

```
1  PROGRAM PID_ST
2  VAR
3      // Integral gain for position loop PID
4      lIILocal : LTIME ;
5
6      // Definition of PID controller type 03 Velocity loop
7      PID_Velocity : IL_PIDType03 ;
8      // Definition of PID controller type 03 Position loop
9      PID_Position : IL_PIDType03 ;
10
11     // Definition of IIR filter type 01
12     LP_Filter_Position_Sensor : IL_IIRType01 ;
13     fbIIR1 : CXA_LOOPCONTROL . IL_IIRType01 ; // Declaration of filter type
14     tIIR_Coff : CXA_LOOPCONTROL . IL_IIR_COEFF ; // Configuration of filter
        coefficient
15
16
17     // Output from position loop PID to velocity setpoint
18     rPIDOut : LREAL ;
19
20     rdt : LREAL := 0.002 ; // time step for derivation
21     rVelocity : LREAL ; // Velocity output from derivation of ball position
        feedback signal
22     position : LREAL ; // stored position, used for derivation
23     prev_pos : LREAL := 0.0 ; // previous position, used for derivation
24     // Output from filtering of the ball position feedback
25     rLP_PosFilter_out : LREAL ;
26     // Angle generator for the sinus function
27     AngleGenerator : IL_AngleGeneratorType01 ;
28     // Sinus signal generator
29     SinusSignal : IL_SignalGeneratorType01 ;
30     // Output from angle generator to sinus generator angle input
31     rSinusAngle : LREAL ;
32     // Output from sinus generator to amplitude calculation
33     rSinusOut : LREAL ;
34     // kp gain for Position loop PID
35     rPILocal : LREAL ;
36     // Derivative gain for position loop PID
37     lDILocal : LTIME ;
38     // kp gain for Velocity loop PID
39     rPILocalVel : LREAL ;
40     // Integral gain for velocity loop PID
41     lIILocalVel : LTIME ;
42     // Derivative gain for velocity loop PID
43     lDILocalVel : LTIME ;
44 END_VAR
45
46
47 // Ball position control program
48
49
```

```

4      // Assigning global variables
5      GVL.rActualBeamPos := arAxisStatus_gb [ 1 ] . Data . ActualPosition ;
6      gvl . rFollowingError := ABS ( gvl . rBallPosCmd - gvl . rBallFeedback ) ;
7
8      // Setting local position loop pid variables to global variables from HMI
9      // rPLocal := gvl.rPGainPID; // reset DControl to original value
10     rPLocal := gvl . rPGainPID / 1000 ; // reset pControl to original value
11     lILocal := gvl . lICtrlPID ; // reset iControl to original value
12     lDLocal := gvl . lDCtrlPID ; // reset DControl to original value
13     // Setting local velocity loop pid variables to global variables from HMI
14     rPLocalVel := gvl . rPGainPIDVel ; // reset pControl to original value
15     lILocalVel := gvl . lICtrlPIDVel ; // reset iControl to original value
16     lDLocalVel := gvl . lDCtrlPIDVel ; // reset DControl to original value
17
18     // IIR filter coefficients from c2d conversion in matlab. Forward coefficients
19     // are the numerator values, b0,b1,b2..., and
20     // the Backward coefficients are the denominator values, 1,a1,a2....
21     // Single LP filter at frequency= 8.8 rad/s
22     tIIR_Coff . BackwardCoeffs [ 1 ] := gvl . rBwdCoeff1 ;
23     tIIR_Coff . BackwardCoeffs [ 2 ] := gvl . rBwdCoeff2 ;
24     tIIR_Coff . ForwardCoeffs [ 0 ] := gvl . rFwdCoeff1 ;
25     tIIR_Coff . ForwardCoeffs [ 1 ] := gvl . rFwdCoeff2 ;
26     tIIR_Coff . NumberOfCoeffs := 4 ;
27
28     //-----
29     // Generator for sinus signal for the sinus function in the HMI
30     AngleGenerator (
31         Enable := gvl . bEnablePID ,
32         InOperation => ,
33         Error => ,
34         ErrorID => ,
35         ErrorIdent => ,
36         Pause := ,
37         Frequency := gvl . rSinusFreq ,
38         InPause => ,
39         OutputAngle => rSinusAngle ,
40         ActScanTime => ) ;
41
42     SinusSignal (
43         Enable := gvl . bEnablePID ,
44         InOperation => ,
45         Error => ,
46         ErrorID => ,
47         ErrorIdent => ,
48         Angle := rSinusAngle ,
49         CurveType := 1 ,
50         Duty := gvl . rSinusDuty ,
51         DutyRamp := 25 ,
52         Rounding := ,
53         RoundingRamp := ,

```

```
54         OutputValue => rSinusOut ,
55         ActScanTime => ) ;
56
57         // Sinus output to ball position control PID;
58         gvl.rSinusOutputPID := ( rSinusOut ) * gvl.rAmplitude + 330 ;
59
60         //-----
61         // Low pass filter for filtering ball position feedback signal
62         LP_Filter_Position_Sensor (
63             Enable := gvl.bEnablePID ,
64             InOperation => ,
65             Error => ,
66             ErrorID => ,
67             ErrorIdent => ,
68             Pause := ,
69             Value := gvl.rBallFeedback ,
70             StartValue := ,
71             Coefficients := tIIR_Coff ,
72             InPause => ,
73             ActScanTime => ,
74             OutputValue => rLP_PosFilter_out ) ;
75         //-----
76         // PID function block for position closed loop
77         PID_Position (
78             Enable := gvl.bEnablePID ,
79             InOperation => gvl.bPIDActive ,
80             Error => gvl.bPIDError ,
81             ErrorID => ,
82             ErrorIdent => ,
83             Pause := ,
84             Preset := ,
85             Setpoint := gvl.rBallPosCmd ,
86             Feedback := rLP_PosFilter_out ,
87             PresetValue := GVL.rPresetVal ,
88             HighLimit := GVL.rCtrlMax ,
89             LowLimit := GVL.rCtrlMin ,
90             PControl := rPLocal ,
91             IControl := lILocal ,
92             DControl := lDLocal ,
93             bControl := gvl.rbCtrlPID ,
94             cControl := gvl.rcCtrlPID ,
95             InPause => ,
96             PresetAck => ,
97             ActScanTime => ,
98             HighLimitActive => ,
99             LowLimitActive => ,
100             ControlValue => gvl.rPIDOut ) ;
101         //-----
102         // Finding the derivative from the position signal to get velocity
103         // Calculate velocity using derivation, rdt is the cycle time 0.002s
```

```
104     gvl.rVelocity := ( ( rLP_PosFilter_out / 1000 ) - prev_pos ) / rdt ;
105
106     // update previous position
107     prev_pos := ( rLP_PosFilter_out / 1000 ) ;
108     //-----
109     // PID fucntion block for velocity closed loop
110     PID_Velocity (
111         Enable := gvl.bEnablePID ,
112         InOperation => gvl.bPIDActive ,
113         Error => ,
114         ErrorID => ,
115         ErrorIdent => ,
116         Pause := ,
117         Preset := ,
118         Setpoint := gvl.rPIDOUT , // rPIDOut_Filter
119         Feedback := gvl.rVelocity ,
120         PresetValue := GVL.rPresetVal ,
121         HighLimit := GVL.rCtrlMax ,
122         LowLimit := GVL.rCtrlMin ,
123         PControl := rPLocalVel ,
124         IControl := lILocalVel ,
125         DControl := lDLocalVel ,
126         bControl := gvl.rbCtrlPID ,
127         cControl := gvl.rcCtrlPID ,
128         InPause => ,
129         PresetAck => ,
130         ActScanTime => ,
131         HighLimitActive => ,
132         LowLimitActive => ,
133         ControlValue => gvl.rPIDCtrlOut ) ;
134
135
136
137
138
139
140
141
```