WAGO I/O SYSTEM 750

Using the Stepper_02.lib for the module 750-670 and 750-671

Application note

A114900, English
Version 1.0.6

WAGO®
INNOVATIVE CONNECTIONS

WAGO Kontakttechnik GmbH & Co. KG
Hansastraße 27
D-32423 Minden

Phone: +49 (0) 571/8 87 – 0
Fax: +49 (0) 571/8 87 – 1 69

E-Mail: info@wago.com

Web: http://www.wago.com

Technical Support
Phone: +49 (0) 571/8 87 – 5 55
Fax: +49 (0) 571/8 87 – 85 55

E-Mail: support@wago.com

Every conceivable measure has been taken to ensure the correctness and completeness of this documentation. However, as errors can never be fully excluded we would appreciate any information or ideas at any time.

We wish to point out that the software and hardware terms as well as the trademarks of companies used and/or mentioned in the present manual are generally trademark or patent protected.

TABLE OF CONTENTS

1 Important comments

To ensure fast installation and start-up of the units described in this manual, we strongly recommend that the following information and explanation is carefully read and adhered to.

1.1 Legal principles

1.1.1 Copyright

This manual is copyrighted, together with all figures and illustrations contained therein. Any use of this manual which infringes the copyright provisions stipulated herein, is not permitted. Reproduction, translation and electronic and photo-technical archiving and amendments require the written consent of WAGO Kontakttechnik GmbH& Co. KG. Non-observance will entail the right of claims for damages.

1.1.2 Personnel qualification

The use of the product detailed in this manual is exclusively geared to specialists having qualifications in PLC programming, electrical specialists or persons instructed by electrical specialists who are also familiar with the valid standards. WAGO Kontakttechnik GmbH& Co. KG declines all liability resulting from improper action and damage to WAGO products and third party products due to non-observance of the information contained in this manual.

1.1.3 Intended use

For each individual application, the components supplied are to work with a dedicated hardware and software configuration. Modifications are only admitted within the framework of the possibilities documented in the manuals. All other changes to the hardware and/or software and the non-conforming use of the components entail the exclusion of liability on part of WAGO Kontakttechnik GmbH& Co. KG.

Please direct any requirements pertaining to a modified and/or new hardware or software configuration directly to WAGO Kontakttechnik GmbH& Co. KG.

1.2 Range of validity

This application note is based on the stated hardware and software of the specific manufacturer as well as the correspondent documentation. This application note is therefore only valid for the described installation.

New hardware and software versions may need to be handled differently. Please note the detailed description in the specific manuals.

1.3 Symbols

Danger
Always observe this information to protect persons from injury.

Warning
Always observe this information to prevent damage to the device.

Attention
Marginal conditions must always be observed to ensure smooth operation.

ESD (Electrostatic Discharge)
Warning of damage to the components by electrostatic discharge. Observe the precautionary measure for handling components at risk.

Note
Routines or advice for efficient use of the device and software optimisation.

More information
References to additional literature, manuals, data sheets and INTERNET pages

2 Description

This document illustrates how to use the Stepper_02 library to handle the stepper modules 750-670 and 750-671.

2.1 Reference Material

The following hardware and software components were used to test this application note:

Assigned hardware components	Producer	Type
Ethernet-Controller	WAGO	750-841
Stepper Module (Indexer)	WAGO	750-670 (SW02HW02)
Stepper Module	WAGO	750-671 (SW02HW02)
Stepper_02.lib	WAGO	Version 2.2.1
Stepper with integrated power amplifier	Sonceboz	
PowerMax II	Pacific Scientific	1,8˚Step Motor

3 Wiring

The wiring of the stepper module is depending on the module as well as on the drive or drive interface.

3.1 Module 750-670

The following wiring assumes a 5 Volt single ended drive interface of the stepper power amplifier:

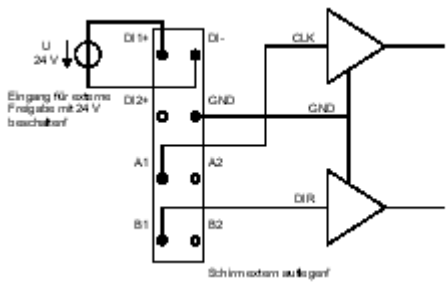
Input Ch1: Enable Drive(24VDC)

Input Ch2: Reference Switch(24VDC)

Ch-: Common ground for Ch1 and Ch2

Output A1: Clock(5VDC)

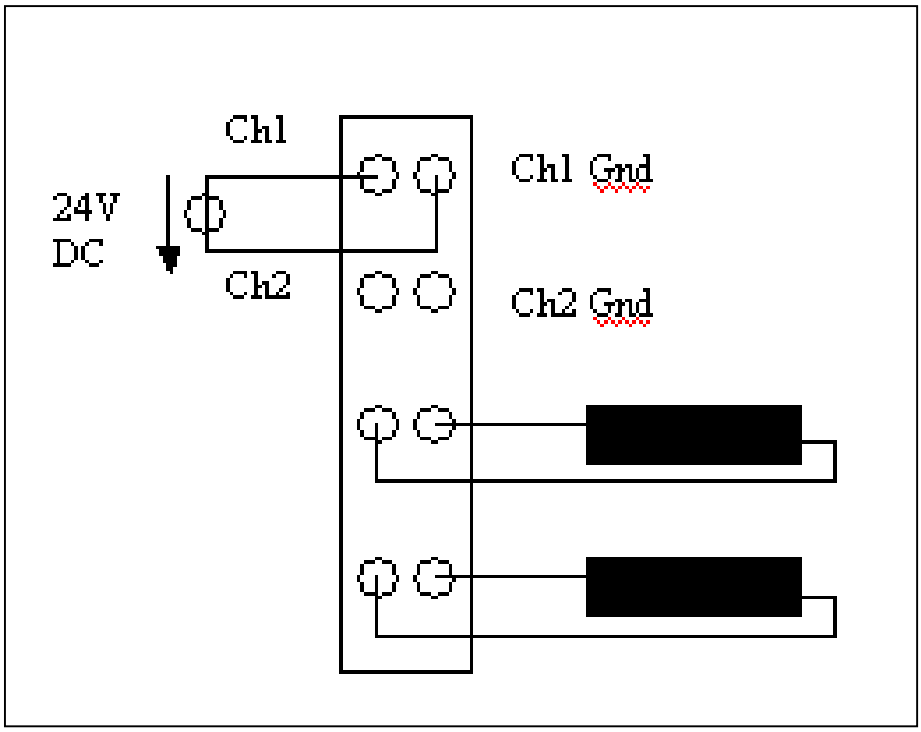
Output B1: Direction(5VDC)



For details on connecting a 24 VDC power amplifier, please reference the manual.

3.2 Module 750-671

Input Ch1+:	Enable Drive(24VDC)
Ch1-:	GND for channel 1
Input Ch2+:	Reference Switch(24VDC)
Ch2-:	GND for channel 2
Output A1,A2:	Coil A stepper drive
Output B1,B2:	Coil B stepper drive



4 General programming rules

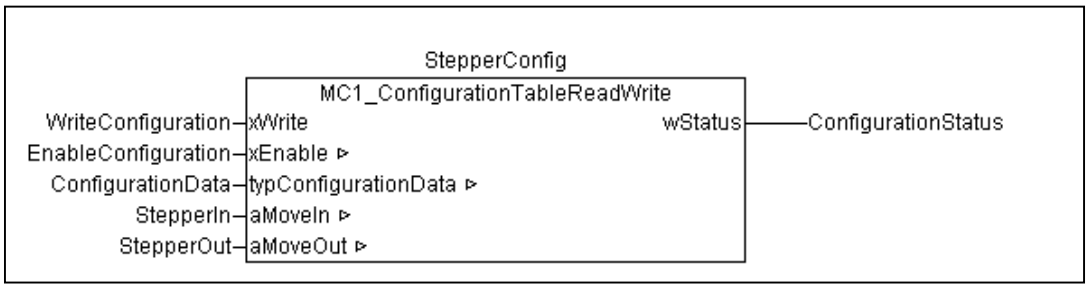
The library stepper_02.lib offers a wide range of function blocks to perform the different tasks of a stepper drive. The different function blocks need to be synchronized by the application program.


At each time only one function block is allowed to be active.

Due to the large number of function blocks in the library stepper_02, this application note will not describe each block in detail. This is done in the library description ML02600e.

5 Configuration

Using function block MC1_ConfigurationTableReadWrite



This block allows to reading and writing of the module's configuration values. All parameters are described in one structure (MC1_typConfiguration). This structure is subdivided in two parts. The parameters up to address 124 can be directly assigned. The parameters from address 128 up to 223 are pointers. They need the additional Bit I/O driver table(for details see stepper manual) for the source or destination information.

It is recommended to use this function block in the initialization phase to setup the modules parameters. If this block is used between positioning jobs, be careful since the actual position will be set to zero with each writing job.

If the function block MC1_StepperBasicControl is used, please make sure the input xStop is not set while the configuration block should run.

5.1 Homing parameters

The behavior of the homing job is mainly influenced by the parameters Reference_Offset, Reference_Mode, SetupAcceleration and SetupSpeed. Especially for the 750-671 module, the values for SetupAcceleration and SetupSpeed need to be increased due to the 64-folded microstepping.

Details on Reference_Mode:

0: Use Reference switch, reference to negative side of reference switch

1: Use Limit Switch for referencing

2: Use Reference switch, reference to positive side of reference switch

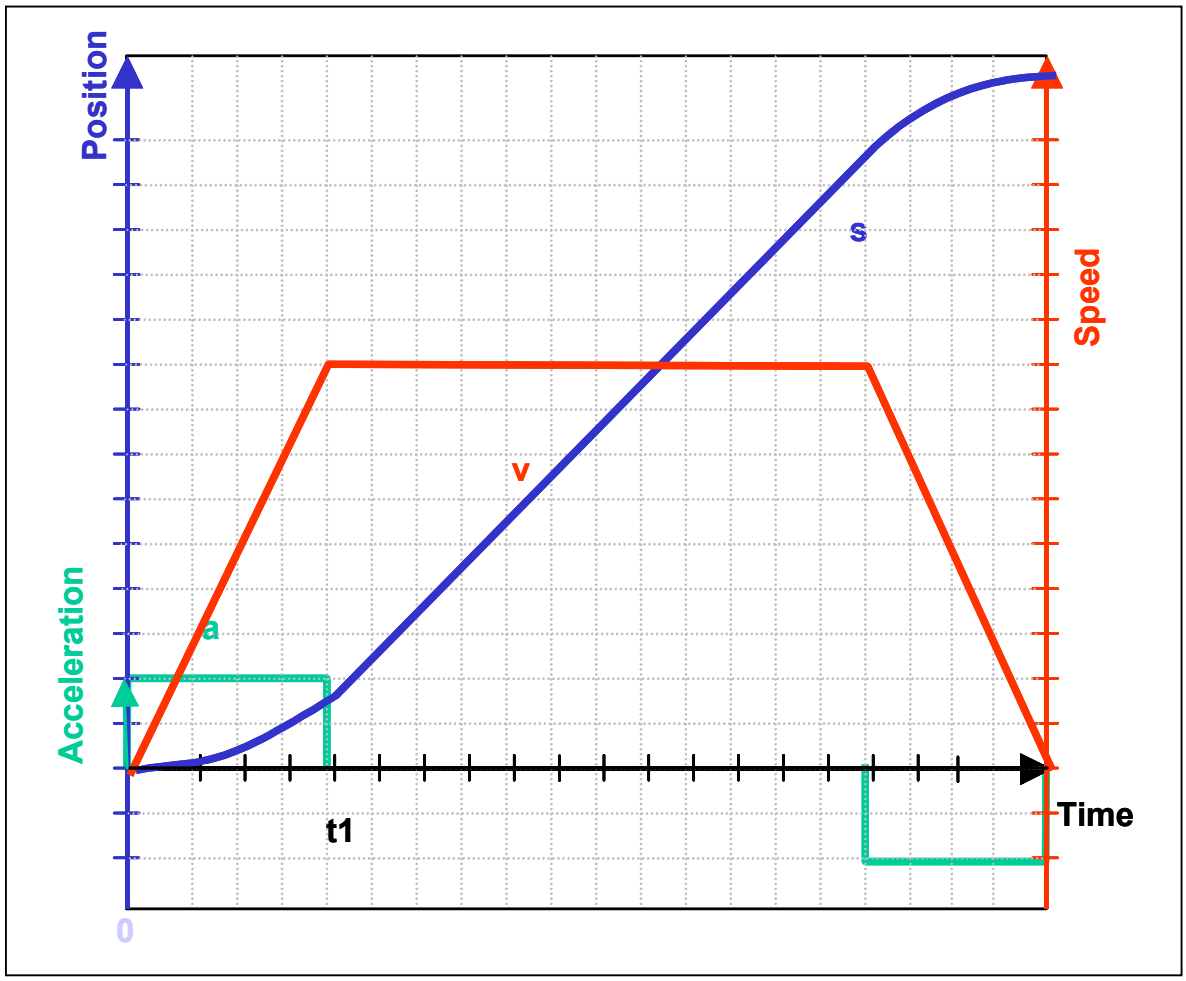
3: Use Limit Switch for referencing

5.2 Ramps

5.2.1 Constant acceleration

This acceleration profile is setup by:

ConfigurationData.Acceleration_Modes:=0;



Assuming the drive should reach v=v_max and the acceleration given by the process image ACCELERATION=32767 then:

t1 is calculated by:

v_max =2MHz/Freq_Div

a_max=32767*ACC_Fact/Freq_Div

t1=v_max/a_max=2MHz/(32767*ACC_Fact)

ACC_Fact	80	800	8000
t1	760ms	76ms	7,6ms

5.2.2 Linear acceleration

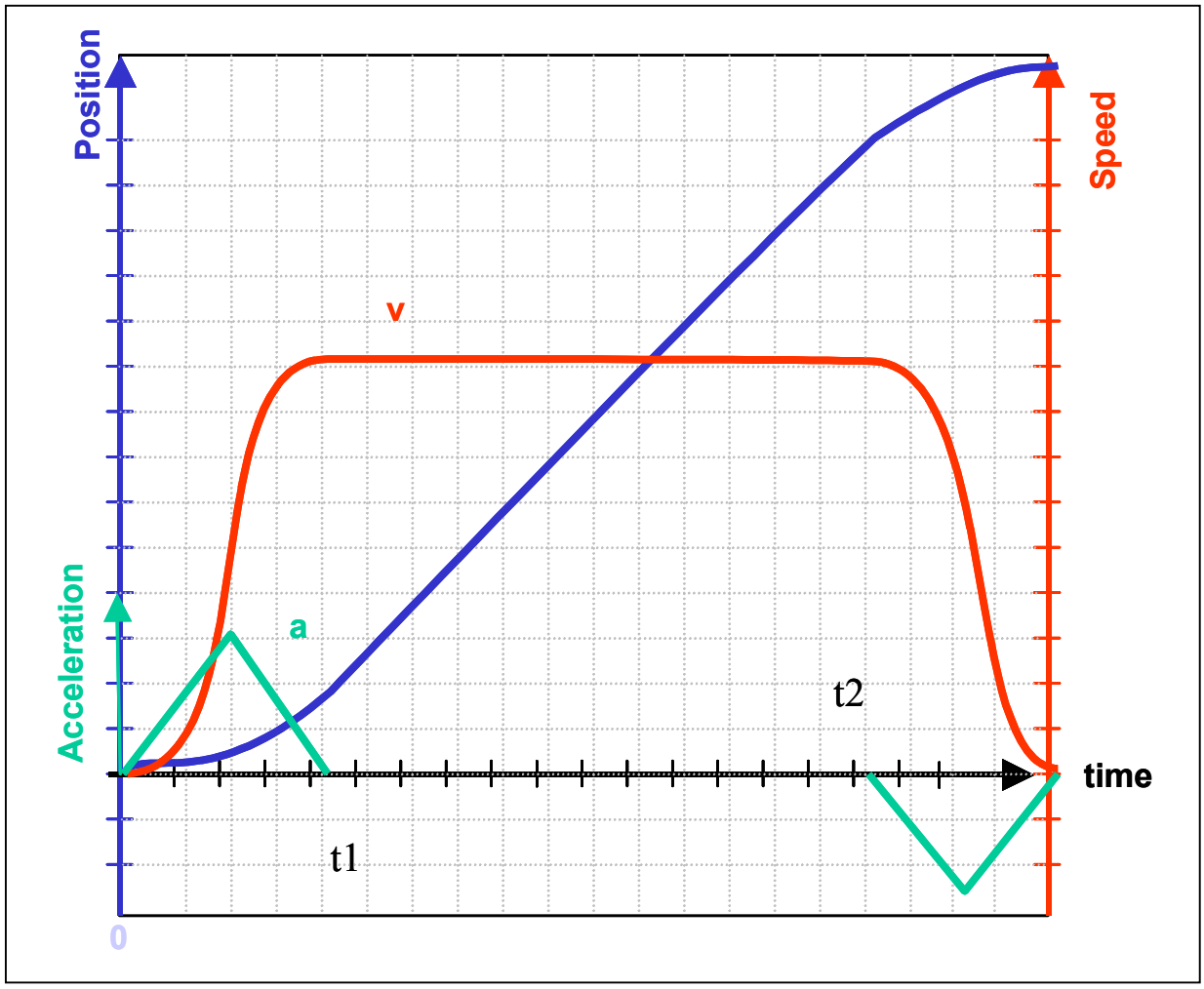
This acceleration profile is e.g. setup by:

ConfigurationData.Acceleration_Modes:=2#01010101

ConfigurationData.Acceleration_RampDown_Param:=2000
ConfigurationData.Acceleration_RampUp_Param:=2000

The parameter RampUp and RampDown define the duration of the increase or decrease in ms:

t1=2 sec
t2=2 sec



5.2.3 sin² acceleration

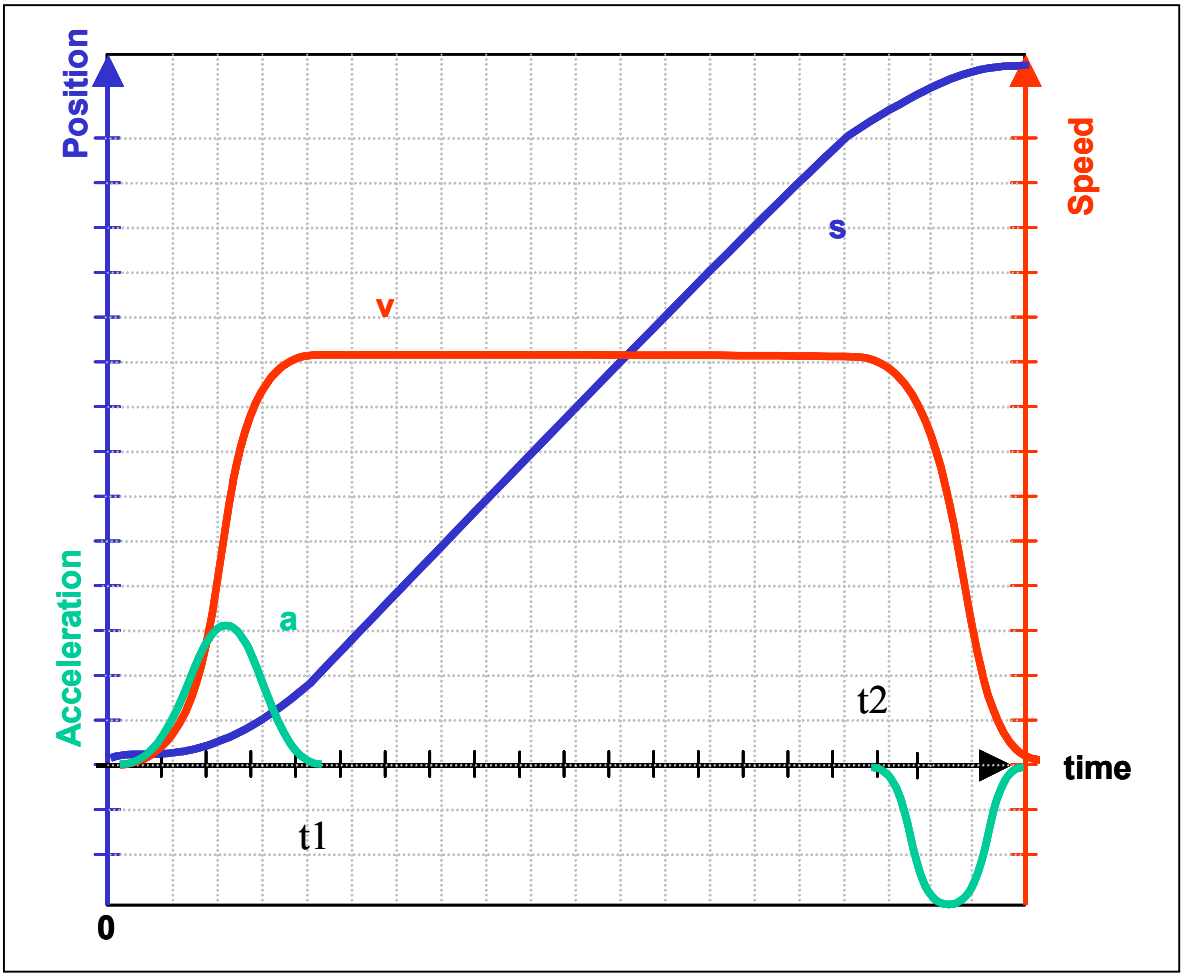
Example setting for sin² acceleration with 2 seconds to reach target speed and 2 seconds to stop.

ConfigurationData.Acceleration_Modes:= 2#01100110

ConfigurationData.Acceleration_RampDown_Param:=2000
ConfigurationData.Acceleration_RampUp_Param:=2000

The parameter RampUp and RampDown define the duration of the increase or decrease in ms:

t1=2 sec
t2=2 sec



5.3 Linking of Bits

The stepper module allows to change the functionality of e.g. the inputs or Control Bit`s. The default setting is:

Stop1_N	Input 1	(Ptr_Stop1_N:= 16#30)
Set_Reference:	Input 2	(Ptr_Set_Reference:= 16#31)
LimitSwitch_Pos	Control 3.4	(Ptr_LimitSwitch_Pos:= 16#54)
LimitSwitch_Neg	Control 3.5	(Ptr_LimitSwitch_Neg:= 16#55)

5.3.1 Example 1

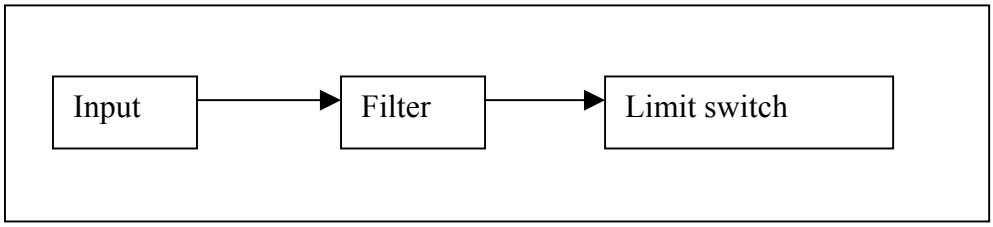
Link the positive limit switch to the first digital input.

LimitSwitch_Pos	Input 1	(Ptr_LimitSwitch_Pos:= 16#30)
Stop1_N	TRUE	(Ptr_Stop1_N:= 16#01)

5.3.2 Example 2

Use the inputs for the positive and negative limit switch with an active high level(inverting of input logic).

This example makes use of the Filter functionality. Inverting will be achieved by setting the Filter function to “Inversion”.



Ptr_FILT1:=0x30	(linking input 1 to filter 1)
Ptr_FILT2:=0x31	(linking input 2 to filter 2)
Ptr_C3_LimitSwitch_Pos:=0xA8	(linking filter 1 output to positive limit switch)
Ptr_C3_LimitSwitch_Neg:=0xA9	(linking filter 2 output to positive limit switch)
Filter1_Function :=1	(set filter 1 function to “inverting”)
Filter2_Function :=1	(set filter 2 function to “inverting”)
Ptr_Set_Reference:=0	(deactivate reference switch)
Ptr_Stop1_N:=1	(set Stop1_N functionality to TRUE)

5.3.3 Current setting for module 750-671

Using the module 750-671 it is necessary to adapt the current to the motor. Therefore the configuration parameter

ConfigurationData.Current

needs to be assigned to the motor value.

5.3.3.1 Example

Assuming the motor current is 0.5A:

ConfigurationData.Current:=5

5.3.4 Calculating steps per revolution for module 750-671

Assuming a 1,8° stepper motor should be used with the module 750-671, the number of steps per revolution is calculated according to:

$$\frac{\frac{360°}{\text{Revolution}}}{\frac{1{,}8°}{\text{Full Step}}} \times 64 \frac{\text{Microsteps}}{\text{Full Step}} = 12800 \frac{\text{Steps}}{\text{Revolution}}$$

5.3.5 Calculating speed and acceleration values

An Excel sheet will help calculate the configuration values for Freq_Div and Acc_Fact.

Example: A motor should travel at 200 rpm. This speed should be reached within 500ms.

The motors step angle is 1.8°.

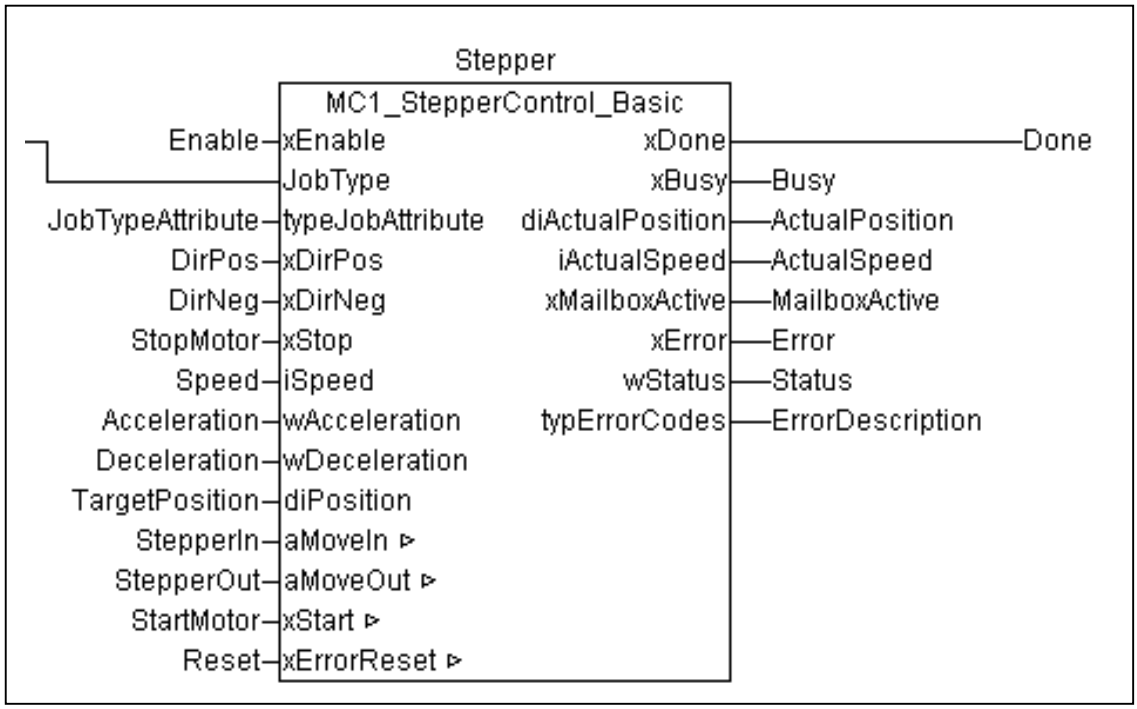
module	**750-671**			
module oscilator freq.	2000000	*Hz*		
speed quantisation range	25000	*levels*		
acc quantisation range	32767	*levels*		
microsteps/step	64	*usteps*		
motor				
deg/step	1,8	*deg*		
steps/revolution	200	*steps*		
microsteps/revolution	12800	*usteps*		
required				
max speed	**200,0**	*rpm*		
start-ramp time	**0,500**	*sec*		
result			min	max
Freq_Div	**47**		4	32767
Acc_fact	**122**		1	32767

Configuring Freq_div=47 and Acc_Fact=122 you have to apply

iSpeed=25000 and wAcceleration=32767 to the function block MC1_StepperControlBasic.

6 Positioning

Using function block MC1 _StepperControl_Basic



The function block MC1_StepperControl_Basic is used to run different positioning jobs. The input JobType determines the mode like Homing(2), Absolute Positioning(0),Relative Positioning(1),Jogging(3), VelocityControl(4) and drive Program(6).

If a stop is performed while a job is running the acceleration_Stop_Fast ramp will be used.

If the function block is used with the variable typeJobAttribute=0 the following table shows which values are taken for the different movements.

CFG: Values from configuration

	Homing	Jogging	MoveAbsolute	MoveRelative	VelocityControl
diPosition	CFG	CFG	TargetPosition	TargetPosition	Don`t care
iSpeed	CFG	CFG	Speed	Speed	Speed
wAcceleration	CFG	CFG	Acceleration	Acceleration	Acceleration
wDeceleration	CFG	CFG	Deceleration	Deceleration	Deceleration

For a detailed explanation of this function block please reference to the library description.

7 Drive program

To use the drive program capability of the module it is necessary to download a drive program first. The function block MC1_Drive_ProgramTable handles this download.

The drive program is stored in a byte array organized like:

Opcode	data_LSB	data	data_MSB	Description
16#25	16#20	16#4E	16#00	set velocity to 20.000
16#22	16#B8	16#0B	16#00	set acceleration to 3.000
16#04	16#02	16#00	16#00	Move_L to position 2 from position table
16#70	16#D0	16#07	16#00	wait 2000 ms

If an Opcode like Move_L is programmed, which makes use of a position table, an additional function block is necessary. The MC1_PositionTableReadWrite is used to download the appropriate position table.

A position table is a byte array with the positions stored as 32-bit values:

	LSB			MSB	Description
0	16#00	16#20	16#00	16#00	Position 0->8192
1	16#00	16#40	16#00	16#00	Position 1->16384
2	16#A0	16#86	16#01	16#00	Position 2->100.000
3	16#C0	16#D4	16#01	16#00	Position 3->120.000

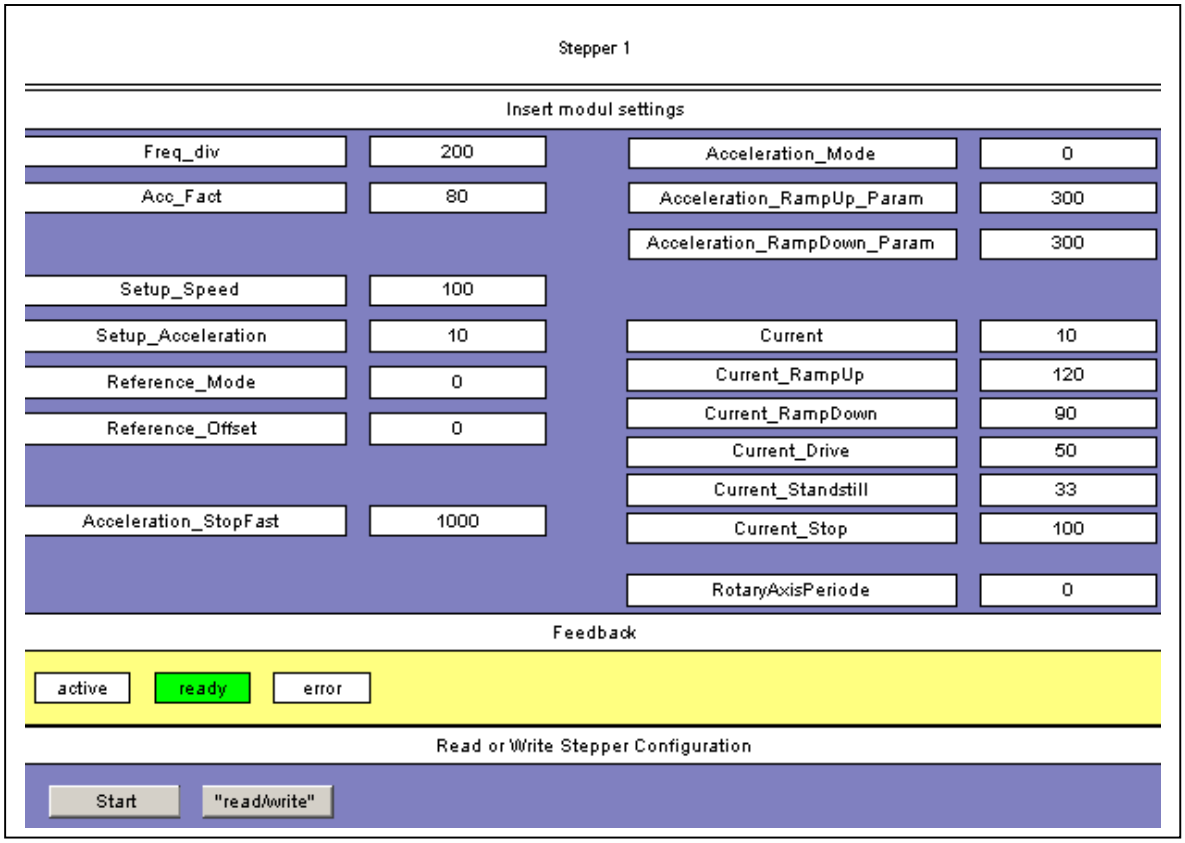
8 CAM functionality

Using the cam functionality needs an active cam table in the stepper module. The function block MC1_ConfigurationCamTable is used to download a cam table.

Cam`s	LSB		MSB	Description
16#03	16#FF	16#FF	16#FF	First position: needs to be negative CAM 1 high, CAM 2 high, CAM 3..8 low Out=2#0000_0011
16#02	16#64	16#00	16#00	Position: 100 CAM 1 low, CAM 2 high, CAM 3..8 low Out=2#0000_0010
16#01	16#C8	16#00	16#00	Position 200 CAM 1 high, CAM 2 low, CAM 3..8 low Out=2#0000_0001
16#03	16#2C	16#01	16#00	Position 300 CAM 1 high, CAM 2 high, CAM 3..8 low Out=2#0000_0011

9 Example

Running the example project Stepper_02_Example_Visu_v2.pro there are 2 different visualisations. The Configuration_Stepper1 visualisation is used to modify some of the most important module parameters



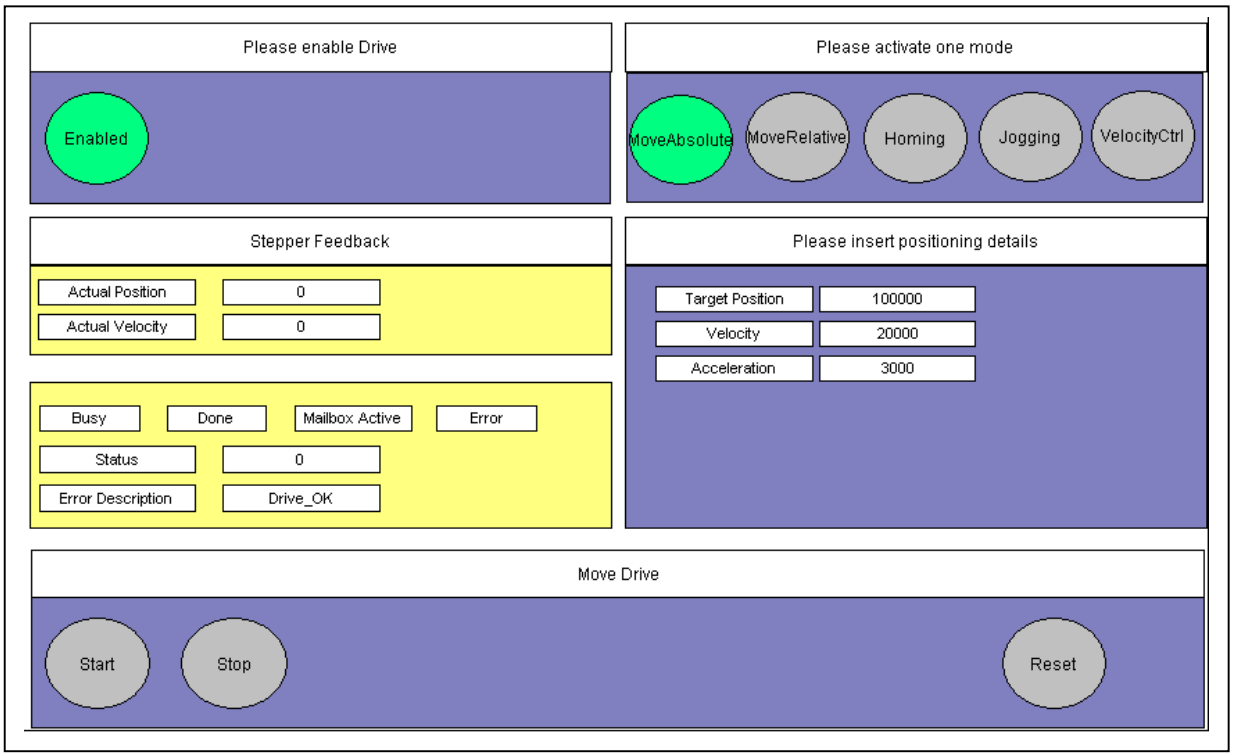
By just pressing the start button the actual configuration will be read.

By pressing the read/write button and then pressing the start button the configuration will be written.

Reading or writing will be indicated through the active flag becoming green.

If the job is done without error the ready flag will become green and error stays white.

The following figure shows the visualization for running the stepper.



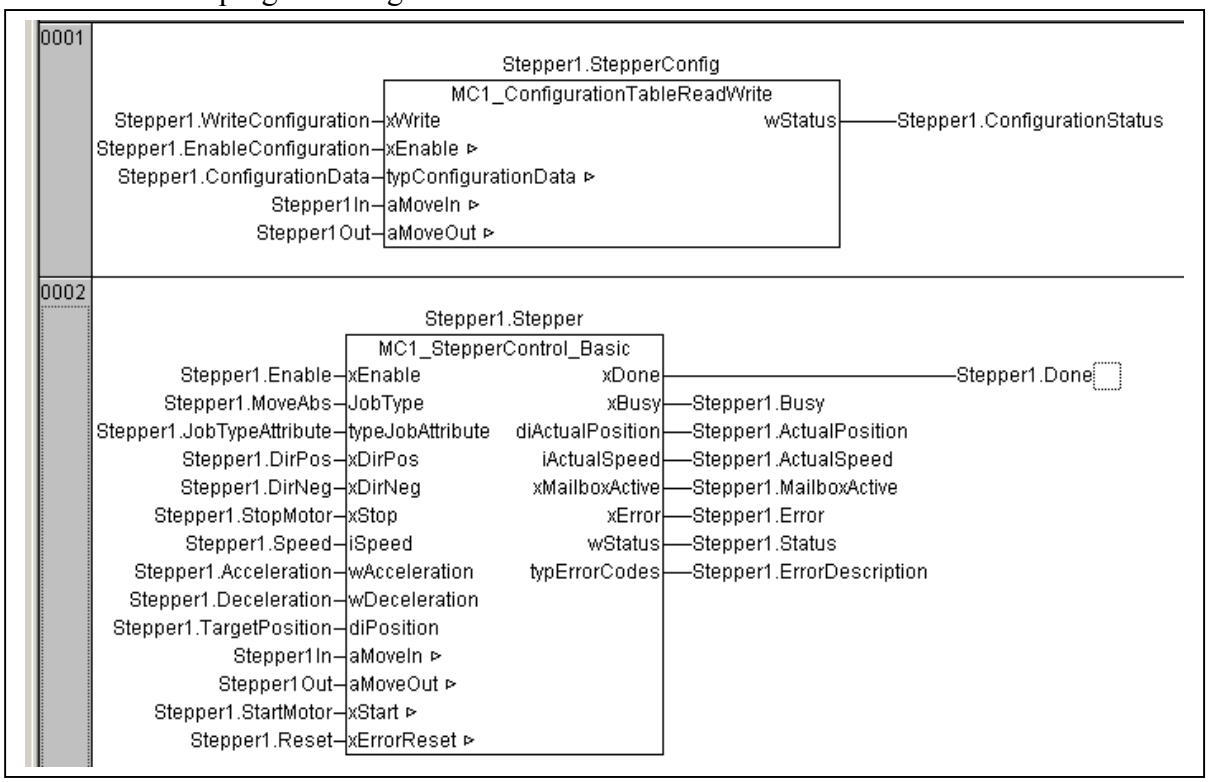
Basic steps of operating:
1)Enable the stepper
2)Press Reset if an error is indicated
2)Choose one operation mode, like MoveAbsolute
3)Press a button like Start to begin positioning

If the stepper feedback indicates an error a reset is necessary. The reset will only be executed if no operation mode is active.

If the modes Homing or Jogging are activated two additional buttons for positive and negative direction are available.

10 Using visualization elements from stepper_02.lib

There are two visualization elements available within the stepper_02.lib. They are based on a program designed like shown below:



Due to the placeholder concept it is easily possible to use this visualization elements for a second or third stepper.
Each stepper needs to have a variable of typ1 StepperVisu, e.g. stepper1, stepper2,etc.
Insert the visualization from the library and assign the placeholder to the appropriate stepper variable.

WAGO Kontakttechnik GmbH& Co. KG
Postfach 2880 • D-32385 Minden
Hansastraße 27 • D-32423 Minden
Telefon: 05 71/8 87 – 0
Telefax: 05 71/8 87 – 1 69
E-Mail: info@wago.com

Internet: http://www.wago.com