PLCopen helps the user with User Guidelines for Motion Control

- Update User Guidelines
- Getting Started
- Programming example: Labeling machine
PLCopen helps the user with User Guidelines for Motion Control

PLCopen has contributed to the integration of logic and motion control in one platform. With the logic based on the worldwide standard IEC 61131-3, the motion control functionality builds on top of the common features of this, especially the function block concepts. The PLCopen motion control specification is implemented on a wealth of different platforms from numerous suppliers, giving the user a broad offering to choose from while using the same knowledge and understanding.

With the update and integration of the first parts of the PLCopen specifications on motion control, the third part containing the user guidelines needed to reflect the new features. With the new release, the set of PLCopen motion control specifications is now made up of the following six cooperating parts:

- Part 1 - with the basic functionalities
- Part 2 - being integrated in Part 1
- Part 3 - the update part with the User Guidelines
- Part 4 for the interpolated motion control
- Part 5 with the homing procedures
- Part 6 with additional functionalities for hydraulics

All these specifications are available for download from the PLCopen website www.PLCopen.org under section ‘TC2 – Function Blocks’. PLCopen now enhanced the User Guidelines, being the 3rd part of the suite of specifications. With this it gives the user examples on how to use the motion control functions blocks to create higher level functionalities. These functionalities than can be made available in companywide libraries for better re-use. Also it shows how it combines with other specifications such as the OMAC PackAL, the Application Layer for packaging machine functionalities.

This article shows a basic example as described in this part.
Representation of the program

For the PLCopen Motion Control Function Blocks there is a graphical representation used for clarification. Of course, this also can be represented in a textual form. The following example shows how this can look like for the MC_MoveAbsolute function block.

Getting started

Before one can use an axis/drive combination they have to be initiated. This can be done via a simple startup procedure using the combination of MC_Power, MC_Home and MC_MoveAbsolute. One needs one MC_Power per axis to enable the axis. For the absolute positioning MC_Home is needed to define the home position. (Note: MC_Home is not always needed, for instance with MC_MoveVelocity and a rotating axis). Now it is possible to position the axis, for instance via issuing MC_MoveAbsolute. The picture below shows the graphical representation of this combined functionality.

The referred axis is called here Axis1, and is linked to all 3 Function Blocks. MC_Power is enabled via setting ‘PowerEnable’. Thereafter ‘Start’ is set to execute MC_Home. After the homing position is reached the ‘Done’ output of MC_Home is set. This executes the MC_MoveAbsolute Function Block which moves to the set position. The output ‘Ready’ will be set when the position is reached.

The homing part of the above procedure can be extended via the functionalities as defined in Part 5 – Homing Procedures. With this set of building blocks one can define a specific homing procedure per axis or per machine.

This basic functionality can be extended upfront with the use of the MC_ReadAxisInfo Function Block. In this way one can check if the communication to the drive is established, and can use the output ‘ReadyForPowerOn’ to enable MC_Power. Feedback is given via the output ‘PowerOn’ which can be the trigger to start MC_Home. This is then reflected in the output ‘IsHomed’ which can then be used to start the absolute movement.

Application example: Labeling machine

The task is to place a label at a particular position on a product.

The application has two motors, one to feed the product via a conveyor belt, the other to feed the labels and to place the labels on the products. The labeling process is triggered by a position detection sensor. From the detection of the product to the start of the labeling movement, there is a delay depending on the velocity of the conveyor, the position of the sensor and the position of the label on the product.

Programming example

This example shows a way to solve this task. Both axes move with the same velocity at the time the label is put on the product. The delay for TON is calculated from the sensor distance and the velocity of the conveyor. After a labeling step the motor for the label drive stops and waits for the next trigger, while the conveyor continuously moves.

Possible Improvements

Although this principle is working, there are some possibilities to improve the functionalities and performance, allowing for faster and more precise machines.

- Compensate for the drift of the label position as a result of the sum of incremental errors.
- A fast touch-probe input to detect the start position of the product more precisely.
- A MC_CamIn or MC_GearInPos function to synchronize the label and product position in
Part 3 promotes the creation of own libraries with user derived function blocks, increasing the re-use of tested software throughout the organization. To support this, it includes more examples. Not only application oriented examples like for warehousing and capping, but also different jogging and inching examples, registration to line up the packaging material, axes interlock to link two axes, the flying shear to cut moving material, the use of 3 segment CAM profiles and master engine, helping to decompose and structure the application software for a machine.

In addition it links to the work done within the OMAC organization and shows the specified winding/unwinding functionality.

The full specification can be downloaded from the PLCopen website www.PLCopen.org. Additional examples are welcomed at PLCopen to share with the community.