# XPlanar Startup Notes

Last updated by | Marc Wilkinson | Apr 14, 2023 at 10:30 AM EDT

Contents

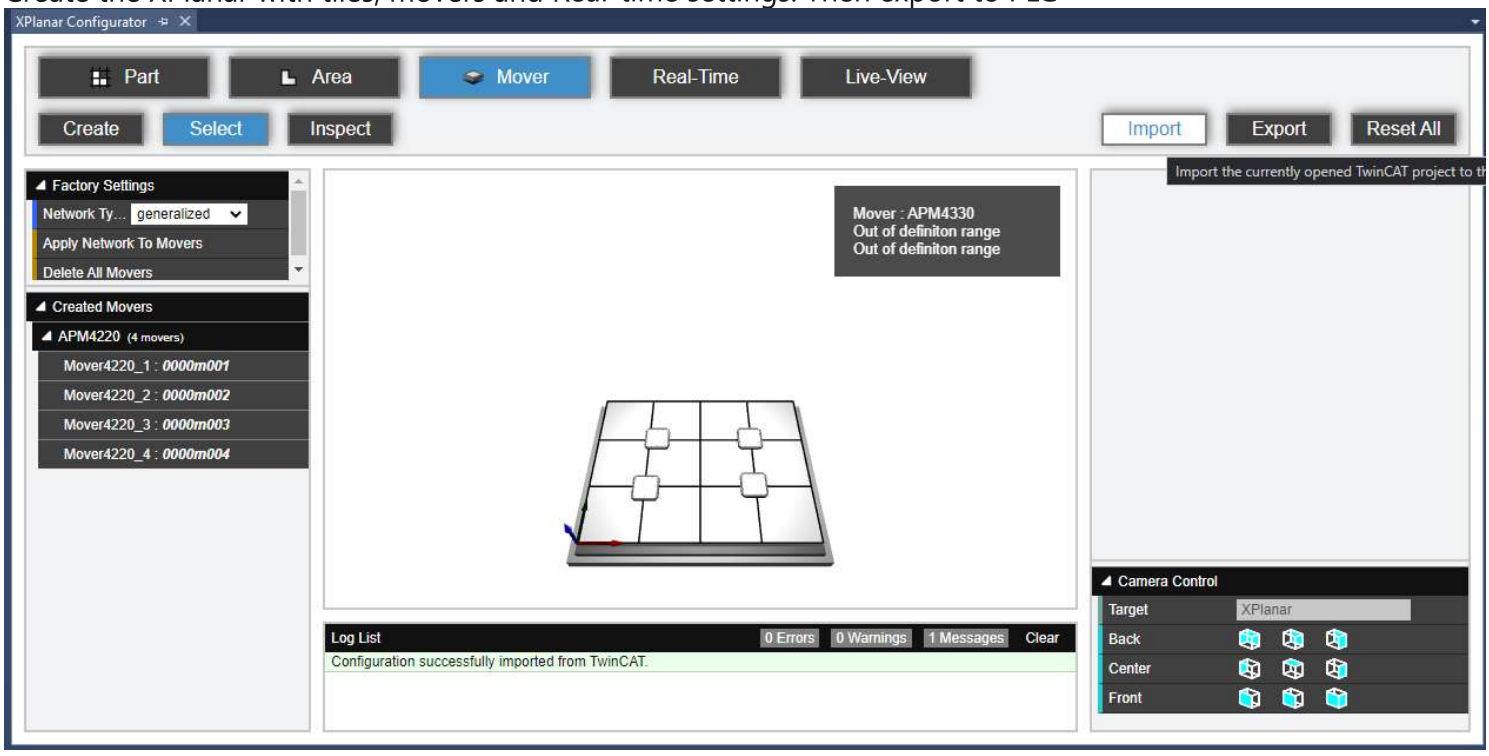


## Overview

This is a collection of notes and screenshots for working with the XPT XPlanar component.

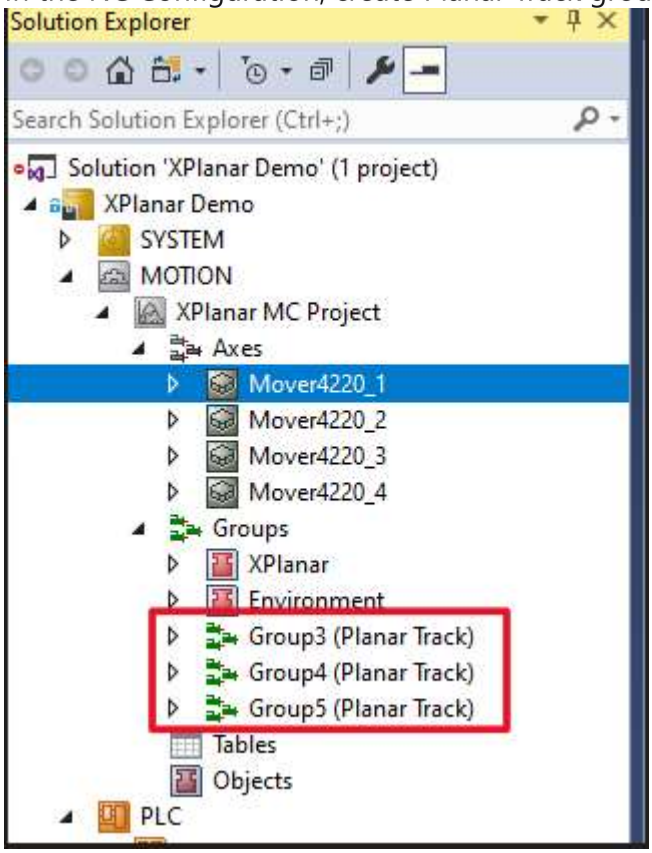
### Configuration with Config tool

Create the XPlanar with tiles, movers and Real-time settings. Then export to PLC



## Create Track Groups

In the NC Configuration, create Planar Track groups for each of your planned track segments



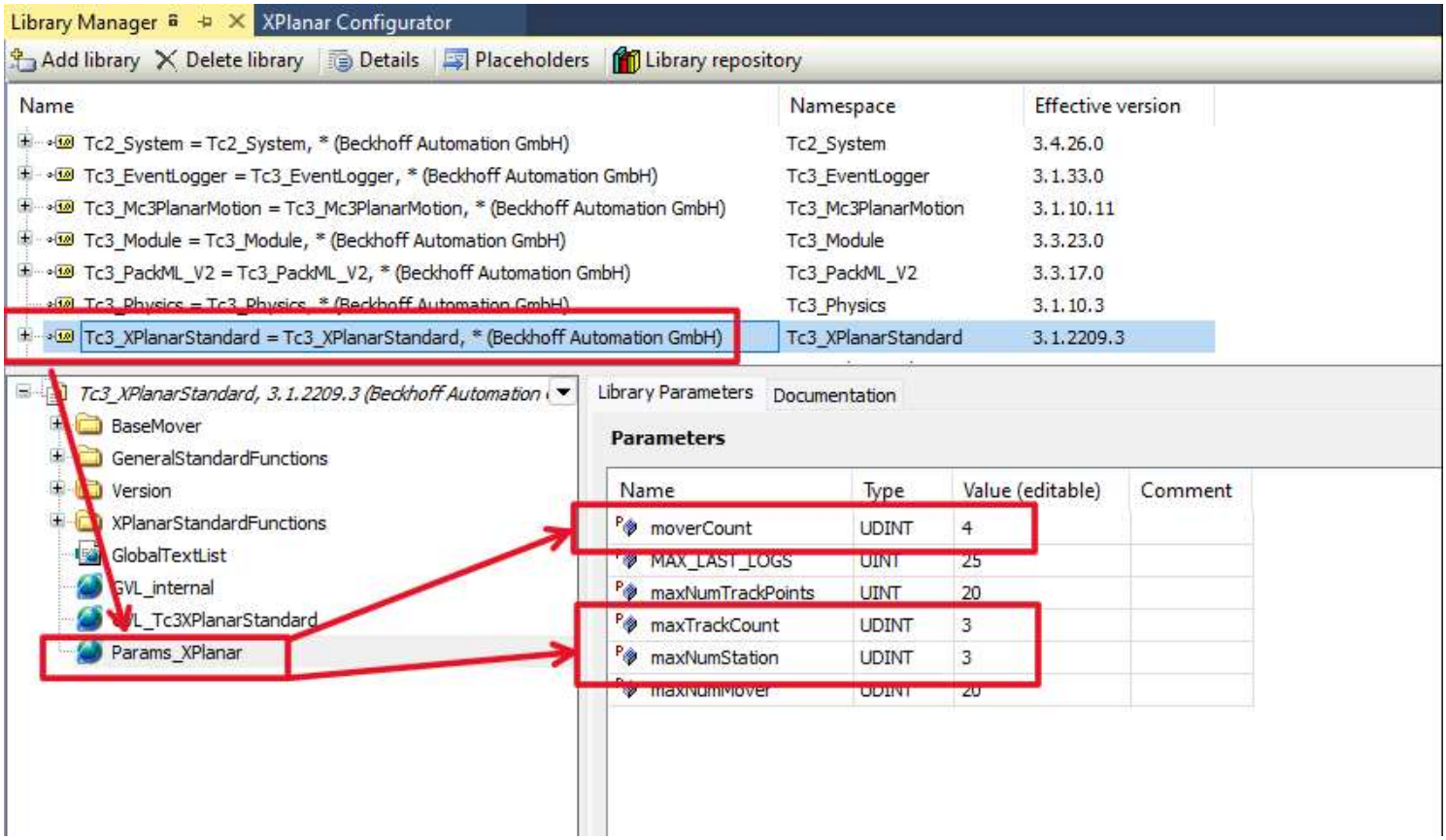

## PLC Program

Add a PLC Project with the following library references

- SPT Base Types
- SPT_XPlanar
- Tc3_PlanarMotion
- Tc3_Physics
- Tc3_XplanarStandard
- Tc3_PackML_V2 (if using SPT Framework)

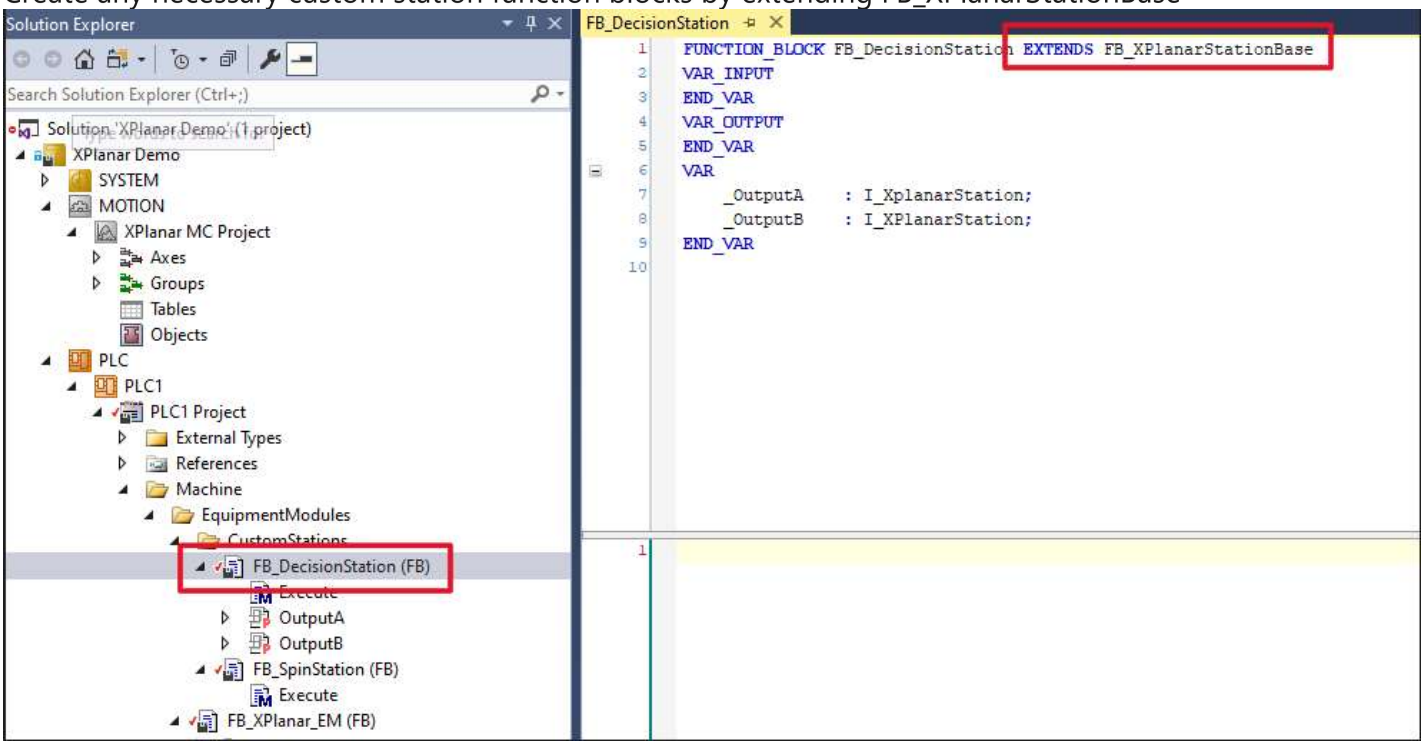
### Application Parameters

Set the number of movers, track segments, and stations in the Params_XPlanar list of the Tc3_XPlanarStandard library

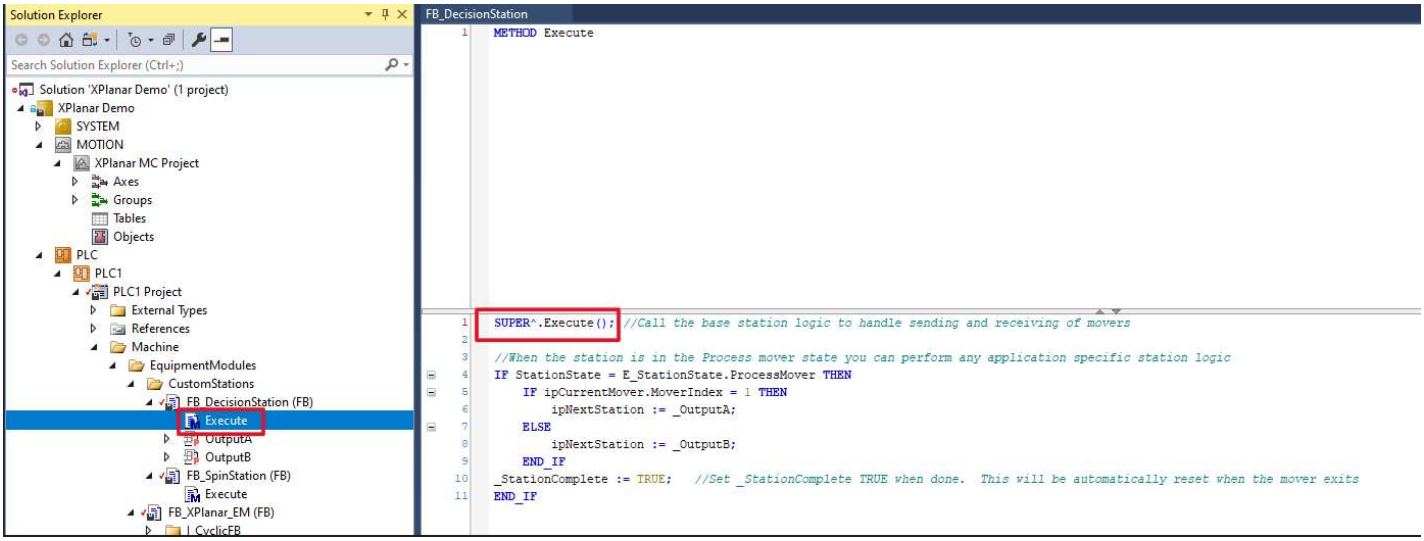



## Application Layer PLC logic

- Create any necessary custom station function blocks by extending FB_XPlanarStationBase




- For operation of a custom station, create and overwriting Execute method. Make sure to add a SUPER call at the beginning to handle base station functionality




- Declare instances of the main XPlanar component as well as instances for each station

```
    // Component definitions here
    XPlanarTable : FB_Component_Xplanar := (Name := 'XPlanarTable');

    //Station Components
    Station1     : FB_DecisionStation    := (Name := 'DecisionPoint');   //Declaration of a custom station
    Station2     : FB_SpinStation        := (Name := 'RotateStation');
    Station3     : FB_XPlanarStationBase := (Name := 'InspectStation');  //If no custom processing is needed simply declare an instance of the Station Base fb
```

- On initialization of the system setup all required station parameters. **For proper operation you must assign the stations to an interface in the station array**

```
//Set Required Process Station Initialization Parameters
Station1.Position.SetValuesXYC(120, 360, 0);         //Set the center of the station
Station1.TrackID     := ip.Tracks[1];                //Set the interface of the track segment the station resides on
Station1.StationType := E_StationType.DecisionPoint; //Change the type to Decision Point only if the station is acting as a divert and processing on the fly
Station1.OutputA     := Station2;                    //Set any custom station parameters
Station1.OutputB     := Station3;
ip.Stations[1]       := Station1;                    //Register the station with the interface array

Station2.Position.SetValuesXYC(600, 360, 0);
Station2.Size.x      := 120;                         //Changes the station size in the X plane
Station2.Size.y      := 120;                         //Changes the station size in the Y plane
Station2.TrackID     := ip.Tracks[2];
Station2.NextStation := Station1;                    //Set the interface for the next station of the mover
ip.Stations[2]       := Station2;

Station3.Position.SetValuesXYC(840.0, 360.0, 0);
Station3.Size.x      := 120;
Station3.Size.y      := 120;
Station3.TrackID     := ip.Tracks[3];
Station3.NextStation := Station1;
ip.Stations[3]       := Station3;
```

- On initialization of the system setup each track segment by adding all desired segment points and starting and ending segment interfaces if required

```
        ip.Tracks[2].ID := 2;                                                   //Set the ID property of the track segment
        IF ip.Tracks[2].TrackTable.ClearTrackTable() THEN                       //Clean up any stray points in the table
            ResultAddPoint := ip.Tracks[2].TrackTable.AddPoint(E_PointType.Line_Start,  //Configure track points
                                                               E_PointOption.none,
                                                               600.0,
                                                               480.0,
                                                               0.0);

            ResultAddPoint := ip.Tracks[2].TrackTable.AddPoint(E_PointType.Line_End,
                                                               E_PointOption.none,
                                                               600.0,
                                                               240.0,
                                                               0.0);

            ip.Tracks[2].TrackTable.P_CloseTheLoop   := FALSE;                  //Set the segment close the loop property
            ip.Tracks[2].TrackTable.P_StartFromTrack := ip.Tracks[1].std;       //If linking to a previous track segment set the interface
            ip.Tracks[2].TrackTable.P_EndAtTrack     := ip.Tracks[1].std;       //If linking to a downstream track segment set the interface
        END_IF
```

## Additional notes for operation

- The CyclicLogic method of the FB_Component_XPlanar function block must be called every PLC cycle (*This happens inherently when using the SPT Framework*)
- The CyclicLogic method for each Station function block must be called every PLC cycle (*This happens inherently when using the SPT Framework*)
- On starting, call the enable methods in the following order
  --EnableGroup()
  --EnableMovers()
  --EnableTracks()
- Once enabled you can perform any mover recovery actions required by the application.
- Once mover recovery is complete you can call the RecoverStations() method to do a first pass registration for moves that are in a station. This will squaure up the mover and join them to the appropriate track segment
- Once everything is initialized and enabled to "Run" the system simply call the Execute() method for each station. This can be done using a FOR loop like below.

```
3          FOR i := 1 TO Params_XPlanar.maxNumStation DO
4              ip.Stations[i].Execute();
5          END_FOR
6
```

- For any simple stations that just use the FB_XPlanarStationBase and require no additional processing, simple set the StationComplete property to TRUE to send the mover to the next station.

```
7      //Simulate the "Inspection Done" signal
8      InspectSimTimer(IN := ip.Stations[3].StationReady, PT := T#1S);
9      ip.Stations[3].StationComplete := InspectSimTimer.Q;
10
```

## Linking and Parameter Setup

### TcCOM

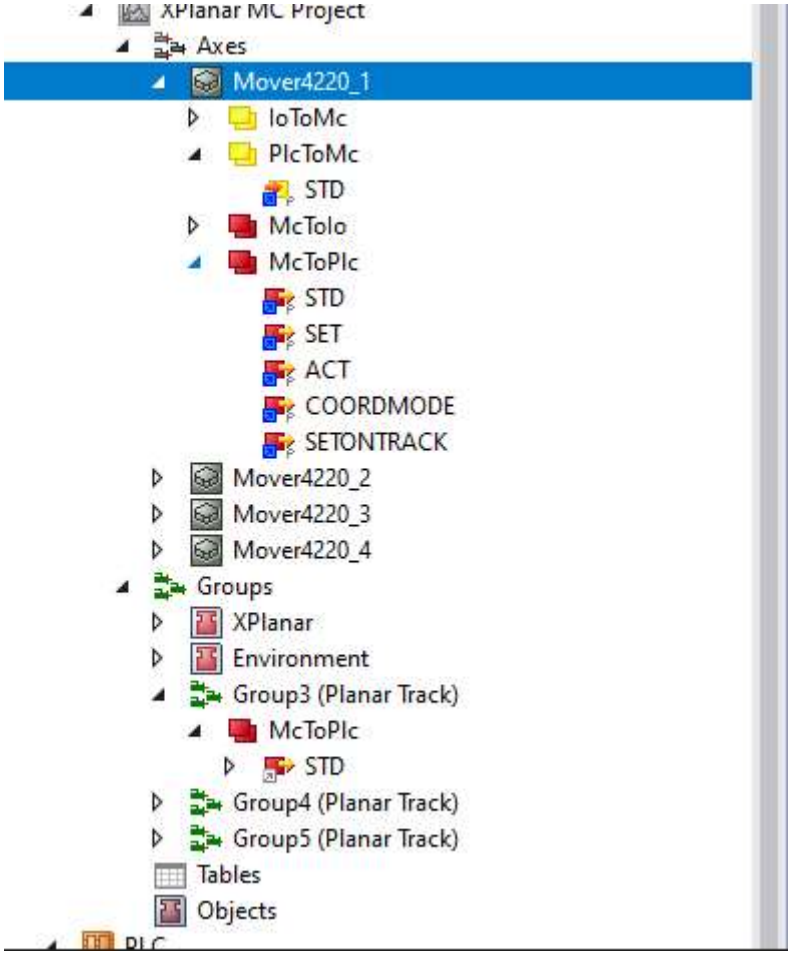
XPlanar - Set Operation Mode to BasicSimulation if simulating

### Motion

Click on Axis group and link Movers to PLC

### Groups

Link XPlanar and Environment To PLC
Add all the tracks needed and Link to PLC


XPlanar MC Project
Axes
Mover4220_1
IoToMc
PlcToMc
STD
McToIo
McToPlc
STD
SET
ACT
COORDMODE
SETONTRACK
Mover4220_2
Mover4220_3
Mover4220_4
Groups
XPlanar
Environment
Group3 (Planar Track)
McToPlc
STD
Group4 (Planar Track)
Group5 (Planar Track)
Tables
Objects