

# Twincat Library: Defining cyclic telegrams

## Application Note AN105

Version	Date	Editor	Comment
002	2012-11-27	mvx	Convert to new document style
003	2013-04-04	mvx	Modulo registers and time shift
004	2014-01-20	mvx	Warning on units

Document AN105\_TwinCAT-CyclicTelegrams\_EP  
 Version 004  
 Source Q:\doc\ApplicationNotes\AN105\_TwinCAT-CyclicTelegrams\  
 Destination T:\doc\ApplicationNotes  
 Owner mvx

Copyright © 2014	Triamec Motion AG	Phone +41 41 747 4040
Triamec Motion AG	Industriestrasse 49	Email <a href="mailto:info@triamec.com">info@triamec.com</a>
All rights reserved.	6300 Zug / Switzerland	Web <a href="http://www.triamec.com">www.triamec.com</a>

### Disclaimer

This document is delivered subject to the following conditions and restrictions:

- This document contains proprietary information belonging to Triamec Motion AG. Such information is supplied solely for the purpose of assisting users of Triamec products.
- The text and graphics included in this manual are for the purpose of illustration and reference only. The specifications on which they are based are subject to change without notice.
- Information in this document is subject to change without notice.

### Table of Contents

1 Target and Purpose.....1	4 Time shift.....2
2 PLC Code.....2	5 Restriction.....3
3 Modulo.....2	

## 1 Target and Purpose

The Triamec TwinCat library comes with basic sample codes for NCI and CNC. This application note describes additional functions available in this library.

The standard sample code sends position setpoint data with 10kHz to the drives. The actual position is only received with the (slow) state data in the task MAIN\_SLOW at typically 2ms.

If an application requires faster actual position information, it should use this code. The same method may be used to get any content of internal (dsp) registers of the drive.

## 2 PLC Code

Add this to GLOBAL\_VARIABLES\_TRIAMEC

```
publish1      : TL_publishSlave2Master;
```

Add this to the task MAIN\_SLOW

```
(* publish the actual position of an axis to the fast task *)
(* publish1.intern.ts := 0.001; only, if publish time should be slower than MAIN_FAST *)
(* be aware, that src1 is transmitted as Float40, the others are transmitted as Float32 *)
publish1.src1  := axis.MC_Axis.register.AxisSig.PositionController.ActualPosition;
publish1.station := axis.MC_Axis.station;
publish1.CallSlow(Trialink:= Trialink);
```

Add this to MAIN\_FAST

```
publish1.CallFast( Trialink:= Trialink );
(* result in publish1.out.val1 *)
```

Be aware, that any position or velocity data will be shown in the units of the drive. There is no correction with axis.Config.GearFactor in this case.

## 3 Modulo

If one of the registers being read uses modulo mode consider the following:

- Only the first register (src1) should be used for modulo.
- Specify the modulo value in the publisher using  
publish.mod\_wrap1 := PI\*2;  
with the axis scale that is used in the drive configuration, used with the TAM System Explorer. There is no correction with GearFactor here.

## 4 Time shift

The data received corresponds to the timestamp at the last TwinCAT MainFast tick, i.e., it is delayed by one tick. The data may be shifted some using the parameter

- publish1.out.ShiftTicks
- with a recommended range between -1.0 and +1.0

Be aware that using values larger than 0.0 may result in extrapolation.

The default interpolation and extrapolation mode is set to linear. It may be extended to quadratic which uses the three last data points:

- `publish1.out.interpolator[1].mode := InterpolPol2_XXX;`
- for the register `src1`

## 5 Restriction

Be aware, that receiving data over PCI consumes much more CPU time than sending. It is not recommended to use more than 5 fast loggers at 5kHz sampling rate.

If the information is not required at a high update rate, use the register read method instead, which polls data, see application note AN109.