter ** 2 + 0.25 * #UnWinder.Core_Diameter ** 2);
    #tmp_unwinder_diameter_ratio := #UnWinder.Roll_Diameter / #UnWinder.Full_Roll_Diameter;
    #UnWinder.Torque_Additive_T750 := LIMIT(MN := #UnWinder.Torque_Min, IN := #UnWinder.Tension_SetPoint *
                                            0.5 * #UnWinder.Roll_Diameter * #tmp_unwinder_diameter_ratio, MX := #UnWinder.Torque_Max);
    #ReWinder.Roll_Weight := #Web.Width * #Web.Thickness * #Web.Densety * (#Web.Length - #Web.Rest_Length);
    #ReWinder.Inertia_Calculated := 0.5 * #ReWinder.Roll_Weight * (0.25 * #ReWinder.Roll_Diameter ** 2 + 0.25 * #ReWinder.Core_Diameter ** 2);
    #tmp_rewinder_diameter_ratio := #ReWinder.Roll_Diameter / #ReWinder.Full_Roll_Diameter;
    #ReWinder.Torque_Additive_T750 := LIMIT(MN := #ReWinder.Torque_Min, IN := #ReWinder.Tension_SetPoint *
                                            0.5 * #ReWinder.Roll_Diameter * #tmp_rewinder_diameter_ratio, MX := #ReWinder.Torque_Max);
END_IF;
```

Network 7: Kp_Adaption UnWinder Telegramm 750



```
IF #UnWinder.Kp_Adaption_Mode THEN
    #Kp_Adaption_UnW.Adaption_Value := 100.0 * #tmp_unwinder_diameter_ratio; // [%]
ELSE
    #Kp_Adaption_UnW.Adaption_Value := 100.0 * #UnWinder.Inertia_Calculated / #UnWinder.Inertia_Max; // [%]
END_IF;
// to show on HMI
#UnWinder.Kp_Additive_Value := #Kp_Adaption_UnW.Adaption_Value;
// Send Telegram 750 diameter adaption
#Kp_Adaption_UnW(ENABLE := #tmp_impulse,
                 SEND_KP_ADAPTION := #Enable_To_Start,
                 HW_ADDRESS := #Interface.UnWinder.READ."HW_SubModule");
```

Network 8: Kp_Adaption ReWinder Telegramm 750



```
IF #ReWinder.Kp_Adaption_Mode THEN
    #KP_Adaption_ReW.Adaption_Value := 100.0 * #tmp_rewinder_diameter_ratio; // [%]
ELSE
    #KP_Adaption_ReW.Adaption_Value := 100.0 * #ReWinder.Inertia_Calculated / #ReWinder.Inertia_Max; // [%]
END_IF;
// to show on HMI
#ReWinder.Kp_Additive_Value := #KP_Adaption_ReW.Adaption_Value;
// Send Telegram 750 diameter adaption
#KP_Adaption_ReW(ENABLE := #tmp_impulse,
                 SEND_KP_ADAPTION := #Enable_To_Start, //#ReWinder.Kp_Adaption_toSend,
                 HW_ADDRESS := #Interface.ReWinder.READ."HW_SubModule");
```

14. **Dancer rolls PID control as a Supplement to Rewinder Speed.**

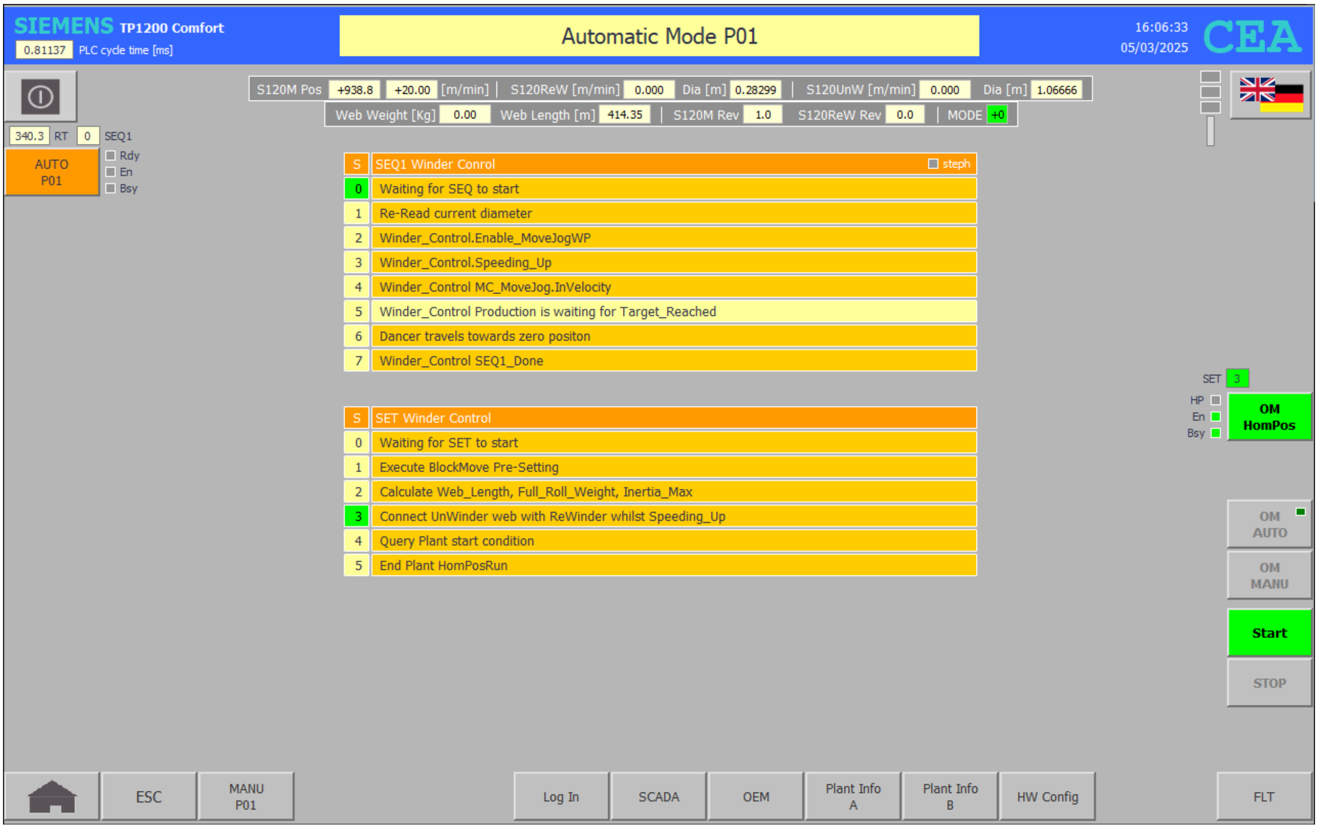
The dancer supervises the surface tension at the Rewinder zone. If tension decries the dancer roll writes a positive value to the Rewinder offset, if tension increases a negative value, hence eliminate speed oscillation.
The offset is being added to Rewinder speed set point. PID control reads the result set point and compares it with Rewinder current speed. The PID output value lies between -10.0 and +10.0 percent. This value serves as a speed correction in OB30.

15. Load Cell Measurement Control as a Supplement to Unwinder Speed

The load cell censes the surface tension at the Unwinder zone. If tension decries the load cell writes a negative value to the Unwinder offset, if tension increases a positive value, hence eliminate speed fluctuation. The load cell output serves as correction and is being added to the speed set point in DI OUT P01.

16. Operating mode HomPos

There is no way to start automatic or semi-automatic without going through plant start position. Operation mode HomPos calls FB SET P01.



SIEMENS TP1200 Comfort
0.81137 PLC cycle time [ms]
Automatic Mode P01
16:06:33
05/03/2025

S120M Pos +938.8 +20.00 [m/min] | S120ReW [m/min] 0.000 Dia [m] 0.28299 | S120UnW [m/min] 0.000 Dia [m] 1.06666
Web Weight [Kg] 0.00 Web Length [m] 414.35 | S120M Rev 1.0 S120ReW Rev 0.0 | MODE +0

340.3 RT 0 SEQ1
□ Rdy
□ En
□ Bsy

S	SEQ1 Winder Conrol □ steph
0	Waiting for SEQ to start
1	Re-Read current diameter
2	Winder_Control.Enable_MoveJogWP
3	Winder_Control.Speeding_Up
4	Winder_Control MC_MoveJog.InVelocity
5	Winder_Control Production is waiting for Target_Reached
6	Dancer travels towards zero positon
7	Winder_Control SEQ1_Done

S	SET Winder Control
0	Waiting for SET to start
1	Execute BlockMove Pre-Setting
2	Calculate Web_Length, Full_Roll_Weight, Inertia_Max
3	Connect UnWinder web with ReWinder whilst Speeding_Up
4	Query Plant start condition
5	End Plant HomPosRun

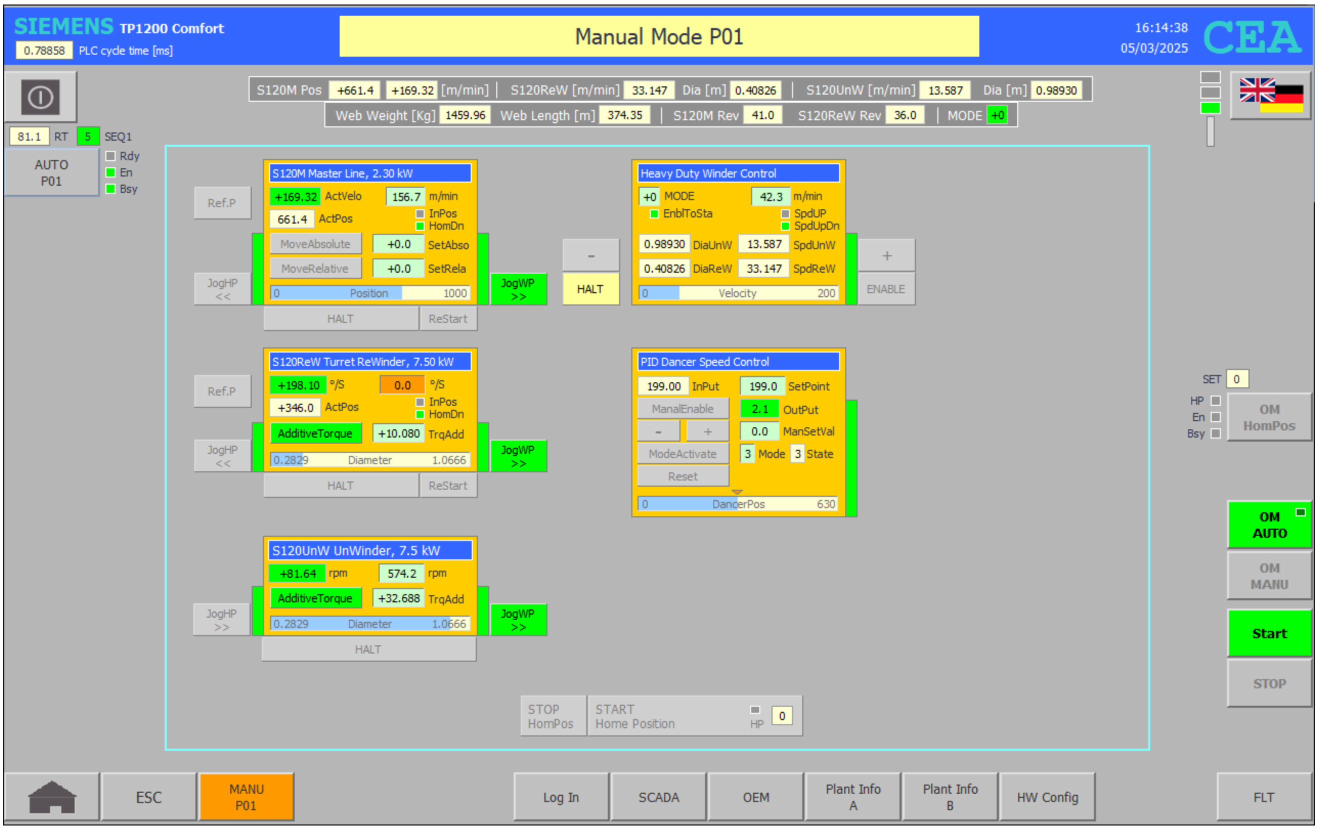
SET 3
□ HP
🟩 En
🟩 Bsy

The function copies the data written in SCADA into the provided process memory DI OUT P01.
Amongst others, the function calculates the web length, web weight, and inertia of the Unwinder.
Further, the function creates a web connection between full roll Unwinder and the Rewinder mandrel. In this case the web is traveling through the machine starting off at the Unwinder, through Master Line drive roll, through dancer roll ending up at the Rewinder mandrel. Before ending, the function checks whether the intended calculation mode (0 to 4) has been applied.

- The default [-1] means no value has been set.

17. Running in Automatic Mode

FB MAIN P01 calls FB SEQ1 P01 to run the plant process sequencer.



SIEMENS TP1200 Comfort

0.78858 PLC cycle time [ms]

Manual Mode P01

16:14:38
05/03/2025

S120M Pos +661.4 +169.32 [m/min] | S120ReW [m/min] 33.147 Dia [m] 0.40826 | S120UnW [m/min] 13.587 Dia [m] 0.98930

Web Weight [Kg] 1459.96 Web Length [m] 374.35 | S120M Rev 41.0 S120ReW Rev 36.0 | MODE +0

81.1 RT 5 SEQ1

AUTO P01 □ Rdy ■ En ■ Bsy

S120M Master Line, 2.30 kW
+169.32 ActVelo 156.7 m/min
661.4 ActPos □ InPos ■ HomDn
MoveAbsolute +0.0 SetAbso
MoveRelative +0.0 SetRela
0 Position 1000

Heavy Duty Winder Control
+0 MODE 42.3 m/min
■ EnblToSta □ SpdUP ■ SpdUpDn
0.98930 DiaUnW 13.587 SpdUnW
0.40826 DiaReW 33.147 SpdReW
0 Velocity 200

S120ReW Turret ReWinder, 7.50 kW
+198.10 °/S 0.0 °/S
+346.0 ActPos □ InPos ■ HomDn
AdditiveTorque +10.080 TrqAdd
0.2829 Diameter 1.0666

PID Dancer Speed Control
199.00 InPut 199.0 SetPoint
ManalEnable 2.1 OutPut
0.0 ManSetVal
ModeActivate 3 Mode 3 State
Reset
0 DancerPos 630

S120UnW UnWinder, 7.5 kW
+81.64 rpm 574.2 rpm
AdditiveTorque +32.688 TrqAdd
0.2829 Diameter 1.0666

STOP HomPos | START Home Position □ HP 0

SET 0 □ HP □ En □ Bsy

Note: in this very plant, the presented S120M Master Line velocity (ActVelo) you are seeing on HMI is the velocity on the motor shaft side and does not meet the load rotation. In other words, it is the velocity assigned to Winder Control – i.e. 42.3 m/min times Gear_Ratio.

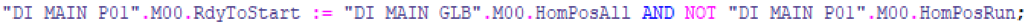
The same applies to actual position - it is the position of at the load side i.e. 661.4, ranged: 0 – 1000.

18. Physical Equation used in this Application

Web.Rest_Length := Pi * (Roll_Diameter^2 – Core_Diameter^2) / (4.0 * Web.Thickness);
Roll_Weight := Web.Rest_Length * Web.Width * Web.Thickness * Web.Densety;
Speed_At_Roll_Diameter := MasterLine_Velocity / (Roll_Diameter * Pi);
Inertia_Calculated := 0.5 * Roll_Weight * (0.25 * Roll_Diameter^2 + 0.25 * Core_Diameter^2);
Diameter_Ratio := Roll_Diameter / Full_Roll_Diameter;
Torque_Additive := Tension_SetPoint * 0.5 * Roll_Diameter * Diameter_Ratio;
Kp_Adaption_Value_1 := 100.0 * Diameter_Ratio;
Kp_Adaption_Value_2 := 100.0 * Inertia_Calculated / Inertia_Max;
Web_Target_Reached := MasterLine.Modulo_Lenght * Web.Revolution > Web.Target_Length;
Diameter_Target_Reached := Roll_Diameter + Web.Thickness >= Target_Roll_Diameter;
WRITE.Velocity := Speed_At_Roll_Diameter * READ.Gear_Ratio;

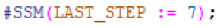
19. Winder Control Sequencer

Network 1: Pre-Setting



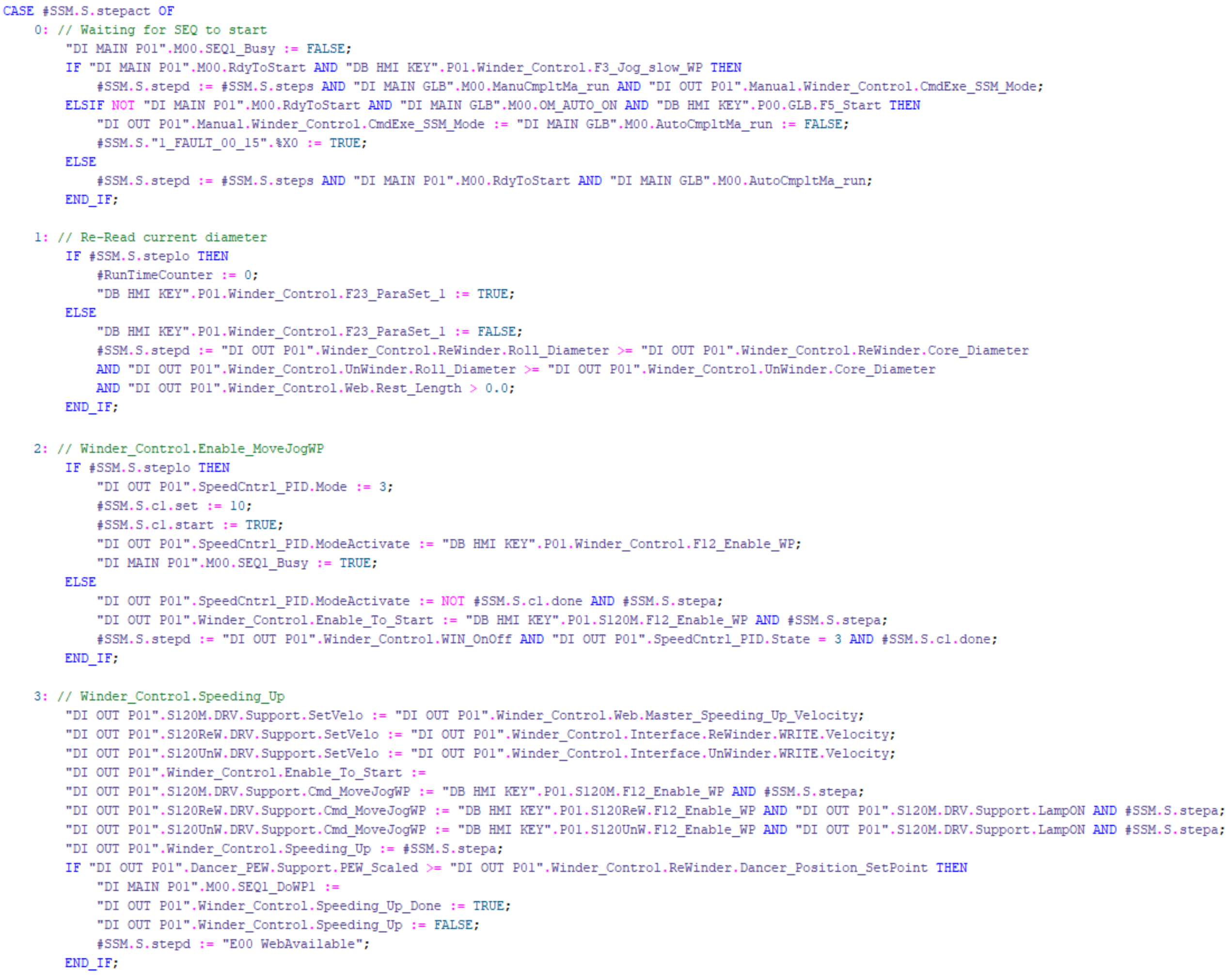
```
"DI MAIN P01".M00.RdyToStart := "DI MAIN GLB".M00.HomPosAll AND NOT "DI MAIN P01".M00.HomPosRun;
```

Network 2: Define last step



```
#SSM(LAST_STEP := 7);
```

Network 3: Step Switching Mechanism (SSM)

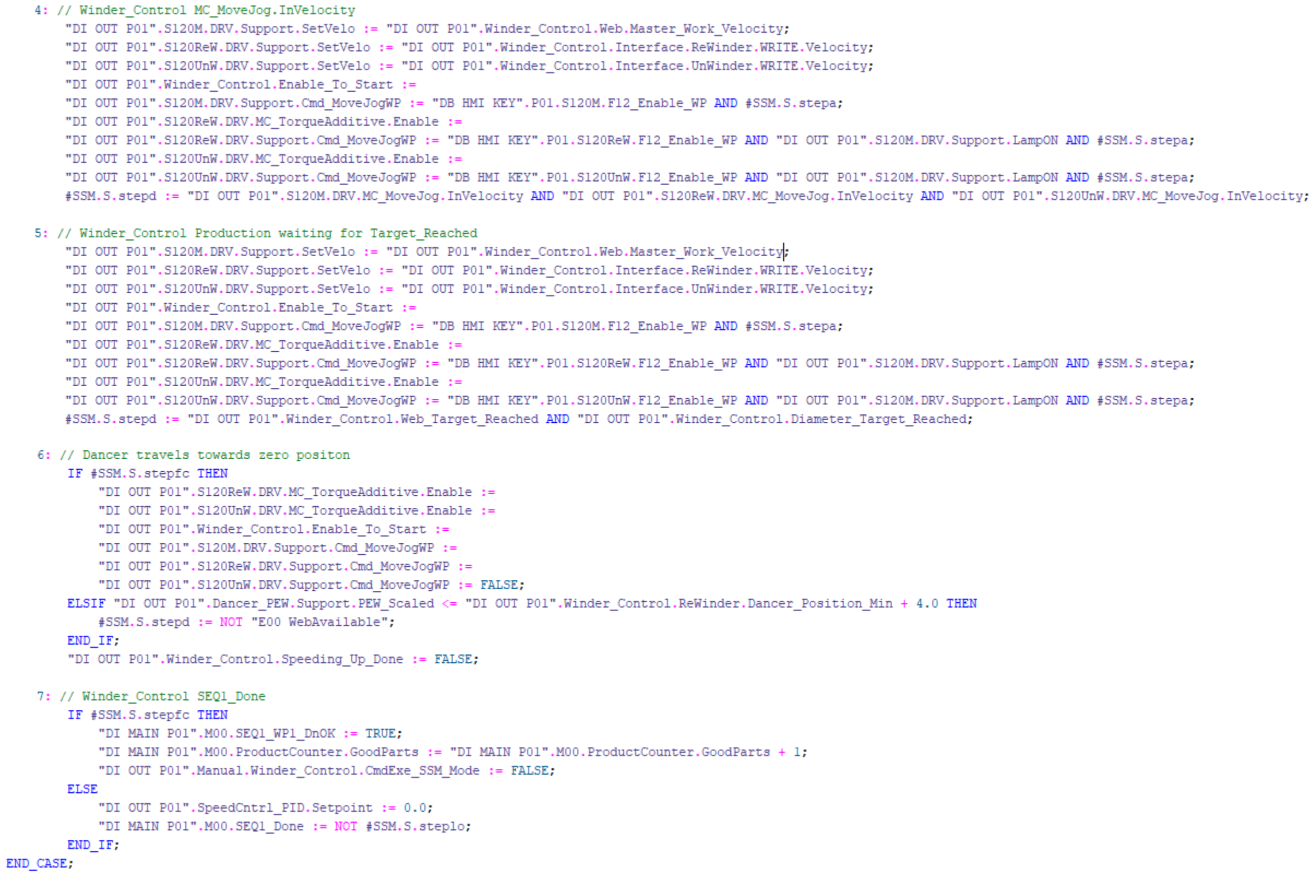


```
CASE #SSM.S.stepact OF
    0: // Waiting for SEQ to start
        "DI MAIN P01".M00.SEQ1_Busy := FALSE;
        IF "DI MAIN P01".M00.RdyToStart AND "DB HMI KEY".P01.Winder_Control.F3_Jog_slow_WP THEN
            #SSM.S.stepd := #SSM.S.steps AND "DI MAIN GLB".M00.ManuCmpltMa_run AND "DI OUT P01".Manual.Winder_Control.CmdExe_SSM_Mode;
        ELSIF NOT "DI MAIN P01".M00.RdyToStart AND "DI MAIN GLB".M00.OM_AUTO_ON AND "DB HMI KEY".P00.GLB.F5_Start THEN
            "DI OUT P01".Manual.Winder_Control.CmdExe_SSM_Mode := "DI MAIN GLB".M00.AutoCmpltMa_run := FALSE;
            #SSM.S."1_FAULT_00_15".%X0 := TRUE;
        ELSE
            #SSM.S.stepd := #SSM.S.steps AND "DI MAIN P01".M00.RdyToStart AND "DI MAIN GLB".M00.AutoCmpltMa_run;
        END_IF;

    1: // Re-Read current diameter
        IF #SSM.S.steplo THEN
            #RunTimeCounter := 0;
            "DB HMI KEY".P01.Winder_Control.F23_ParaSet_1 := TRUE;
        ELSE
            "DB HMI KEY".P01.Winder_Control.F23_ParaSet_1 := FALSE;
            #SSM.S.stepd := "DI OUT P01".Winder_Control.ReWinder.Roll_Diameter >= "DI OUT P01".Winder_Control.ReWinder.Core_Diameter
            AND "DI OUT P01".Winder_Control.UnWinder.Roll_Diameter >= "DI OUT P01".Winder_Control.UnWinder.Core_Diameter
            AND "DI OUT P01".Winder_Control.Web.Rest_Length > 0.0;
        END_IF;

    2: // Winder_Control.Enable_MoveJogWP
        IF #SSM.S.steplo THEN
            "DI OUT P01".SpeedCntrl_PID.Mode := 3;
            #SSM.S.cl.set := 10;
            #SSM.S.cl.start := TRUE;
            "DI OUT P01".SpeedCntrl_PID.ModeActivate := "DB HMI KEY".P01.Winder_Control.F12_Enable_WP;
            "DI MAIN P01".M00.SEQ1_Busy := TRUE;
        ELSE
            "DI OUT P01".SpeedCntrl_PID.ModeActivate := NOT #SSM.S.cl.done AND #SSM.S.stepa;
            "DI OUT P01".Winder_Control.Enable_To_Start := "DB HMI KEY".P01.S120M.F12_Enable_WP AND #SSM.S.stepa;
            #SSM.S.stepd := "DI OUT P01".Winder_Control.WIN_OnOff AND "DI OUT P01".SpeedCntrl_PID.State = 3 AND #SSM.S.cl.done;
        END_IF;

    3: // Winder_Control.Speeding_Up
        "DI OUT P01".S120M.DRV.Support.SetVelo := "DI OUT P01".Winder_Control.Web.Master_Speeding_Up_Velocity;
        "DI OUT P01".S120ReW.DRV.Support.SetVelo := "DI OUT P01".Winder_Control.Interface.ReWinder.WRITE.Velocity;
        "DI OUT P01".S120UnW.DRV.Support.SetVelo := "DI OUT P01".Winder_Control.Interface.UnWinder.WRITE.Velocity;
        "DI OUT P01".Winder_Control.Enable_To_Start :=
        "DI OUT P01".S120M.DRV.Support.Cmd_MoveJogWP := "DB HMI KEY".P01.S120M.F12_Enable_WP AND #SSM.S.stepa;
        "DI OUT P01".S120ReW.DRV.Support.Cmd_MoveJogWP := "DB HMI KEY".P01.S120ReW.F12_Enable_WP AND "DI OUT P01".S120M.DRV.Support.LampON AND #SSM.S.stepa;
        "DI OUT P01".S120UnW.DRV.Support.Cmd_MoveJogWP := "DB HMI KEY".P01.S120UnW.F12_Enable_WP AND "DI OUT P01".S120M.DRV.Support.LampON AND #SSM.S.stepa;
        "DI OUT P01".Winder_Control.Speeding_Up := #SSM.S.stepa;
        IF "DI OUT P01".Dancer_PEW.Support.PEW_Scaled >= "DI OUT P01".Winder_Control.ReWinder.Dancer_Position_SetPoint THEN
            "DI MAIN P01".M00.SEQ1_DoWP1 :=
            "DI OUT P01".Winder_Control.Speeding_Up_Done := TRUE;
            "DI OUT P01".Winder_Control.Speeding_Up := FALSE;
            #SSM.S.stepd := "E00 WebAvailable";
        END_IF;
```



```
4: // Winder_Control MC_MoveJog.InVelocity
    "DI OUT P01".S120M.DRV.Support.SetVelo := "DI OUT P01".Winder_Control.Web.Master_Work_Velocity;
    "DI OUT P01".S120ReW.DRV.Support.SetVelo := "DI OUT P01".Winder_Control.Interface.ReWinder.WRITE.Velocity;
    "DI OUT P01".S120UnW.DRV.Support.SetVelo := "DI OUT P01".Winder_Control.Interface.UnWinder.WRITE.Velocity;
    "DI OUT P01".Winder_Control.Enable_To_Start :=
    "DI OUT P01".S120M.DRV.Support.Cmd_MoveJogWP := "DB HMI KEY".P01.S120M.F12_Enable_WP AND #SSM.S.stepa;
    "DI OUT P01".S120ReW.DRV.MC_TorqueAdditive.Enable :=
    "DI OUT P01".S120ReW.DRV.Support.Cmd_MoveJogWP := "DB HMI KEY".P01.S120ReW.F12_Enable_WP AND "DI OUT P01".S120M.DRV.Support.LampON AND #SSM.S.stepa;
    "DI OUT P01".S120UnW.DRV.MC_TorqueAdditive.Enable :=
    "DI OUT P01".S120UnW.DRV.Support.Cmd_MoveJogWP := "DB HMI KEY".P01.S120UnW.F12_Enable_WP AND "DI OUT P01".S120M.DRV.Support.LampON AND #SSM.S.stepa;
    #SSM.S.stepd := "DI OUT P01".S120M.DRV.MC_MoveJog.InVelocity AND "DI OUT P01".S120ReW.DRV.MC_MoveJog.InVelocity AND "DI OUT P01".S120UnW.DRV.MC_MoveJog.InVelocity;

5: // Winder_Control Production waiting for Target_Reached
    "DI OUT P01".S120M.DRV.Support.SetVelo := "DI OUT P01".Winder_Control.Web.Master_Work_Velocity;
    "DI OUT P01".S120ReW.DRV.Support.SetVelo := "DI OUT P01".Winder_Control.Interface.ReWinder.WRITE.Velocity;
    "DI OUT P01".S120UnW.DRV.Support.SetVelo := "DI OUT P01".Winder_Control.Interface.UnWinder.WRITE.Velocity;
    "DI OUT P01".Winder_Control.Enable_To_Start :=
    "DI OUT P01".S120M.DRV.Support.Cmd_MoveJogWP := "DB HMI KEY".P01.S120M.F12_Enable_WP AND #SSM.S.stepa;
    "DI OUT P01".S120ReW.DRV.MC_TorqueAdditive.Enable :=
    "DI OUT P01".S120ReW.DRV.Support.Cmd_MoveJogWP := "DB HMI KEY".P01.S120ReW.F12_Enable_WP AND "DI OUT P01".S120M.DRV.Support.LampON AND #SSM.S.stepa;
    "DI OUT P01".S120UnW.DRV.MC_TorqueAdditive.Enable :=
    "DI OUT P01".S120UnW.DRV.Support.Cmd_MoveJogWP := "DB HMI KEY".P01.S120UnW.F12_Enable_WP AND "DI OUT P01".S120M.DRV.Support.LampON AND #SSM.S.stepa;
    #SSM.S.stepd := "DI OUT P01".Winder_Control.Web_Target_Reached AND "DI OUT P01".Winder_Control.Diameter_Target_Reached;

 6: // Dancer travels towards zero positon
    IF #SSM.S.stepfc THEN
        "DI OUT P01".S120ReW.DRV.MC_TorqueAdditive.Enable :=
        "DI OUT P01".S120UnW.DRV.MC_TorqueAdditive.Enable :=
        "DI OUT P01".Winder_Control.Enable_To_Start :=
        "DI OUT P01".S120M.DRV.Support.Cmd_MoveJogWP :=
        "DI OUT P01".S120ReW.DRV.Support.Cmd_MoveJogWP :=
        "DI OUT P01".S120UnW.DRV.Support.Cmd_MoveJogWP := FALSE;
    ELSIF "DI OUT P01".Dancer_PEW.Support.PEW_Scaled <= "DI OUT P01".Winder_Control.ReWinder.Dancer_Position_Min + 4.0 THEN
        #SSM.S.stepd := NOT "E00 WebAvailable";
    END_IF;
    "DI OUT P01".Winder_Control.Speeding_Up_Done := FALSE;

 7: // Winder_Control SEQ1_Done
    IF #SSM.S.stepfc THEN
        "DI MAIN P01".M00.SEQ1_WP1_DnOK := TRUE;
        "DI MAIN P01".M00.ProductCounter.GoodParts := "DI MAIN P01".M00.ProductCounter.GoodParts + 1;
        "DI OUT P01".Manual.Winder_Control.CmdExe_SSM_Mode := FALSE;
    ELSE
        "DI OUT P01".SpeedCntrl_PID.Setpoint := 0.0;
        "DI MAIN P01".M00.SEQ1_Done := NOT #SSM.S.steplo;
    END_IF;
END_CASE;
```

20. Learning from - and about

- SSD Drives, Inc. USA – Speed calculation of speed below or above speed at roll diameter, speed at speeding up phase, inertia calculation
- CAC Wind Cap. USA – Tension control webinar
- ABB Oy Finland – Technological assembly
- Siemens AG Germany – Torque Additive, Kp_Adaption, Motion Control V8

See also mp4 videos..