Exercise 8

Contents

Introduction

In this exercise you will use the TwinCAT scope and Matlab to analyze the output of a simulated first order lag element that is feed with different input signals .

Exercises

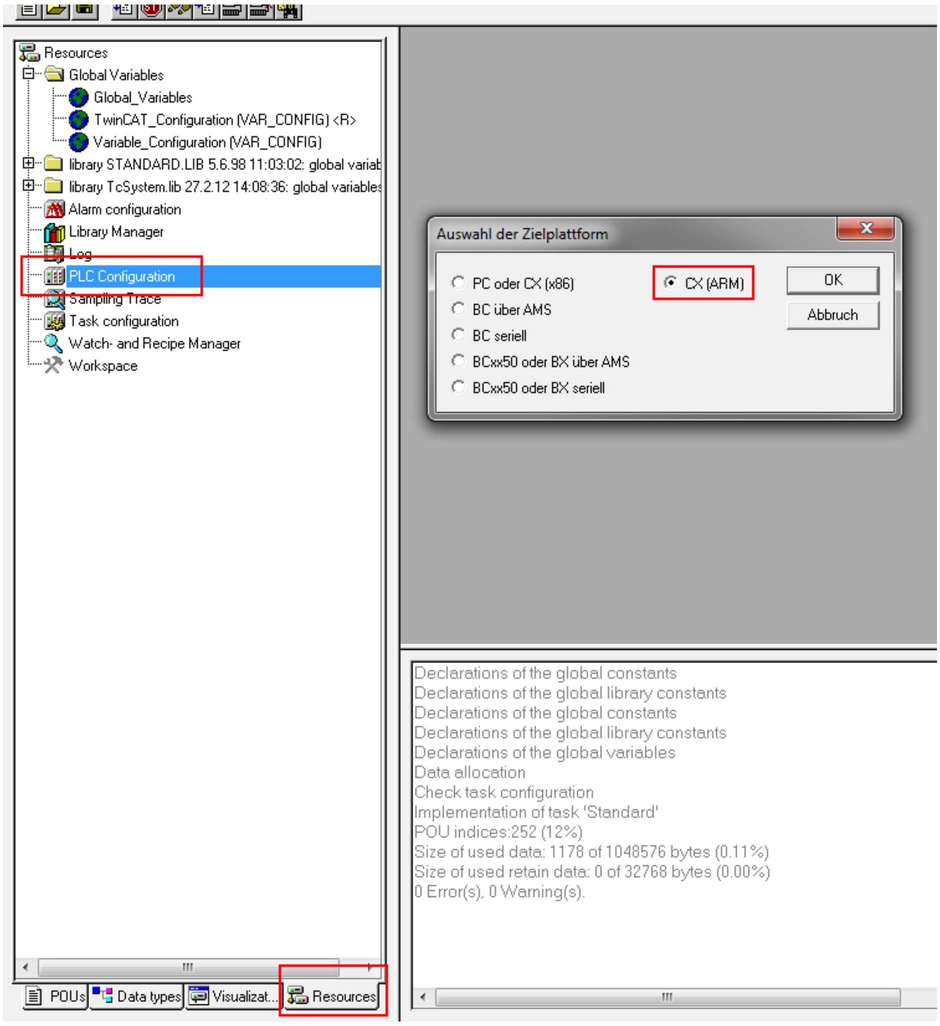
1. Connect PC and PLC

To establish a connection between your PC and the PLC work through the following topics of the "TwinCAT Guide":

1. Power the PLC
2. Establish connection
3. Open System Manager
4. Select Target System

2. Download and build project

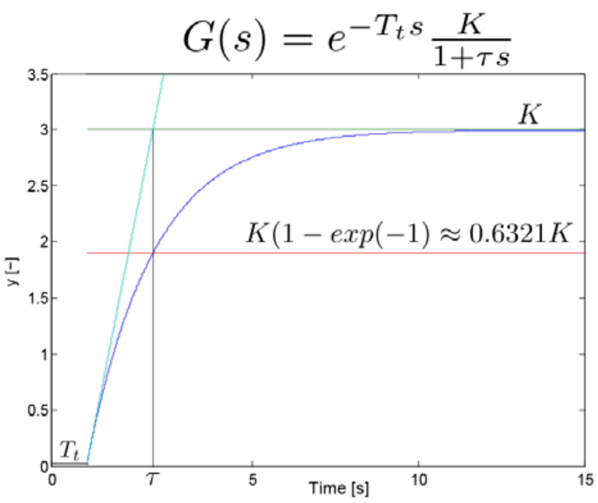
1. Download and save the exercise template project called Ex8_template.pro
2. Open the template project in "PLC Control".
3. If you have an ARM PLC you need to change the PLC Configuration to CX (ARM)



4. Build the project.

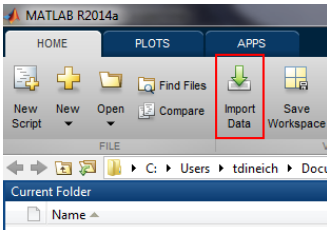
5. First order lag identification

The step response of a first order lag with delay looks like the following:

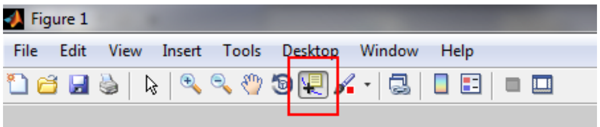


Provided that the measurement was conducted properly the values of the parameters *K*, τ and T_t can be determined (identified) from the resulting graph. This will be done for a simulated first order lag element:

1. Run the exercise template.
2. Open "TwinCAT Scope" and configure it to measure the following signals:
   a. Input of first order lag: MAIN_StepIdentification.myFOL.u_k
   b. Output of first order lag: MAIN_StepIdentification.myFOL.y_k
3. In "PLC Control" open the visualization "FirstOrderLagIdentification".
4. Record the signals in "TwinCAT Scope" and run the step signal in "PLC Control".
5. Export the measured signals from "TwinCAT Scope".
6. Open Matlab and import the measured signals.



7. Plot the measured signals in Matlab and determine the parameters *K*, τ and T_t. Hint: The data cursor is quite handy.



5. First order lag with different input signals

Let us now analyse the first order lag element with different input signals. The following behaviour should be observed:

1. The output signal should have the same frequency as the input signal.
2. The ouput signal should be proportional to the input signal (twice the input signal -> twice the output signal).
3. Over time the output signal behaves the same with the same input signal.
4. A finite input signal will result in a finite output signal.

We use 3 different input signals to proof the behaviour described above:

1. Sine signal for behaviour 1 to 3
2. Square signal for behaviour 1 to 3
3. Step signal for behaviour 4

The exercise project template has the visualization "FOL_InputSignals" to control the different input signals but the connections to the variables are missing:

1. Open visualization "FOL_InputSignals" in "PLC Control".
2. Connect the FOL ..., Sine ..., Square ... and Step ... input fields to their respective variables in programs MAIN_Sine, MAIN_Square and MAIN_Step.
3. Study how the "Enable ..." and "Apply settings" buttons control their respective MAIN_... program.

Now the different input signals have to be created. There are 3 function blocks that are used for this: SineGenerator, SquareGenerator and StepGenerator. Those function blocks are already used in the MAIN_... programs and hooked up to separate first order lag elements. They still need to be completed thou.

Complete the sine generator:

1. Open SineGenerator and create a sine signal according to the following formula:
   $$sine(t) = amplitude * SIN(2 * \pi * frequency) + offset$$
   Hint: The value of π is available as constant PI
2. Run the program and check with "TwinCAT Scope" if you get the expected behaviour of the first order lag element in MAIN_Sine (u_k is the input signal and y_k is the ouput signal). Repeat with different parameter settings.

Complete the square generator:

1. Open SquareGenerator and create a square signal with the given input variables.
2. Run the program and check with "TwinCAT Scope" if you get the expected behaviour of the first order lag element in MAIN_Square (u_k is the input signal and y_k is the ouput signal). It should look like loading and unloading of a capacitor. Repeat with different parameter settings.

Complete the step generator:

1. Open StepGenerator and create a step signal according to the following formula:
   $$step(t) = \begin{cases} t < timeTrigger: levelLow \\ t \geq timeTrigger: levelHigh \end{cases}$$
   The input variable reset should bring the step signal back to 0 so it can be triggered again.
2. Run the program and check with "TwinCAT Scope" if you get the expected behaviour of the first order lag element in MAIN_Step (u_k is the input signal and y_k is the ouput signal). Repeat with different parameter settings.