SIEMENS

SIMATIC
Easy Motion Control

Manual Edition 02/2003

Position-Controlled Positioning with Software

SIEMENS

SIMATIC

Easy Motion Control

Manual

Edition 02/2003
A5E00100694-02

Safety Guidelines

This manual contains notices intended to ensure personal safety, as well as to protect the products and connected equipment against damage. These notices are highlighted by the symbols shown below and graded according to severity by the following texts:

Danger
indicates that death, severe personal injury or substantial property damage will result if proper precautions are not taken.

Warning
indicates that death, severe personal injury or substantial property damage can result if proper precautions are not taken.

Caution
indicates that minor personal injury can result if proper precautions are not taken.

Caution
indicates that property damage can result if proper precautions are not taken.

Notice
draws your attention to particularly important information on the product, handling the product, or to a particular part of the documentation.

Qualified Personnel

Only **qualified personnel** should be allowed to install and work on this equipment. Qualified persons are defined as persons who are authorized to commission, to ground and to tag circuits, equipment, and systems in accordance with established safety practices and standards.

Correct Usage

Note the following:

Warning
This device and its components may only be used for the applications described in the catalog or the technical description, and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens.

This product can only function correctly and safely if it is transported, stored, set up, and installed correctly, and operated and maintained as recommended.

Trademarks

SIMATIC®, SIMATIC HMI® and SIMATIC NET® are registered trademarks of SIEMENS AG.

Third parties using for their own purposes any other names in this document which refer to trademarks might infringe upon the rights of the trademark owners.

Disclaimer of Liability

We have checked the contents of this manual for agreement with the hardware and software described. Since deviations cannot be precluded entirely, we cannot guarantee full agreement. However, the data in this manual are reviewed regularly and any necessary corrections included in subsequent editions. Suggestions for improvement are welcomed.

Siemens AG
Bereich Automation and Drives
Geschaeftsgebiet Industrial Automation Systems
Postfach 4848, D- 90327 Nuernberg

Siemens Aktiengesellschaft

A5E00100694-02

Excellence in Automation & Drives: Siemens

Preface

Purpose of the Manual

This manual provides you with a complete overview of programming with Easy Motion Control. It supports you during the installation and setting up of the software. It includes explanations of how to create a program and of the structure of user programs.

It is intended for persons who are responsible for the realization of control tasks using SIMATIC automation systems.

We recommend that you familiarize yourself with the Getting Started.

Required Experience

To understand the manual, you should have general experience of automation engineering.

You should also be familiar with working on computers or PC-type machines (for example programming devices).

Scope of the Manual

The manual is valid for the software package Easy Motion Control V2.0.

Further Support

If you have questions related to the use of the products which are not answered in this manual, please consult your Siemens representative in your local agency.

Training Centers

Siemens offers a number of training courses to familiarize you with the SIMATIC S7 automation system. Please contact your regional training center or our central training center in D 90327 Nuremberg, Germany for details:
Telephone: +49 (911) 895-3200.

http://www.sitrain.com/

A&D Technical Support

Worldwide, available 24 hours a day:

Worldwide (Nuernberg) **Technical Support** 24 hours a day, 365 days a year Phone: +49 (0) 180 5050-222 Fax: +49 (0) 180 5050-223 E-Mail: adsupport@siemens.com GMT: +1:00		
Europe / Africa (Nuernberg) **Authorization** Local time: Mon.-Fri. 8:00 to 17:00 Phone: +49 (0) 180 5050-222 Fax: +49 (0) 180 5050-223 E-Mail: adsupport@siemens.com GMT: +1:00	**United States** (Johnson City) **Technical Support and Authorization** Local time: Mon.-Fri. 8:00 to 17:00 Phone: +1 (0) 423 262 2522 Fax: +1 (0) 423 262 2289 E-Mail: simatic.hotline@sea.siemens.com GMT: -5:00	**Asia / Australia** (Beijing) **Technical Support and Authorization** Local time: Mon.-Fri. 8:00 to 17:00 Phone: +86 10 64 75 75 75 Fax: +86 10 64 74 74 74 E-Mail: adsupport.asia@siemens.com GMT: +8:00
The languages of the SIMATIC Hotlines and the authorization hotline are generally German and English.		

Service & Support on the Internet

In addition to our documentation, we offer our Know-how online on the internet at:

http://www.ad.siemens.de/support

where you will find the following:

- Current Product Information leaflets, FAQs (Frequently Asked Questions), Downloads, Tips and Tricks.
- A newsletter giving you the most up-to-date information on our products.
- The Knowledge Manager helps you find the documents you need.
- Users and specialists from all over the world share information in the forum.
- Your local customer service representative for Automation & Drives in our customer service representative data bank.
- Information on field service, repairs, spare parts and more under "Services".

Contents

1 Product Overview

1.1 Overview

Easy Motion Control allows you to control the positioning of drives. Movement is controlled solely by software (function blocks) in the CPU. **No special hardware** is required for connection to the drives, although corresponding I/Os must of course be available in order to record actual positions and output speed setpoint values.

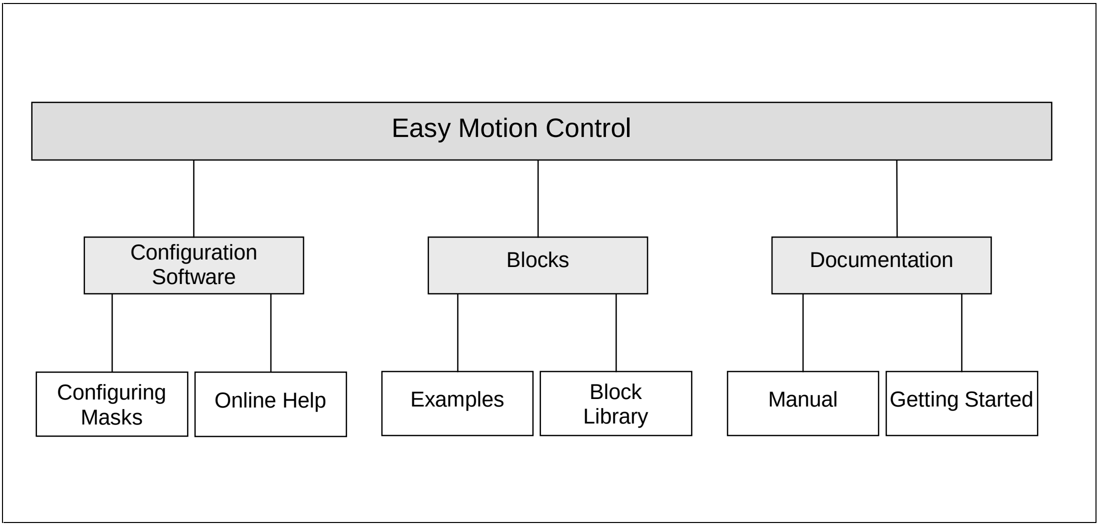
Easy Motion Control function blocks were developed according to the "Technical Specification V1.0" of **PLCopen**.

They assign parameters using a configuration software which also supports commissioning using test functions. All parameters of one axis are entered in one data block which from now on is referred to as axis DB

You can connect the drives to central I/Os or, via a DP master integrated into the CPU, to distributed I/Os.

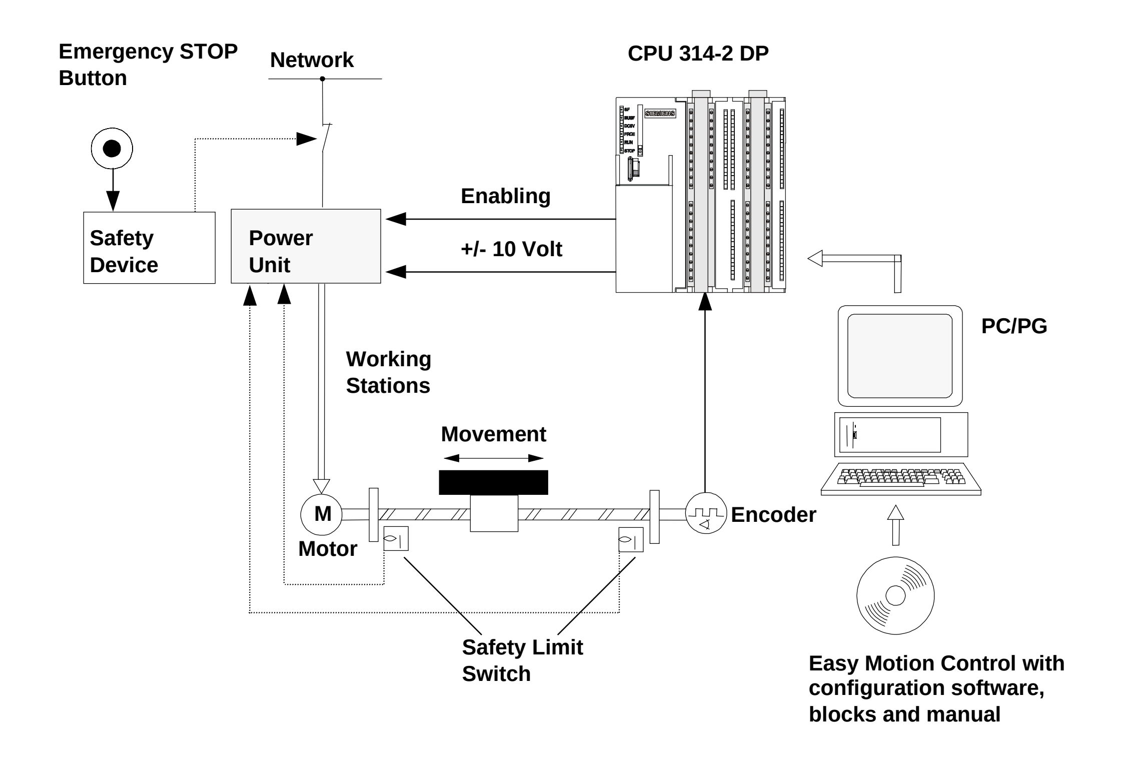
Perform the motion by interconnecting the corresponding function blocks and letting these run in the STEP 7 program.

Easy Motion Control components are shown in the following diagram.



1.2 Components for Positioning

The following diagram shows a configuration model with Easy Motion Control.



Safety device

When the safety device is actuated, the emergency STOP button disconnects the power supply.

Power unit

The power unit controls the motor. The safety limit switches disconnect the power unit.

Motor

The motor is controlled by the power unit and drives the axis.

Encoder

The encoder provides information on distance and direction.

Easy Motion Control

Easy Motion Control provides the following functions:

- Jogging
- Reference search
- Absolute/relative positioning
- Electronic gears
- Reference setting
- All common monitoring functions for positioning
- Velocity override
- Position control
- Simulation
- Preassembled input and output drivers for encoder or analog output groups

Modules for distance measurement and setpoint output

The modules form the interface between the CPU and the machine. They read axis positions and output setpoint values to the power unit. In the configuration model, this is implemented by IM 178-4.

CPU

The CPU executes the user program in which the Easy Motion Control blocks are embedded.

PC/PG

A PC/PG is used for the following functions:

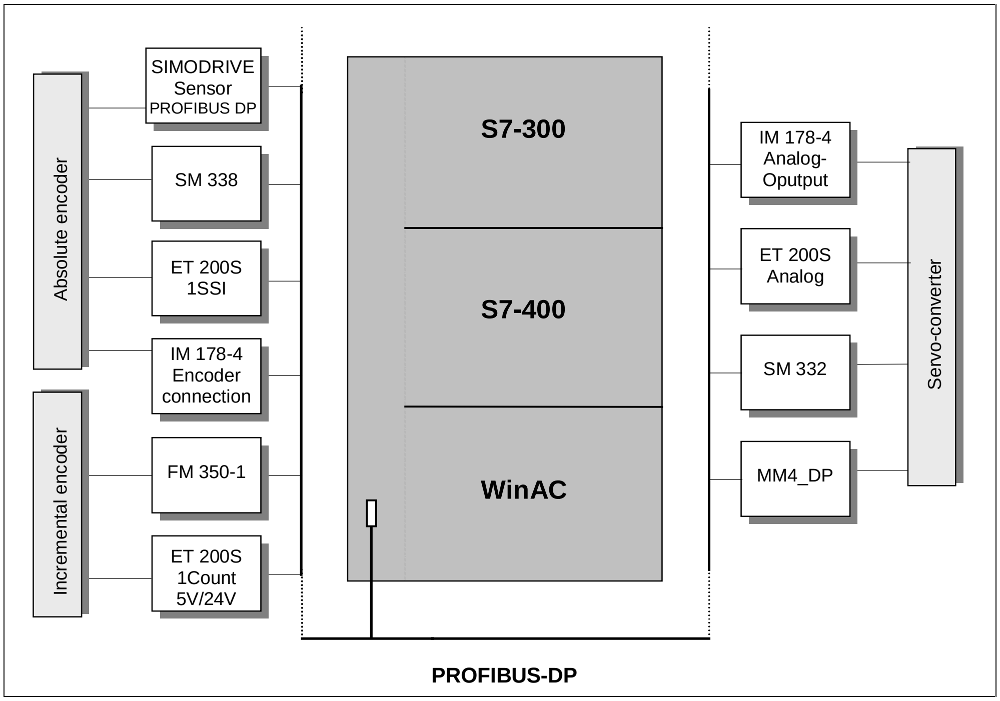
- Parameter assignment: you assign parameters to Easy Motion Control either with configuration software or directly in the axis data block (see Axis Data Block Setup).
- Programming: you link the Easy Motion Control blocks directly into your program.

1.3 Supported Hardware Configurations

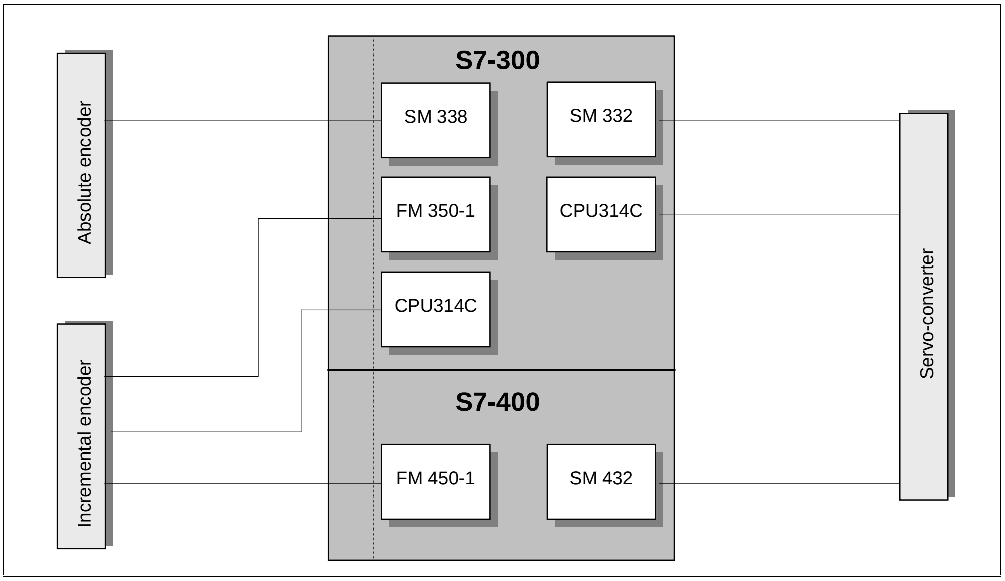
The following hardware configurations are comfortably supported by driver FBs. These are supplied with Easy Motion Control.

It is possible to adapt to configurations that are not directly supported using universal driver FBs that are also supplied.

Distributed configurations for Easy Motion Control.



Central Configurations for Easy Motion Control.



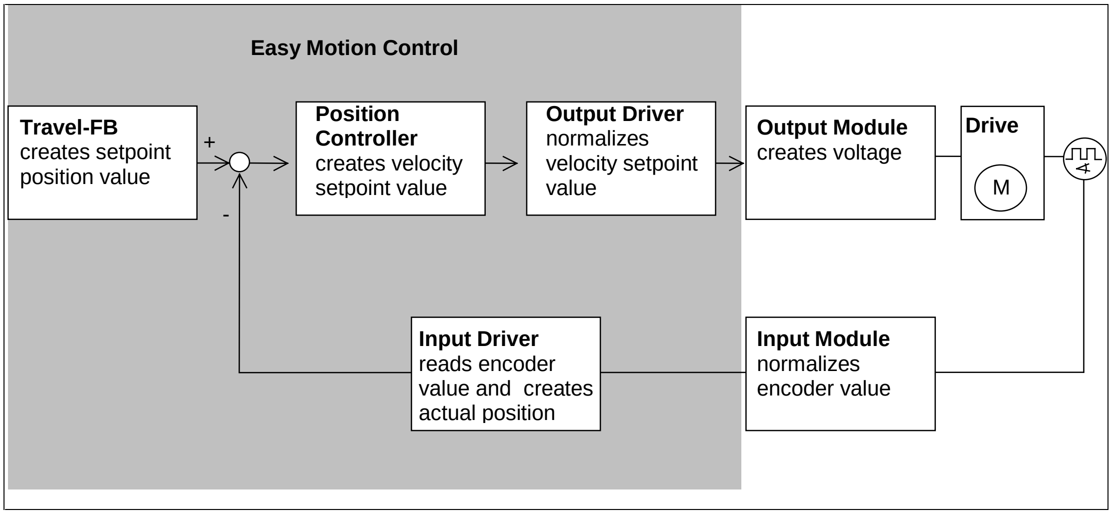
2 Principles

2.1 Principles of Positioning with Easy Motion Control

Setting up position control

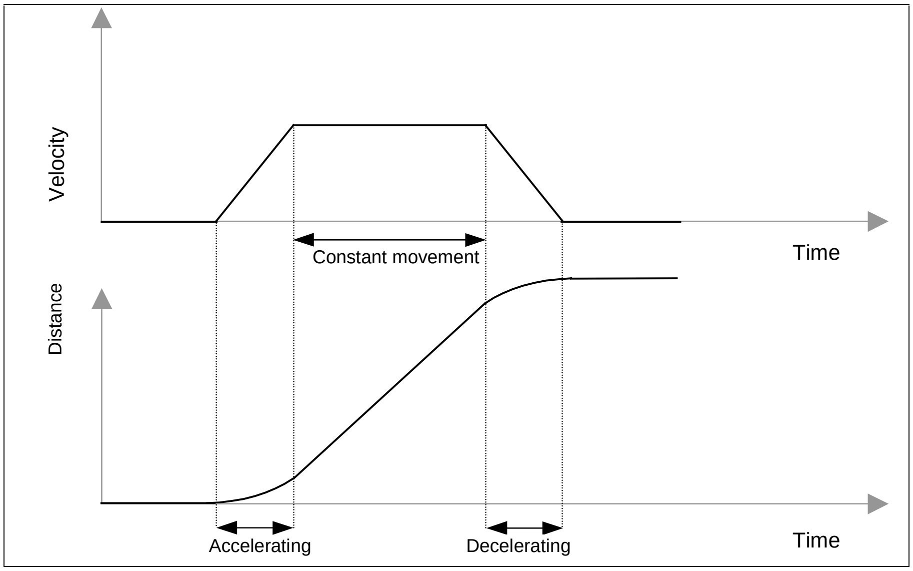
With controlled positioning, the current position of the axis is compared with the setpoint position. Any deviations are compensated using the position controller both during a travel movement as well as at a standstill.

The following diagram shows the general arrangement for controlled positioning with Easy Motion Control. A travel block specifies setpoint position values, the input driver records the actual position of the axis, the position controller compares the actual position with the setpoint position and from the difference generates a velocity setpoint value and the output driver outputs the standardized velocity setpoint values.



Easy Motion Control
Travel-FB creates setpoint position value
+
-
Position Controller creates velocity setpoint value
Output Driver normalizes velocity setpoint value
Output Module creates voltage
Drive
M
Input Driver reads encoder value and creates actual position
Input Module normalizes encoder value

The following diagram shows the connection between axis velocity and distance covered during a travel movement.



2.2 Concept Definitions

Target

The target is the absolute or relative position of the axis which is approached during a positioning.

Travel movement

Travel movements are triggered by launching function blocks which are referred to below as travel FBs

Residual distance

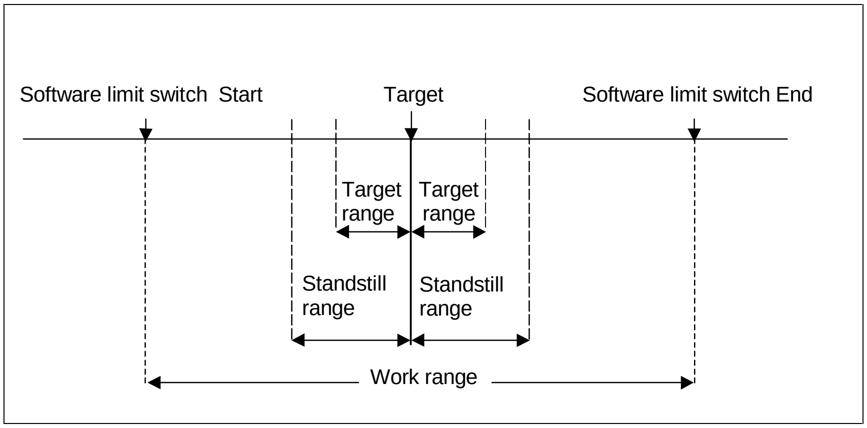
Residual distance is the difference between target position and current position setpoint.

Following distance/following error

During a travel movement, a difference builds up between the current position setpoint and the actual position value due to axial inertia. This is called following distance. If following distance exceeds a parametrizable value, there is a following error.

Monitoring ranges

- Work range:
  Defines your axis permitted travel range which you determine for your task using the software limit switch or the end of the rotary axis.
- Standstill range:
  Defines a parametrizable range which the axis should not leave when at a standstill. The standstill range lies to the left and right of the target.
- Target range:
  If the setpoint value has reached target at the end of a motion, the approach of the axis into the target range is monitored. Target range is used to monitor positioning precision of your application and lies to the left and right of the target.
  Immediately after the target range is reached, standstill range monitoring commences, which is why the standstill range should be greater than the target range.



2.3 Functions

Easy Motion Control is used to position an axis on a target with selectable acceleration, velocity and deceleration or move it at constant velocity. The necessary travel movements are triggered by launching travel FBs.

When a travel order starts, the target or distance of the travel movement and possibly the required acceleration, velocity or deceleration is configured on the travel FB. The measurement system (unit of length) of these values can be selected freely (e.g. mm).

The possible functions and travel movements are presented briefly below. For detailed description of the travel FBs, see FB MC_MoveJog, FB MC_MoveAbsolute, FB MC_MoveRelative, FB MC_Home and FB MC_GearIn.

2.3.1 Jogging

The jogging function allows you to move the drive in the positive or negative direction. No target is specified.

The jogging function is implemented using FB MC_MoveJog.

2.3.2 Synchronization

In order to assign a reproducible encoder value to the real position, a reference (synchronization) must be established between axis position and encoder value.

The synchronization function is implemented using FB MC_Home.

Synchronizing with an incremental encoder

An incremental encoder does not give any information about the immediate position of the axis when the system is switched on. Instead, it starts counting at the current position of the axis.

Resynchronization is always necessary when the system is switched on again after occurrence of errors or after replacing the encoder.

An incremental encoder may be synchronized

- by a reference search
- by reference setting.

Synchronizing with an absolute encoder

When commissioning absolute encoders, the value indicated by the encoder must be synchronized with the actual axis position.

Resynchronization is not necessary when the system is switched back on or after errors have occurred. After the encoder has been exchanged, however, resynchronization is necessary.

An absolute encoder can only be synchronized by reference setting (also called absolute encoder adjustment).

Reference search

The axis is moved in the direction of the reference point. After overriding the reference point, the axis is synchronized, i.e. a position value is assigned to the numerical value of the incremental encoder. A new coordinate value can be assigned to the reference point with every reference search.

A reference search is only possible with incremental encoders.

Reference setting

With reference setting, a new coordinate value is assigned to the current position.

Reference setting is possible with incremental and absolute encoders.

2.3.3 Absolute or Relative Positioning

Easy Motion Control can move the axis

- onto **absolute** targets
- in a specific direction in relation to a distance.

Absolute and relative positioning functions are implemented using FB MC_MoveAbsolute and FB MC_MoveRelative.

2.3.4 Gears

Easy Motion Control creates an electronic gear linking two axes in FB MC_GearIn.

2.3.5 Stopping Travel

You can stop current travel at any time.

The stop function is implemented using FB MC_StopMotion.

2.3.6 Overriding Travel

A travel FB that is currently active can at any time

- be overridden by starting another travel FB or
- can be interrupted by the FB MC_StopMotion (see Sequence of Travel Movements).

2.3.7 Tracking

With tracking, the position setpoint is continuously adjusted (tracked) in the position controller to the actual position value. During tracking, the axis is not monitored for leaving the standstill range. The function is activated by removing the "EnableDrive" signal on FB MC_Control or after an error occurs.

2.3.8 Velocity Override

Velocity override allows the configured velocity of a travel FB to be varied dynamically. Acceleration and deceleration values are not affected.

For further information on velocity override, see Velocity Override Control Value.

2.3.9 Error Responses and Diagnosis

All errors detected by the blocks are displayed in the axis DB by

- setting a group error and
- an error-specific display.

With errors requiring acknowledgement, an error acknowledgement must be set after rectifying the error.

If the error has been caused or detected by a travel FB (e.g. FB MC_MoveJog or FB MC_MoveAbsolute), the "error" output is set at this FB ("CommandAborted").

If the error was detected by a driver block, however, the exact cause of the error is indicated in the static data of FB (see Input Driver).

For detailed information on error response, error display and error acknowledgement, see Error Displays and Error Acknowledgement in Axis DB.

2.3.10 Simulation

In simulation mode, you can test your travel program without moving your axis.

Simulation is implemented using FB MC_Simulation.

2.4 Linear Axis

The **work range** extends from the software limit switch start to the software limit switch end and is limited to 1 x 2^{24} length units (in the coordinate range between -2^{24} and +2^{24}).

The **physical axis length** is limited by the mechanical stops of your axis.

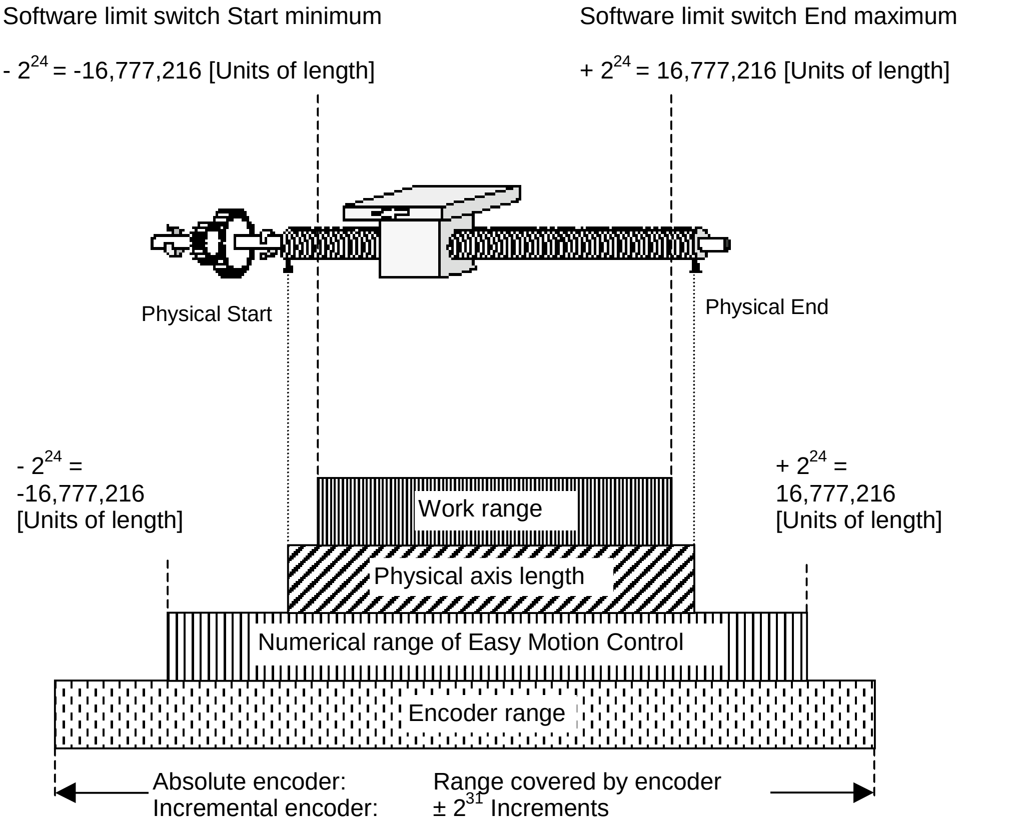
The **numerical range** of Easy Motion Control is only relevant if work range monitoring is switched off. It is limited to 2 x 2^{24} length units (from -2^{24} to +2^{24}).

The **encoder range** is

- the range covered by the encoder in the case of an absolute encoder and
- the distance from - 2^{31} to + 2^{31} steps in the case of an incremental encoder.

With linear axes, the **work range** (= travel range) is limited as follows:

Work range < physical axis length < numerical range of Easy Motion Control < encoder range.



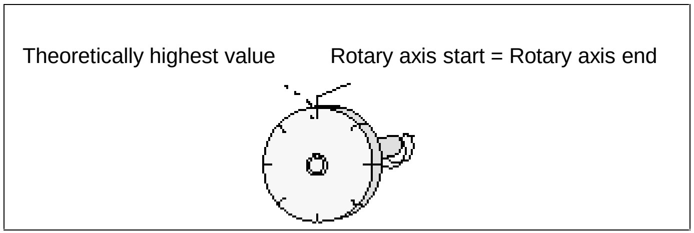
2.5 Rotary Axis

The **work range** extends from the start of the rotary axis to the end of the rotary axis and is limited to 1 x 2^{24} length units (in the coordinate range between -2^{24} and $+2^{24}$).

Encoder range

- With an incremental encoder, the encoder range is arbitrary
- With an absolute encoder, the rule is as follows:
  encoder range = work range x n
  with
  n = 1 with single turn encoders
  n = integer with multi-turn encoders

With a rotary axis, the actual value is reset to the start of the rotary axis again after every revolution. Rotary axes can thus be moved continuously beyond the end or start of the rotary axis. The axis position is always displayed as a value between the start and end of the rotary axis.


Theoretically highest value
Rotary axis start = Rotary axis end

3 Installation

3.1 Installation of Easy Motion Control

Easy Motion Control contains a setup program which carries out installation automatically. Input prompts on the screen guide you step by step through the entire installation procedure. The program is selected with the standard software installation procedure under Windows.

The important phase of the installation is copying the data onto your PC/PG.

Installed Components

After installation, you will find the following components on your PC:

- an S7 library of loadable blocks
  (library name: EMC2 Easy Motion Control),
- a configuration software (including online help) that is executable with STEP7
  (select: **Start > Simatic > STEP 7 > Easy Motion Control V2**),
- an S7 sample project
  (project name: ZEn20_02_EMC2),
- an electronic manual
  (select: **Start > Simatic > Documentation > English > Easy Motion Control V2**),
- a readme file
  (select: **Start > Simatic > Product Information > English > Easy Motion Control V2 - Readme**) and
- Getting Started
  (select: **Start > Simatic > Documentation > English > Easy Motion Control V2 - Getting Started**).

Installation Requirements

- PC or programming device with specific requirements (see README.wri file)
- Microsoft Windows operating system as for STEP 7
- SIMATIC STEP 7 basic package from V5.0 + Service Pack 3
- Available memory:
  For required hard disk space, see README.wri.

Installation Preparations

Before you can begin installation, Windows must be started.

Starting the Installation Program

For installation, proceed as follows:

1. Under Windows, start the software installation dialog by double-clicking the "software" symbol in the "control panel".
2. Click on "Install".
3. Insert the CD-ROM and click on "Continue". Windows now automatically searches for the installation program setup.exe.
4. Follow the installation program instructions step-by-step.

The program will guide you through the installation process step-by-step. In each case you can advance to the next step or go back to the previous step.

- During the installation process, questions are displayed in dialog boxes or options are offered for you to select. Please read the following information in order to be able to answer the dialogs faster and more easily.

Errors during Installation

The following errors will cause installation to abort:

- If an initialization error occurs immediately after Setup has started, Setup was very probably not started under Windows.
- Out of memory: depending on the extent of installation, you need enough free memory on your hard disk (see README.wri).
- Defective diskette/CD: if you notice that a diskette/CD is defective, please contact your SIEMENS representative.
- Operating error: start installation again and follow the instructions carefully.

3.2 Uninstalling Easy Motion Control

Use the standard Windows uninstaller:

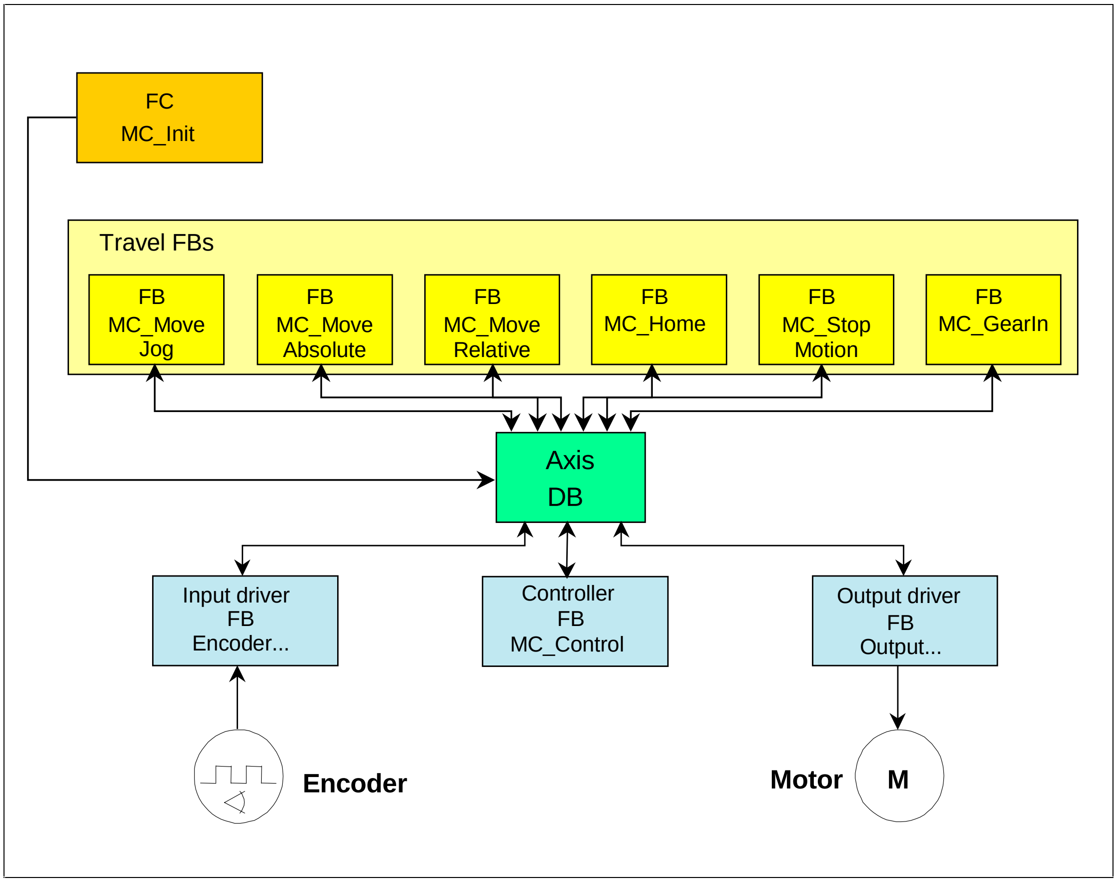
1. Start the software installation dialog by double-clicking the "software" symbol in the Windows control panel.
2. Select the Easy Motion Control entry from the list of installed software that is displayed. Then press the "Remove" button to uninstall the software.

If the dialog boxes "Remove Enabled File" appear, click on the "No" button if in doubt.

4 Blocks for Easy Motion Control

4.1 User Program Setup

Cooperation of Modules



FC MC_Init is used to **initialize** the blocks.

The input driver (FB Encoder…) reads in the actual position of an axis from the I/O module and standardizes this into the unit of length that you have selected.

One of the **travel FBs** generates the position setpoint value.

The position controller (FB MC_Control) compares the actual position value with the position setpoint and from the difference generates a velocity setpoint value.

The **output driver** (FB Output…) standardizes the velocity setpoint value to the correct format and writes it to the connected I/O module.

The program for a **closed-loop control circuit** consists of an input driver (FB encoder…), at least one motion controller, the controller (FB MC_Control) and one output driver (FB Output…).

All blocks of an axis work together with one **axis DB**. This contains all the axis data necessary for using the axis and is available for each axis. See information on loading the axis DBs in Axis Data Block Setup.

Call environment

To ensure correct functioning of position control, it is important that all functions of one axis are processed in a fixed **time frame**. This time frame is determined by the call of the Easy Motion Control blocks in **one** run level. The following two methods can be applied, depending on the CPU and type of I/O interconnection used:

Cyclic Interrupt OB (OB30-OB38)
Used for central or distributed interconnection of the I/O at a DP bus WITHOUT constant bus cycle time. The watchdog cycle corresponds with the positioning control cycle.

Synchronous Cycle Interrupt OB (OB61-OB64)
The positioning control cycle on a continuously cycled DP bus for I/O interconnections corresponds with the value you have set in the bus configuration for "Constant DP cycle time".

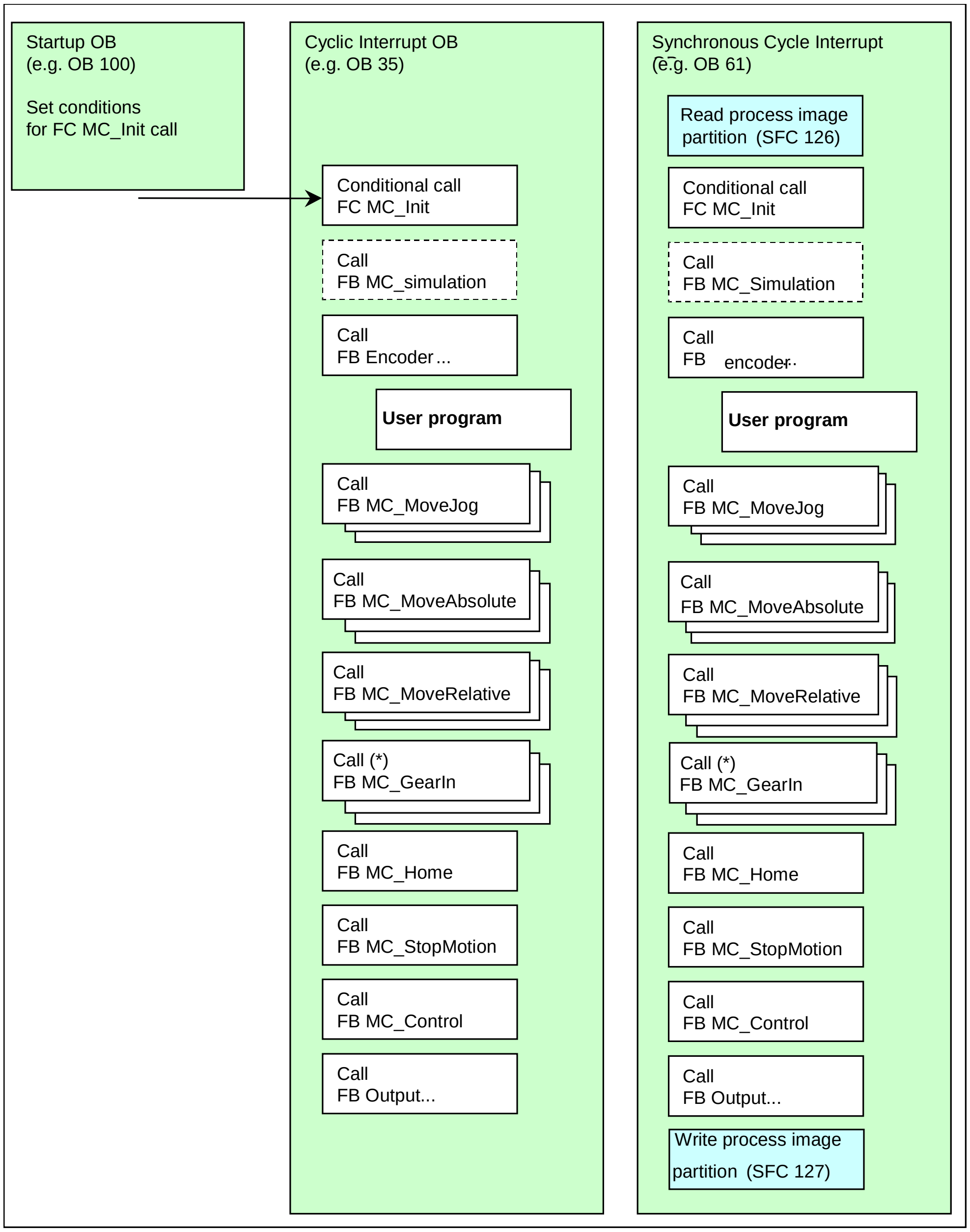
This value must be entered at the axis DB (Scan Cycle Time) in both cases.

General information on clocking is found in the STEP 7 Online-Help.

Please note the following special features of Easy Motion Control:

- In your DP slave and DP bus (and in their modules, if necessary) configuration, you must set and configure the "Constant bus cycle time".
- The I/O addresses used for the acquisition of the position or for the output of the setpoint must be placed in a process image partition (PIP). Both gear axes must be placed into the same PIP.
- In the CPU configuration, this PIP is assigned to the clocked interrupt used.
- You need to update this PIP in the interrupt OB before and after each call of the EMC blocks by means of the system function blocks SFC 126 "SYNC_PI"/SFC 127 "SYNC_PO".
- Evaluate these SFC feedbacks in your program and react accordingly (in the event of an error, the PIP may not have been updated and the process value is thus faulty).

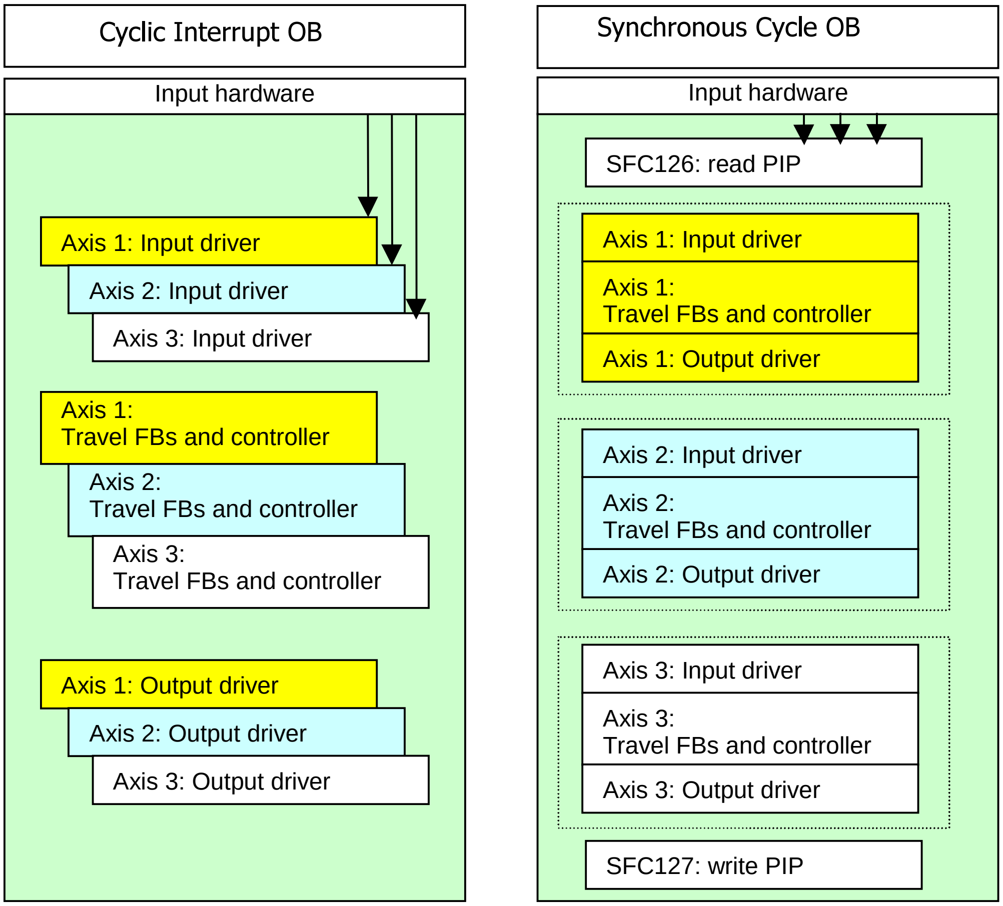
Recommended call sequence



(*) Calling the gear block MC_GearIn is appropriate only for the slave axes!

Multiple axes

We recommend the following user program structure for controlling multiple axes in one CPU:



Recommended for the **cyclic interrupt OB**.

Advantage: The call of the input drivers at the start of the produces a minimum time offset and jitter when reading the I/O.

Disadvantage: It is not possible to group *all* blocks of an axis in a *single* multiple instance FB.

Recommended for the **synchronous cycle OB**.

Advantage: It is possible to group *all* blocks of an axis in a *single* multiple instance FB.

Due to the PIP, which is required for the clocking OB, the read I/O data are consistent irrespective of the call of the driver consistent.

All interacting axes must be called in the same OB.

All axes of a isochrone application must be placed into the same process image partition.

Handling Instance Data

You can enter **instance data** for each FB in its own DB or you can enter instance data for all FBs in one multi-instance DB.

After an FB's instance data is loaded into the module, initialization of the FB is necessary. To do this, the input "Init" of the FB must be set. See also Initialization and Parameter Changes and Saving and Loading the Axis Data Block.

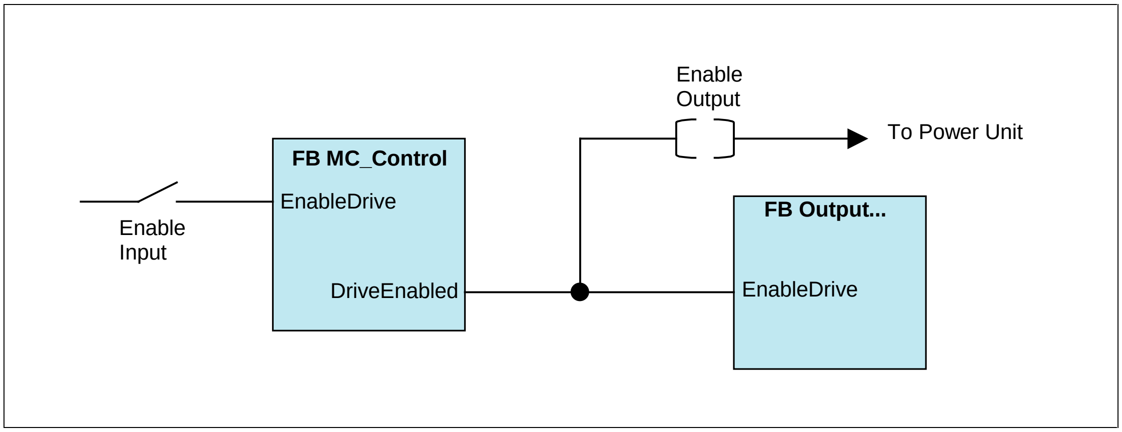
Interconnection of Blocks

The connections between FBs and axis DB shown in the diagram under Combination of Blocks are made via the parameters (see Importance of shared parameters):

- Axis and
- Init

The following connections are also necessary:

- The "EnableDrive" input of FB MC_Control must be interconnected with an enable switch. Only if "EnableDrive" = TRUE will FB MC_Control output a speed setpoint value.
- The "EnableDrive" input of the FB Output ... must be interconnected with the "DriveEnabled" output of FB MC_Control. Only if "EnableDrive" = TRUE will the FB Output ... produce a value that is not zero.
- You should interconnect the "DriveEnabled" output of FB MC_Control with an enable input of your drive.



4.2 Influences on Position Control in the CPU

Recommended for optimal position control:

- Scan time must be adhered to as closely as possible.
  In HW Config of the CPU, parameterize as high a priority as possible for the cyclic interrupt OB by selecting the blocks.
- In HW Config of the CPU, parameterize as low a cycle load through communication as possible.
- Use a isochrone environment to avoid having to take additional measures.
- With encoder recording linked via distributed I/Os (PROFIBUS DP), set as high a transfer speed of the PROFIBUS as possible (e.g. 12 Mbit/s).

4.3 Importance of Shared Parameters

The parameters listed in the following table are used in all blocks.

P type	Parameter	Data type	Meaning
IN	EN	BOOL	Enable input of LAD/FBD-Box
OUT	ENO	BOOL	Enable output of LAD/FBD-Box
INOUT	Axis	STRUCT	Interconnect this parameter with the axis DB With a DB of type AXIS_REF, the interconnection is • Axis := DBname.Ax With a DB containing a variable of type AXIS_REF, the interconnection is • Axis := DBname.Variable name.Ax
	Init	BOOL	Interconnect this parameter with an initialization bit in the axis DB which is not used by any other FB ("Init.I0" .. "Init.I31", see Bit Field for the Initialization Function). The block carries out its initialization at Init= TRUE and then resets the bit. With a DB of type AXIS_REF, the interconnection is • Init := DBname.Ax.Init.Ix With a DB containing a variable of type AXIS_REF, the interconnection is • Init := DBname.Variablenname.Ax.Init.Ix with Ix = I0 .. I31

4.4 Initialization and Parameter Changes

Easy Motion Control block functioning is based on the data entered in the axis DB. This is where Parameter can also be found which describe the basic axis itself. If these parameters are changed (they are identified in Axis Data Block Setup by I), it is necessary for **each** Easy Motion Control block selected in your user program to perform its initialization **once** after the change.

For this purpose, the bit field "Init.Ix" is defined in the axis DB and each block has the parameter "Init". The blocks run through their initialization routine if their input "Init" is set and then reset the interconnected initialization bit.

The bits in the bit field "Init.Ix" are set in the axis DB:

- every time FC MC_Init is selected as well as
- by configuration software in the event of any change in corresponding parameters of the axis DB.

With every Easy Motion Control block selected in your user program, interconnect the "Init" parameter with an initialization bit in the DB axis that is not used by any other function block. This ensures that the blocks initialize themselves if this is necessary as the result of a parameter change or CPU startup ("Init I0" .. "InitI31", see Bit Field for Initialization Function).

You call the FC MC_Init once

- after every startup of the CPU or
- if you have changed one of the parameters identified by I in the Axis Data Block Setup without having changed the configuration software.

For additional information on the initialization block, see FC MC_Init

Warning

Changing the parameters identified by I in Axis Data Block Setup is only permitted when the axis is at a standstill.

Auxiliary values are calculated during initialization and these are entered in the static parameters of the function blocks and in the axis DB. For this reason, you should only overwrite instance data blocks and the axis DB when the axis is at a standstill.

If you do not observe these requirements, controlled axis movement is not guaranteed.

4.5 Sequence of Travel Movements

Start of a Travel Order

The input parameters of the travel blocks are interpreted once at the start of a travel order. Changing these values while an order is running does not affect current travel.

Maximum Travel Distance

Travel distance per motion is limited to the distance corresponding to 2^{24} steps of the encoder. It is calculated as follows:

$$\text{Maximum Travel Distance} = 2^{24} \times \frac{\text{"Axis distance per encoder revolution"}}{\text{"Steps per encoder revolution"}}$$

Example:

Axis distance per encoder revolution	= 2.5 mm
steps per encoder revolution	= 4096

maximum distance = 2^{24} x 2.5 mm / 4096 = 10240 mm

Exception:
With a rotary axis, FB MC_MoveJog and FB MC_GearIn have no restriction of travel distance.

Overriding Travel

A travel FB that is currently active can at any time

- be overridden by starting another travel FB or
- be interrupted by starting FB MC_StopMotion
- be aborted by setting manual mode.

When overridden by another travel FB, the axis is slowed down or accelerated to the speed of the overriding block and then moves to the new target without any interim stop. The overridden travel FB sets the "CommandAborted" output.

If the new motion requires a change in direction of the axis, it is first stopped and then moved in the opposite direction (reversal).

For reasons of safety, the deceleration of the two overriding travel FBs must coincide. This ensures that, on reversing direction, the **original** target of the movement is never exceeded.

Multiple Use of a Travel FB

If required in your program, you can call a travel FB several times for one axis.

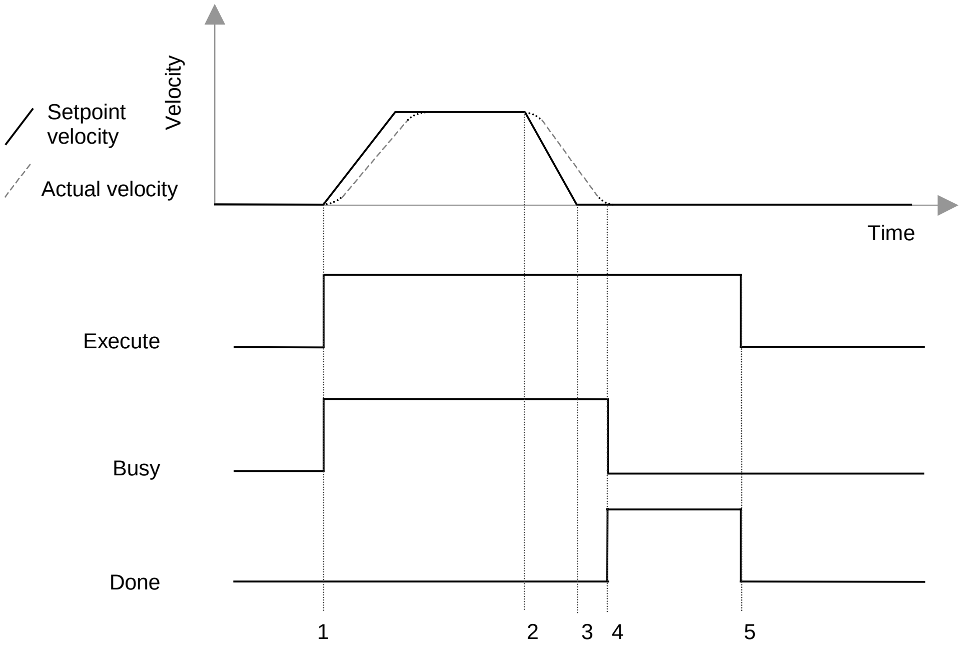
You will find examples of multiple use of a travel FB under Example 5 and Example 6.

Error Response

All errors which occur are indicated in the axis DB, where they must be acknowledged by the user (see Error Displays and Error Acknowledgement in the DB Axis). Each error will abort any travel order which is currently active. A new travel order can only start if there is no longer any error.

Signal flow diagrams

Travel without error:



1 Start of the Travel FB

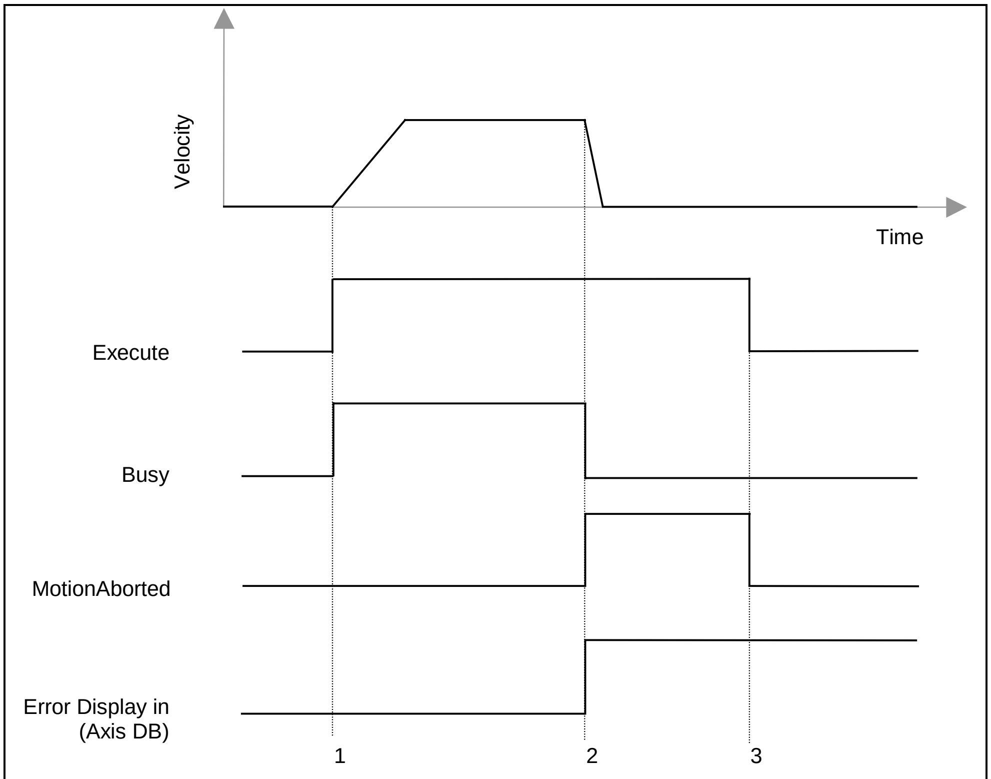
2 Brake location

3 Setpoint value has reached the target position.

4 Axis has reached the target range.

5 It will be reset to "Done" by resetting "Execute". The procedure is completed.
If "Execute" is reset before the target position has been reached, "Done" will be set for a cycle when the target position has been reached and then be reset.

Travel with error:



1 Start of the Travel FB

2 Error occurs

3 By resetting "Execute", the feedback "CommandAborted" will be reset
If "Execute" is reset before an error occurs, "MotionAborted" will be set for a cycle when an error occurs and then be reset

4.6 FC MC_Init (FC 0)

Function

FC MC_Init prepares the initialization of all blocks of an axis by setting all bits in the bit field "Init.Ix" of the axis DB (see Initialization and Parameter Changes).

FC MC_Init **prevents any inadvertent approach of the axis**, by setting the error displays "Error" = TRUE and "Err.StoppedMotion" = TRUE in the Axis DB. If you want to move the axis after selecting FC MC_Init, you first have to remove the error displays by acknowledging them (see Error Displays and Error Acknowledgement in the axis DB).

FC MC_Init checks the parameters in the axis DB. If a configuration error is found, the bits "Error" and "Err.ConfigErr" are set in the axis DB for (collective) error display, as well as one of the bits in the "ConfigErr" structure for more accurate error display.

Configuration errors **cannot** be acknowledged. You have to recall FC MC_Init after correcting the error.

For more information on error strategy, see Error Displays and Error Acknowledgement in the Axis DB and Configuration Errors.

Note

It is only permitted to call FC MoveInit if the axis is at a standstill.

Call

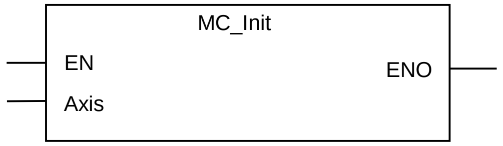
You call FC MC_Init conditionally in the same program block in which you also call the other FBs of this axis. The conditions of the call are as follows:

- after every startup of the CPU (OB100 and OB101) or
- after changing one of the parameters identified in the Axis Data Block Setup by I without the configuration software.

Note

Reset the condition used for launching FC MC_Init if the call has been made once. Otherwise the Easy Motion Control blocks continuously repeat their initialization because the initialization bits in axis DB are set again.

Ladder diagram display



4.7 Input Drivers (FB 21 to FB 29)

Function

The input driver is used to record the actual positions of an axis. It reads the non-standardized axis position and converts this into the selected unit of length.
For a number of distance measurement modules, suitable input drivers are available as described individually in the following sections. Adaptation to configurations that are not directly supported is possible by way of FB EncoderUniversal.

Initialization

- With an **incremental encoder,** synchronization is reset with the axis (parameter "Sync" = FALSE in the axis data). A new reference search or reference setting is then necessary.
- With an **absolute encoder**, synchronization is maintained with the axis.

Call

The input driver of an axis must be launched unconditionally.

Isochrone mode

The output drivers support isochrone mode at all suitable DP modules. The driver automatically recognizes isochrone mode if it is called in a synchronous cycle OB.

Error Handling

There are the following mechanisms for detecting errors:

- An error detected by the user program (e.g. module removed or station failure) may be notified to the input driver at the input "EncErr". Provided input "EncErr" = TRUE, the input driver does not access the distance measurement module.
- Some distance measurement modules provide error information which is evaluated automatically by the input driver.
- Other distance measurement modules only provide their error information via diagnostic interrupt and data record 1 (DS1). You must therefore
  - In HW Config, enable the diagnostic interrupt for the module and
  - In OB82, compare the address provided in the local date "OB82_MDL_ADDR" with your module address. If these coincide, set the parameter "ReadDiagErr" of the input driver at TRUE (both when the interrupt comes and goes).
    Using "ReadDiagErr" = TRUE, the input driver is prompted to read data record 1 of the module via SFC RD_REC and file it in the static parameter "DiagStatus".
    It then sets "ReadDiagErr" = FALSE. The return value of SFC RD_REC is filed in the static parameter "JobErr".

Error response:

- In the event of an error, the displays "Error" and "Err.EncoderErr" are set in the axis DB.
- With an incremental encoder, the status bit "Sync" in the axis DB is always reset.
- With an absolute encoder, the status bit "Sync" in the axis DB is never reset.

Please also note specific error handling in the following sections on individual drivers.

For detailed information on error response, error display and error acknowledgement, see Error Displays and Error Acknowledgement in axis DB.

Simulation Mode

Provided the bit "Sim" is set in the axis DB, the input driver does not access the distance measurement module. The encoder values are simulated via the simulation block FB MC_Simulation.

4.7.1 FB EncoderIM178 (FB 21)

Function

FB EncoderIM178 allows the use of IM 178-4 as a distance measurement module for incremental and absolute encoders.

An identical channel number and axis must be set at the IM178-4 input AND output drivers, since certain addresses refer to both drivers. A combination with other drivers is not possible.

Configuration of IM 178-4 in HW Config

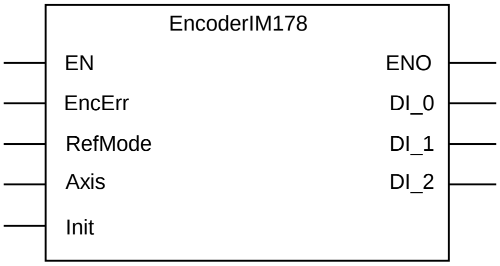
Select the IM 178-4 from the module catalog in HW Config under **PROFIBUS-DP > Function Modules > IM 178-4**. Define slot 4 of the IM 178-4 as "4Word AO/12Word AI/Cons. 1Word".

Parameter	Setting
Synchronization	Software
Edge selection	Rising, falling or both edges.

Parameterize the other settings according to the connected encoder.

Call

Ladder diagram display



Parameter Description

P type	Parameter	Data type	Meaning
IN	EncErr	BOOL	1 = Encoder error: Here you inform the input driver of an error detected by the user program (e.g. module removed or station failure).
	RefMode	BOOL	Referencing type: 0 = with digital input I0 1 = with digital input I1 and zero mark
OUT	DI_0	BOOL	Digital input 0 of corresponding channel of IM 178-4
	DI_1	BOOL	Digital input 1 of corresponding channel of IM 178-4
	DI_2	BOOL	Digital input 2 of corresponding channel of IM 178-4
The following static parameters of the block are only used for diagnostic purposes in the event of an error.			
STAT	Status.PARA	BOOL	PARA = TRUE: IM 178-4 was correctly configured via HW Config. PARA = FALSE: see IM 178-4 manual
	Status.EXTF0	BOOL	1 = Encoder signal line fault
	Status.EXTF1	BOOL	1 = Encoder fault
	Status.EXTF2	BOOL	1 = Zero mark fault

Synchronization

Synchronization may either be performed

- by reference search (with incremental encoder only) or
- by reference setting (with incremental and absolute encoders).

Depending on the setting, connect the reference point switch to digital input I0 or I1.

With a reference search, IM 178-4 can detect the reference point optionally in the following cases:

- with falling, rising or both edges of digital input I0 (RefMode = 0) or
- with the rising edge of the first zero mark after detecting the falling edge of digital input I1 (RefMode = 1).

If the axis has already reached the reference point switch at the time the reference point approach is started, its position is referenced to the current reference position.

Digital Inputs

Digital inputs I0 to I2 of the selected channel of IM 178-4 are displayed in the output parameters "DI_0" to "DI_2" of the input driver.

Error Handling

As error response in the case of a detected error, the input driver sets the error codes "Error" and "Err.EncoderErr" in the axis DB. The input driver resets the status bit "Sync" in the axis DB if you are using an incremental encoder. For more details, see handbook *IM 178-4 Drive Interface.*

For detailed information on error response, error display and error acknowledgement, see Error Displays and Error Acknowledgement in axis DB.

Error Acknowledgement

If you selected error acknowledgement in the axis DB of the Easy Motion Control, errors in the module are also then acknowledged.

4.7.2 FB EncoderFM450 (FB 22)

Function

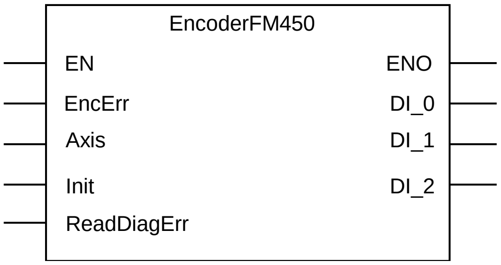
FB encoderFM450 allows the use of one channel of FM 450-1 as a distance measurement module for incremental encoders. The channel number is configured via the axis DB.

Configuration of the Module in HW Config

Setting mask	Setting
Diagnostic data	enable
Operating modes	• Counting range limits: -31 to +31 bit, • Operating mode: continuous counting, • Gate control: periodic counting.
Input signals	If you want to make a reference search, you must select the following setting: • Set counter (DI Set): Single. • The setting "Evaluate zero mark for setting" may be engaged or disabled as a reference point criterion.

Call

Ladder diagram display



Parameter Description

P type	Parameter	Data type	Meaning
IN	EncErr	BOOL	1 = Encoder error: Here you inform the input driver of an error detected by the user program (e.g. module removed or station failure).
INOUT	ReadDiagErr	BOOL	Set this bit in OB82 if the module has triggered a diagnostic interrupt (see below under Error Handling).
OUT	DI_0	BOOL	Digital input 0 of selected channel
	DI_1	BOOL	Digital input 1 of selected channel
	DI_2	BOOL	Digital input 2 of selected channel
The following static parameters of the block are only used for diagnostic purposes in the event of an error.			
STAT	JobErr	INT	Return value of SFC RD_REC if "ReadDiagErr" = TRUE was set.
	DiagStatus. EXT_VOLTAGE	STRUCT	External auxiliary supply defective
	DiagStatus. CONFIG_ERR	STRUCT	Configuration defective
	DiagStatus. WTCH_DOG_FLT	STRUCT	Time watchdog addressed
	DiagStatus. RAM_FLT	STRUCT	RAM defective
	DiagStatus. CH1_SIGA	STRUCT	Channel 1 signal A defective
	DiagStatus. CH1_SIGB	STRUCT	Channel 1 signal B defective
	DiagStatus. CH1_SIGZ	STRUCT	Channel 1 signal N defective
	DiagStatus. CH1_BETW	STRUCT	Channel 1 fault between channels
	DiagStatus. CH1_5V2	STRUCT	Channel 1 encoder supply 5.2V defective
	DiagStatus. CH2_SIGA	STRUCT	Channel 2 signal A defective
	DiagStatus. CH2_SIGB	STRUCT	Channel 2 signal B defective
	DiagStatus. CH2_SIGZ	STRUCT	Channel 2 signal N defective
	DiagStatus. CH2_BETW	STRUCT	Channel 2 fault between channels
	DiagStatus. CH2_5V2	STRUCT	Channel 2 encoder supply 5.2V defective

Synchronization

Synchronization may either occur

- by reference search or
- by reference setting.

Connect the reference point switch at digital input I2.

With a reference search, the module can detect the reference point optionally in the following cases:

- with a rising edge at digital input I2 or
- on the rising edge of the zero mark, if digital input I2 is set.

If the axis has already reached the reference point switch at the time the reference point approach is started, its position is referenced to the current reference position.

Digital Inputs

The digital inputs I0 to I2 of the selected channel are displayed in the output parameters "DI_0" to "DI_2" of the input driver.

Error Handling

The module only provides its error information via data record 1 (DS1).
In the following situations the input driver reads data record 1 from the module via SFC RD_REC and writes it to the parameter "DiagStatus":

- The input driver is being initialized.
- The module reports an error.
- The driver input parameter "ReadDiagErr" is TRUE.

Then it sets "ReadDiagErr" = FALSE. The return value of SFC RD_REC is written to the static Parameter "JobErr" (see JobErr Messages).

If an error display is registered in data record 1, the input driver sets the error codes "Error" and "Err.EncoderErr" at TRUE in the axis DB. The status bit "Sync" is set at FALSE.

For detailed information on error response, error display and error acknowledgement, see Error Displays and Error Acknowledgement in axis DB.

Note

If you are only using an FM450-1 in single-channel mode, you must parameterize the second channel as "encoder 24V incremental", as this will otherwise transmit diagnostic interrupts (e.g. due to missing wiring).

4.7.3 FB EncoderET200S1SSI (FB 23)

Function

The FB encoderET200S1SSI allows the use of the module ET 200S 1SSI Fast-Mode as a distance measuring module for absolute encoders.

The driver can also be used for the isochrone version of this module.

Configuration of module ET 200S 1SSI Fast-Mode in HW Config

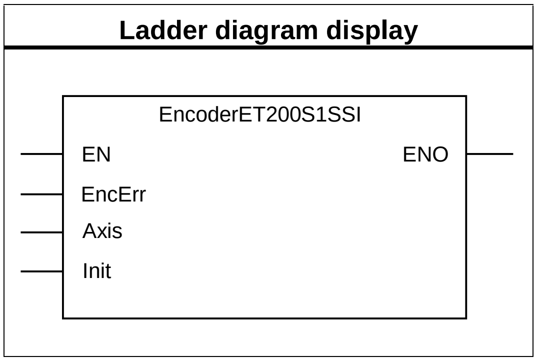
The input driver is only suitable for the module 1SSI **Fast-Mode** from the HW Config catalogue.

Select the submodule from the module catalog in HW Config under **PROFIBUS-DP > ET 200S > IM 151 > FM**.

Parameter	Setting
Group diagnosis	You can engage this when you want to make an evaluation in your program.
Detection	Freewheeling
Reversal of direction of rotation	Switched off Reversal of direction of rotation is possible with the Easy Motion Control blocks (parameter "Set encoder polarity", see Encoder Data).

Configure the other settings according to the connected encoder.

Call



Parameter Description

P type	Parameter	Data type	Meaning
IN	EncErr	BOOL	1 = Encoder error: Here you inform the input driver of an error detected by the user program (e.g. module removed or station failure).
The following static parameters of the block are only used for diagnostic purposes in the event of an error.			
STAT	Status. STS_UP	BOOL	Forward direction: encoder values increase
	Status. STS_DN	BOOL	Reverse direction: encoder values reduce
	Status.STS_DI	BOOL	Status of digital input
	Status.EXTF	BOOL	Group error or short-circuit in encoder supply
	Status. ERR_PARA	BOOL	Configuration error
	Status.RDY	BOOL	Module is ready for operation

Synchronization

With module ET 200S 1SSI Fast-Mode, synchronization is only possible via reference setting.

Error Handling

If the input driver detects an error display in the module, it sets the error codes "Error" and "Err.EncoderErr" in the axis DB.

For detailed information on error response, error display and error acknowledgement, see Error Displays and Error Acknowledgement in Axis DB.

4.7.4 FB EncoderAbsSensorDP (FB 24)

Function

FB encoderAbsSensorDP allows the use of the absolute encoder SIMODRIVE sensor with Profibus DP for recording position.

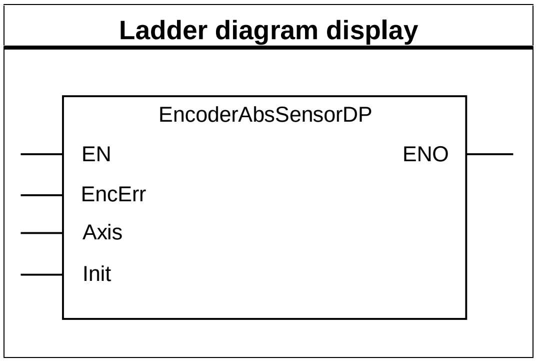
Configuration of Absolute Encoder Simodrive Sensor in HW Config

In HW Config, select encoder "PROFIBUS-DP" > "ENCODER" > "SIMODRIVE sensor" > "Version 2.2 Singleturn" or "Version 2.2 Multiturn".

Parameter	**Setting**	
Code sequence	Increasing clockwise Reversal of direction of rotation is possible with the Easy Motion Control blocks (parameter "Set encoder polarity", see Encoder Data).	
	Multiturn	**Single-turn**
Scaling function	Enable	Enable
Measuring units (high)	0	-
Measuring steps (low)	4096	4096
Physical impulses (high)	0	-
Physical impulses (low)	4096	4096
Desired measuring units per	revolution	revolution
Total measuring range (high)	256	-
Total measuring range (low)	0	4096
Commissioning mode	Disable	Disable
Shorter diagnostics	No	No
Lower limit switch	Disable	Disable
Lower limit switch (high)	0	-
Lower limit switch (low)	0	0
Upper limit switch	Disable	Disable
Upper limit switch (high)	0	-
Upper limit switch (low)	32767	4096
Velocity output unit	Steps/1000 ms	Steps/1000 ms

You make the adaptation to your selected unit of length with the Easy Motion Control blocks. Fixed parameters can therefore be assigned to the encoder - as indicated above. No "teach-in" is necessary for the encoder (see encoder manual). At the parameter "steps per encoder revolution" of Easy Motion Control, enter 4096 ("StepsPerRev").

Call


Ladder diagram display

Parameter Description

P type	Parameter	Data type	Meaning
IN	EncErr	BOOL	1 = Encoder error: Here you inform the input driver of an error detected by the user program (e.g. module removed or station failure).
The following static parameters of the block are only used for diagnostic purposes in the event of an error.			
STAT	Status.Ready	BOOL	1 = encoder is ready for operation
	Status.Mode	BOOL	Operating mode: 0 = startup mode, 1 = normal mode
	Status.SWLimit Exceeded	BOOL	1 = Software limit switch MIN or MAX has triggered
	Status.Dir	BOOL	Direction of rotation: 0 = clockwise, 1 = counter-clockwise

Synchronization

With the Simodrive Sensor absolute encoder, synchronization is only possible by reference setting.

Error Handling

If the input driver detects that there is an error message from the encoder, it sets the error codes "Error" and "Err.EncoderErr" in the axis DB. For detailed information on error response, error display and error acknowledgement, see Error Displays and Error Acknowledgement in axis DB.

4.7.5 FB EncoderSM338 (FB 25)

Function

FB EncoderSM338 allows the use of SM 338 POS_INPUT as a distance measurement module for absolute encoders.

The driver can also be used for the synchronous cycle interrupt version of this module.

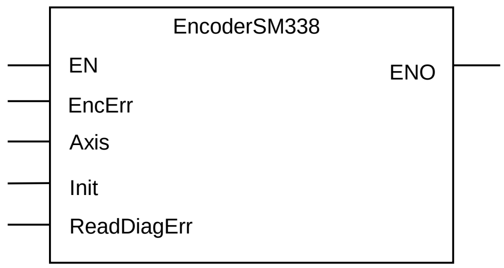
Configuration of SM 338 in HW Config

The freeze function must be disabled.
If the input driver detects freeze status of the encoder input, it sets the error codes "Error" and "Err.EncoderErr" in the axis DB.

Parameterize the other settings according to the connected encoder.

Call

Ladder diagram display



Parameter Description

P type	Parameter	Data type	Meaning
IN	EncErr	BOOL	1 = Encoder error: Here you inform the input driver of an error detected by the user program (e.g. module removed or station failure).
INOUT	ReadDiagErr	BOOL	Set this bit in OB82 if the module has triggered a diagnostic interrupt (see under Error Handling).
The following static parameters of the block are only used for diagnostic purposes in the event of an error.			
STAT	Freeze	BOOL	Freeze status of encoder input
	JobErr	INT	Return value of SFC RD_REC if "ReadDiagErr" = TRUE was set.
	DiagStatus. EXT_VOLTAGE	STRUCT	External auxiliary supply defective
	DiagStatus. CONFIG_ERR	STRUCT	Configuration defective
	DiagStatus. WTCH_DOG_FLT	STRUCT	Time watchdog addressed
	DiagStatus. RAM_FLT	STRUCT	RAM defective
	DiagStatus. CH0_INTF	STRUCT	Configuration or parameterization error (internal channel error)
	DiagStatus. CH0_EXTF	STRUCT	Encoder error (external channel error)

Synchronization

With the SM 338, synchronization is only possible by reference setting.

Error Handling

The module only provides its error information via diagnostic interrupt and data record 1 (DS1). You must therefore

- in HW Config, enable the diagnostic interrupt for the module and
- in OB82, compare the address provided in the local date "OB82_MDL_ADDR" with your module address. If these coincide, set the parameter "ReadDiagErr" of the input driver at TRUE (both when the interrupt comes and goes).
  Using "ReadDiagErr" = TRUE, the input driver is prompted to read the data record 1 of the module via SFC RD_REC and file it in the static parameter "DiagStatus".
  It then sets "ReadDiagErr" = FALSE. The return value of SFC RD_REC is filed in the static parameter "JobErr" (see JobErr-Messages).

If an error display is registered in data record 1, the input driver in the axis DB sets the error codes "Error" and "Err.EncoderErr" at TRUE.

For detailed information on error response, error display and error acknowledgement, see Error Displays and Error Acknowledgement in axis DB.

4.7.6 FB EncoderET200S1Count (FB 26)

Function

The FB encoderET200S1Count allows the use of the module ET 200S 1COUNT 5V or 24V as a distance measuring module for incremental encoders.

The driver can also be used for the isochrone version of this module.

Configuration of module ET 200S 1COUNT in HW Config

The input driver is only suitable for modules "1 CTR 5V/100kHz counting operation" or "1 CTR 24V/100kHz counting operation" from the HW Config catalog.

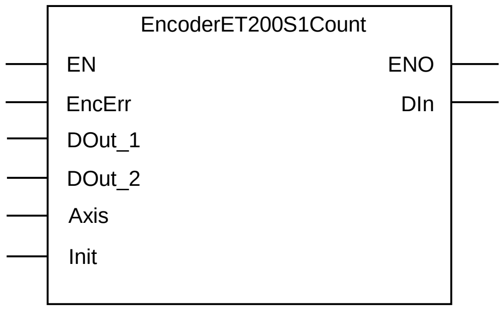
Select the submodule from the module catalog in HW Config under **PROFIBUS-DP > ET 200S > IM 151 > FM**.

Parameter	Setting
Group diagnosis	You can engage this when you wish to make an evaluation in your program.
Type of counting mode	Continuous counting
Function DI, Synchronization	If you wish to make a reference search, select Function DI = synchronization at 0/1 edge and Synchronization = one-time. Otherwise select Function DI = Input
Gate function, Upper counting limit	Arbitrary

Parameterize the other settings according to the connected encoder.

Call

Ladder diagram display



Parameter Description

P type	Parameter	Data type	Meaning
IN	EncErr	BOOL	1 = Encoder error: Here you inform the input driver of an error detected by the user program (e.g. module removed or station failure).
OUT	DOut_1	BOOL	Digital output 1
	DOut_2	BOOL	Digital output 2 (only available with 1Count5V)
	DIn	BOOL	Digital input of module
The following static parameters of the block are only used for diagnostic purposes in the event of an error.			
STAT	Status. ERR_ENCODER	BOOL	Short-circuit / wire break encoder signal (only with 1Count5V)
	Status.ERR_DO2	BOOL	Short-circuit / wire break / overheating DO2 (only with 1Count5V)
	Status. ERR_PARA	BOOL	Configuration error
	Status.ERR_DO1	BOOL	Short-circuit / wire break / overheating DO1
	Status.ERR_24V	BOOL	Short-circuit encoder supply

Synchronization

Synchronization may either occur

- by reference search or
- by reference setting.

Connect the reference point switch to digital input DI.

In a reference search, the module ET 200S 1COUNT can recognize the reference point with a rising edge at digital input DI.

Digital Input

The digital input value of the module is displayed in the output parameter "DIn" of the input driver.

Digital Outputs

The input parameters "DOut_1" and "DOut_2" are output at the corresponding digital outputs.

Error Handling

If the input driver detects an error display of the module, it sets the error codes "Error" and "Err.EncoderErr" in the axis DB and resets the status bit "Sync" in the axis DB.

For detailed information on error response, error display and error acknowledgement, see Error Displays and Error Acknowledgement in Axis DB.

4.7.7 FB EncoderFM350 (FB 27)

Function

FB Encoder FM350 allows the use of FM 350-1 as a distance measurement module for incremental encoders.

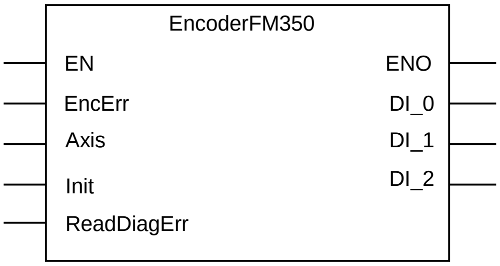
The driver can also be used for the isochrone version of this module.

Configuration of the Module in HW Config

Setting mask	Setting
Diagnostic data	enable
Operating modes	• Counting range limits: -31 to +31 bit, • Operating mode: continuous counting, • Gate control: periodic counting.
Inputs	If you want to make a reference search, you must select the following setting: • Set counter (DI Set): single. • The setting "Evaluate zero mark for setting" may be engaged or disabled as a reference point criterion.

Call

Ladder diagram display



Parameter Description

P type	Parameter	Data type	Meaning
IN	EncErr	BOOL	1 = Encoder error: Here you inform the input driver of an error detected by the user program (e.g. module removed or station failure).
INOUT	ReadDiagErr	BOOL	Set this bit in OB82 if the module has triggered a diagnostic interrupt (see under Error Handling).
OUT	DI_0	BOOL	Digital input 0
	DI_1	BOOL	Digital input 1
	DI_2	BOOL	Digital input 2
The following static parameters of the block are only used for diagnostic purposes in the event of an error.			
STAT	JobErr	INT	Return value of SFC RD_REC if "ReadDiagErr" = TRUE was set.
	DiagStatus. EXT_VOLTAGE	STRUCT	External auxiliary supply defective
	DiagStatus. NO_CONFIG	STRUCT	Configuration missing
	DiagStatus. CONFIG_ERR	STRUCT	Configuration defective
	DiagStatus. WTCH_DOG_ FLT	STRUCT	Time watchdog addressed
	DiagStatus. RAM_FLT	STRUCT	RAM defective
	DiagStatus. CH1_SIGA	STRUCT	Signal A faulty
	DiagStatus. CH1_SIGB	STRUCT	Signal B faulty
	DiagStatus. CH1_SIGZ	STRUCT	Signal N faulty
	DiagStatus. CH1_BETW	STRUCT	Error between channels
	DiagStatus. CH1_5V2	STRUCT	Encoder supply 5.2V faulty

Synchronization

Synchronization may either occur

- by reference search or
- by reference setting.

Connect the reference point switch to digital input I2.

With a reference search, the module can detect the reference point optionally in the following cases:

- With a rising edge at digital input I2
- By the rising edge of the zero mark, if digital input I2 is set.

If the axis has already reached the reference point switch at the time the reference point approach is started, its position is referenced to the current reference position.

Digital Inputs

Digital inputs I0 to I2 are displayed in the output parameters "DI_0" to "DI_2" of the input driver.

Error Handling

The module only provides its error information via data record 1 (DS1).

In the following situations the input driver reads data record 1 from the module via SFC RD_REC and writes it to the parameter "DiagStatus":

- The input driver is being initialized.
- The module reports an error.
- The driver input parameter "ReadDiagErr" is TRUE.

Then it sets "ReadDiagErr" = FALSE. The return value of SFC RD_REC is written to the static Parameter "JobErr" (see JobErr Messages).

If an error display is registered in data record 1, the input driver in the axis DB sets the error codes "Error" and "Err.EncoderErr" at TRUE. The status bit "Sync" is set at FALSE.

For detailed information on error response, error display and error acknowledgement, see Error Displays and Error Acknowledgement in axis DB.

4.7.8 FB EncoderCPU314C (FB 28)

Function

The FB EncoderCPU314C interconnects a 24-V position encoder with a counter channel of the CPU 314C.

Up to four channels are available.

Configuring the counter function of the CPU 314C in HW Config

The input driver is compatible with CPU 314C => firmware version V2.0 only.

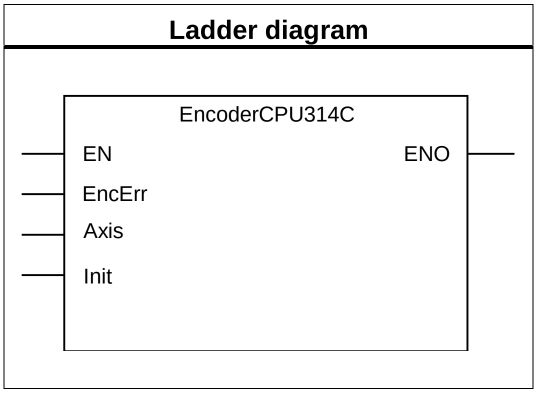
Set the following parameters for each channel in the properties dialog of the module in slot 2.4 (count):

Parameter	**Setting**
Mode	Infinite count
HW gate	No HW gate

Customize the other settings according to the encoder used and to the requirements for your application.

At the axis data parameter "Start address of inputs of the position monitoring module" ("InputModuleInAddress"), enter the address you have assigned in HW Config to the inputs at slot 2.4 (count). Set the address of the counter outputs at the " Start address of outputs of the position monitoring module " ("InputModuleOutAddress").

Call



Parameter description

P type	Parameter	Data type	Meaning
IN	EncErr	BOOL	1 = Encoder error. Here you report errors detected by the user program to the input driver.

Synchronization

You can synchronize either

- with a reference point approach or
- by setting the reference point.

Connect the reference point switch to the digital latch input of the corresponding channel (DI1.4 for channel 0 up to DI1.7 for channel 3).

The 314C CPU can detect the reference point during a reference point approach by a rising edge at the digital input DI.

If the axis has already reached the reference point switch at the time the reference point approach is started, its position is referenced to the current reference position.

During a reference point approach the driver also polls the status of the reference point switch. The time required for these accesses increases the block runtime considerably.

Maximum Counting Frequency

Note

The counter of the 314C CPU does not output any values if its maximum counting frequency is exceeded (see the manual "*CPU 31xC Technological Functions"*). The axis is halted and the error message "Following Distance exceeded " is output.

Hence, you must always conform to the frequency limit of the counter when you set the maximum velocity ("MaxVelocity") in the axis data.

Error handling

The CPU 314C does not output encoder error information.

4.7.9 FB EncoderUniversal (FB 29)

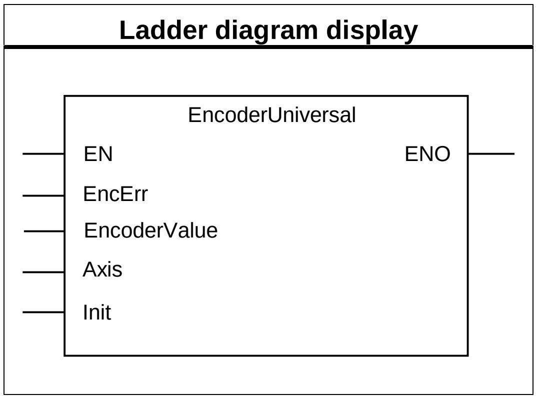
Function

FB EncoderUniversal is used to connect distance measurement modules for which Easy Motion Control does not provide any special input driver.

Configuration of the Module in HW Config

Select parameters according to their hardware and the encoder used.

Call



Ladder diagram display

Parameter Description

P type	Parameter	Data type	Meaning
IN	EncErr	BOOL	1 = Encoder error: Here you inform the input driver of an error detected by the distance measurement module or user program (e.g. module removed or station failure).
	EncoderValue	DINT	Encoder value

Method of Operation

At its input "EncoderValue", the FB EncoderUniversal waits for a value corresponding to the actual position of your axis. It does not access the I/Os itself. You can interconnect the value provided by your distance measurement module directly with the driver or if necessary even convert it beforehand.

If your distance measurement module offers special functions which you wish to use, you should integrate this in your user program.

Synchronization

With the FB EncoderUniversal, synchronization is only possible by reference setting.

Error Handling

Any errors detected by your distance measurement module you should record and process further in your user program.

If such an error impairs the functions which you are performing with Easy Motion Control blocks, you should incorporate this error into Easy Motion Control error handling via the input "EncErr".

For detailed information on error response, error display and error acknowledgement, see Error Displays and Error Acknowledgement in axis DB.

4.8 Output Drivers (FB 31 to FB 37)

Function

The output driver is used to output the velocity setpoint value of an axis calculated by the controller to the connected output module. The FB converts the velocity from [length/s] to the corresponding value range of the output module and outputs this value.

If the input "EnableDrive" = FALSE, the output driver outputs zero to the output module.

Initialization

The output value is set at zero.

Call

The output driver of an axis must be launched unconditionally.

Isochrone mode

The output drivers support isochrone mode at all suitable DP modules. The driver automatically recognizes isochrone mode if it is called in a synchronous cycle OB.

Error Handling

An error detected by the user program (e.g. module removed or station failure) may be notified to the output driver at the input "OutErr". Provided the input "OutErr" = TRUE, the output driver does not access the output module.

If the input "OutErr" is set at TRUE, the output driver sets the error codes "Error" and "Err.OutputErr" in the axis DB.

For detailed information on error response, error display and error acknowledgement, see Error Displays and Error Acknowledgement in axis DB.

Simulation Mode

Provided the bit "Sim" is set in the axis DB, the output driver does not access the output module (see FB MC_Simulation).

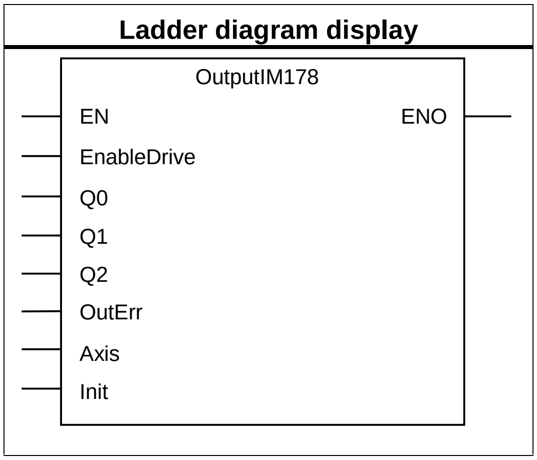
4.8.1 FB OutputIM178 (FB 31)

Function

FB OutputIM178 allows the use of a channel of IM 178-4 as output module.

An identical channel number and axis must be set at the IM178-4 input AND output drivers, since certain addresses refer to both drivers. A combination with other drivers is not possible.

Call



Parameter Description

P type	Parameter	Data type	Meaning
IN	EnableDrive	BOOL	1 = Enabling of analog output If "EnableDrive" = FALSE, the value zero is transferred to the output module. This input can be linked with the "DriveEnabled" output of FB MC_Control.
	Q0	BOOL	Digital output 0 of selected channel
	Q1	BOOL	Digital output 1 of selected channel
	Q2	BOOL	Digital output 2 of selected channel
	OutErr	BOOL	1 = Error on the output module. An error detected by the user program (e.g. module removed or station failure) may be notified to the output driver here.

Digital Outputs

Digital outputs "Q0" to "Q2" of the selected channel of IM 178-4 can be controlled via input parameters "Q0" to "Q2" of the output driver.

4.8.2 FB OutputSM432 (FB 32)

Function

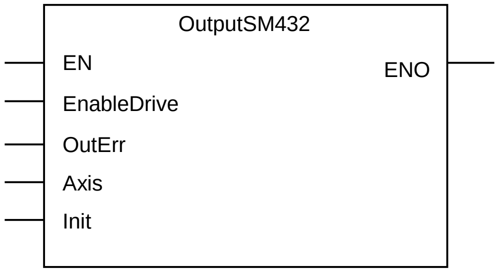
FB Output SM432 allows the use of a channel of SM 432 as output module.

Configuration of SM 432 in HW Config

Parameterize the corresponding channel of SM 432 to ±10V output voltage.

Call

Ladder diagram display



Parameter Description

P type	Parameter	Data type	Meaning
IN	EnableDrive	BOOL	1 = Enabling of analog output If "EnableDrive" = FALSE, the value zero is transferred to the output module. This input can be linked with the "DriveEnabled" output of FB MC_Control.
	OutErr	BOOL	1 = Error on the output module. An error detected by the user program (e.g. module removed or station failure) may be notified to the output driver here.

4.8.3 FB OutputET200S2AO (FB 33)

Function

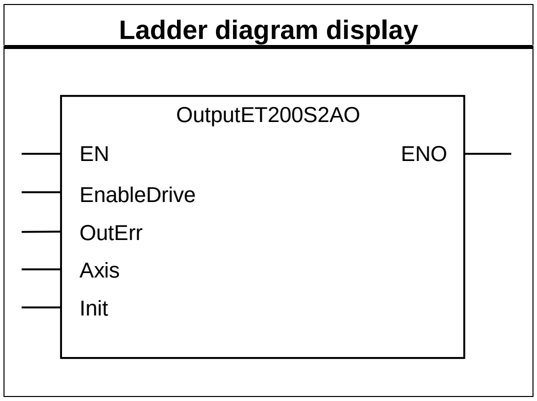
FB OutputET200S2AO allows the use of a channel of ET 200S analog output modules 2AO U as output module.

This driver can also be used for the isochrone versions of this module.

Configuration of ET 200S Analog Output Module 2AO U in HW Config

Parameterize the corresponding channel of the module to ±10V output voltage.

Call



Ladder diagram display

Parameter Description

P type	Parameter	Data type	Meaning
IN	EnableDrive	BOOL	1 = Enabling of analog output If "EnableDrive" = FALSE, the value zero is transferred to the output module. This input can be linked with the "DriveEnabled" output of FB MC_Control.
	OutErr	BOOL	1 = Error on the output module. An error detected by the user program (e.g. module removed or station failure) may be notified to the output driver here.

4.8.4 FB OutputCPU314C (FB 34)

Function

FB OutputCPU314C is used to control a ±10-V analog output channel of the 314C CPU.

Configuring the analog outputs of the 314C CPU in HW Config

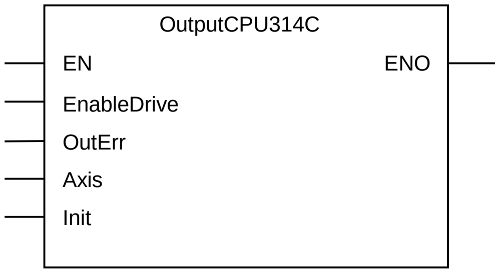
Set the output voltage of the corresponding analog output channel of the 314C CPU to ±10 V.

In the axis data parameter "Start address of the outputs of the output module" ("OutputModuleOutAddress"), set the same address you have assigned in HW Config for the outputs at slot 2.3 (AI5/AO2). You do not need to configure the " Start address of the inputs of the output module " ("OutputModuleInAddress").

Up to two analog output channels are available, allowing you full control over two axes via the CPU 314C.

Call

Ladder diagram display



Parameter description

P type	Parameter	Data type	Meaning
IN	EnableDrive	BOOL	1 = Enable analog output If "EnableDrive" = FALSE, a zero value is transferred to the analog output. This input can be interconnected with the "DriveEnabled" output of FB MC_Control.
	OutErr	BOOL	1 = Analog output error. Errors detected by the user program can here be reported to the output driver.

4.8.5 FB OutputSM332 (FB 35)

Function

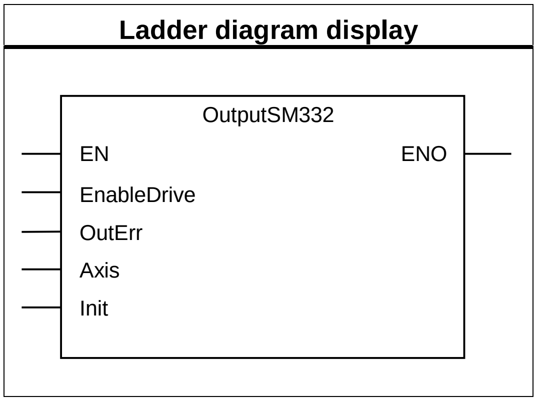
FB Output SM332 allows the use of a channel of SM 332 as output module.

This driver can also be used for the isochrone versions of this module.

Configuration of SM 332 in HW Config

Parameterize the corresponding channel of SM 332 to ±10V output voltage.

Call



Parameter Description

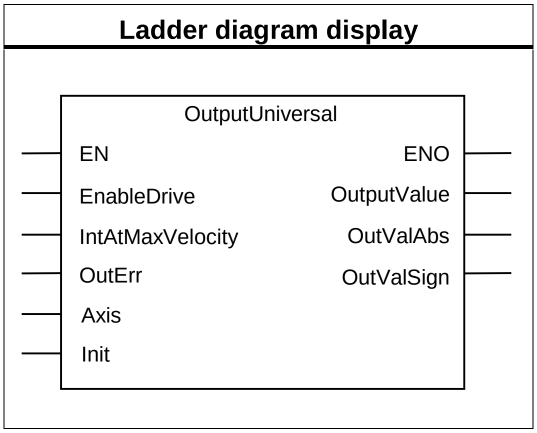
P type	Parameter	Data type	Meaning
IN	EnableDrive	BOOL	1 = Enabling of analog output If "EnableDrive" = FALSE, the value zero is transferred to the output module. This input can be interconnected with the "DriveEnabled" output of FB MC_Control.
	OutErr	BOOL	1 = Error on the output module. An error detected by the user program (e.g. module removed or station failure) may be reported to the output driver here.

4.8.6 FB OutputUniversal (FB 36)

Function

FB OutputUniversal allows you to use an arbitrary (analog) output module together with Easy Motion Control blocks.

Call



Parameter Description

P type	Parameter	Data type	Meaning
IN	EnableDrive	BOOL	1 = Enabling of analog output As soon as "EnableDrive" = FALSE, the value zero is output at the outputs. This input can be linked with the "DriveEnabled" output of FB MC_Control.
	IntAtMaxVelocity	INT	Integer value for converting the velocity setpoint value (see below).
	OutErr	BOOL	1 = Error on the output module. An error detected on the output module by the user program (e.g. module removed or station failure) may be notified here to the output driver
OUT	OutputValue	INT	Calculated speed setpoint value
	OutValAbs	INT	Absolute speed setpoint value
	OutValSign	BOOL	Sign of speed setpoint value (TRUE = negative value)

Method of Operation

With the help of the proportionality factor "IntAtMaxVelocity", FB OutputUniversal calculates a speed setpoint value from the velocity output by the controller and presents this at its output bar.

The driver does **not** access the I/Os itself.

"IntAtMaxVelocity" is the value which you have to use to control your analog output module in order for this to output a voltage which will move the axis with "MaxVelocity" (you have already determined the value of this voltage in the parameter "DriveInputAtMaxVel").

The driver curve always goes through zero and runs linearly.

The speed preset value is output both as an integer value with sign as well as an amount with a separate sign.

If these output values already correspond to your application, you can output them directly at the I/O via the user program. If the values are not yet suitable, you can also make an additional adjustment via the user program and output the value that is thereby generated.

4.8.7 FB OutputMM4_DP (FB 37)

Function

FB OutputMM4DP is used for the implementation of a Micromaster 4 with optional DP add-on module.

Configuring the MM4 in HW Config

Information on the Micromaster 4 is found in the STEP 7 HW catalog under:
PROFIBUS DP > Further FIELD DEVICES > Drives > SIMOVERT > MICROMASTER 4.
First you may have to install the corresponding GSD file.

Install the Micromaster at your DP master system and assign it a DP address. Insert a "0 PKW, 2 PZD (PPO3)" module at slots 0 and 1. Either accept the default I/O addresses or select your own values.

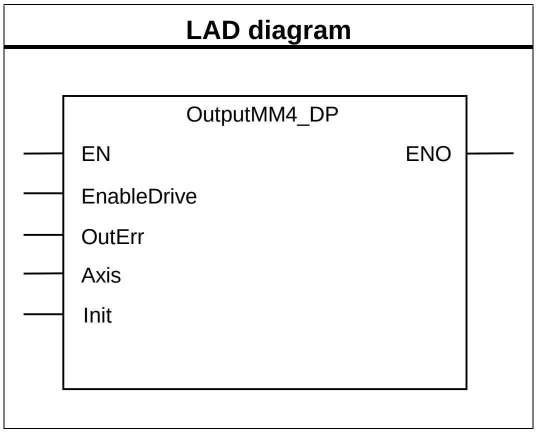
Enter these addresses in "Start address of the outputs of the output module" or "Start address of the inputs of the output module" in the axis OB of your axis.

Configuring the MM4

Customize the Micromaster configuration to suit your requirements (motor data, DP address, operation as DP slave, ...) as shown in your Micromaster documentation.

Enter the "reference frequency" specified in the P2000 parameter as "reference value for 100% speed" ("DriveInputAt100") in the corresponding axis DB.

Call



Parameter description

P type	Parameter	Data type	Meaning
IN	EnableDrive	BOOL	1 = Enable the analog output When "EnableDrive" becomes FALSE, the zero speed value is output to the Micromaster. This input can be interconnected with the "DriveEnabled" output of FB MC_Control.
	OutErr	BOOL	1 = Output error An error detected on the output module by the user program (e.g. module removed or station failure) may be notified here to the output driver

4.9 FB MC_Control (FB 11)

Function

FB MC_Control allows you to implement position control together with the input and output drivers.

FB MC_Control can accept the following controller statuses:

- Closed-loop control
- Tracking control
- Manual control

Closed-loop control

FB MC_Control

- Calculates the velocity setpoint value for the output driver from the position setpoint of a travel FB and the actual position of the input driver
- Monitors
  - stoppage
  - target approach
  - following distance
- Processes error acknowledgement (see Error Displays and Error Acknowledgement

Tracking control

In this state, the setpoint position is corrected according to the actual position. Tracking control becomes active when the input "EnableDrive" = FALSE or after an axis error with hard stop (see Error Displays and Error Acknowledgement). In tracking mode, the output "DriveEnabled" = FALSE.

Manual control

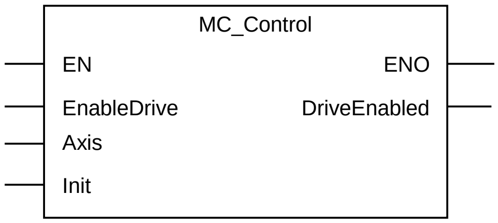
In this state, the manual setpoint value "ManVelocity" is output as velocity setpoint value. This is limited to "MaxVelocity". The setpoint position is corrected according to the actual position (see Position Controller Data).
This state is switched on or off using "ManEnable".

Warning
Manual control is possible at any time, even if the axis is in error status.

Call

Call FB MC_Control once per axis unconditionally in the selected OB.

Ladder diagram display



Parameter Description

P type	Parameter	Data type	Meaning
IN	EnableDrive	BOOL	0 = disable drive, tracking control is active. 1 = enable drive, position control is active. If there is no error, "DriveEnabled" is then set.
OUT	DriveEnabled	BOOL	Drive enable, position control is active. If "EnableDrive" is set and there is no error, "DriveEnabled" is set. if "DriveEnabled" is not set, the axis is in tracking mode and there is no monitoring of the axis for stoppage. This signal should be used in order to control the enable input on the power unit.

Method of Operation

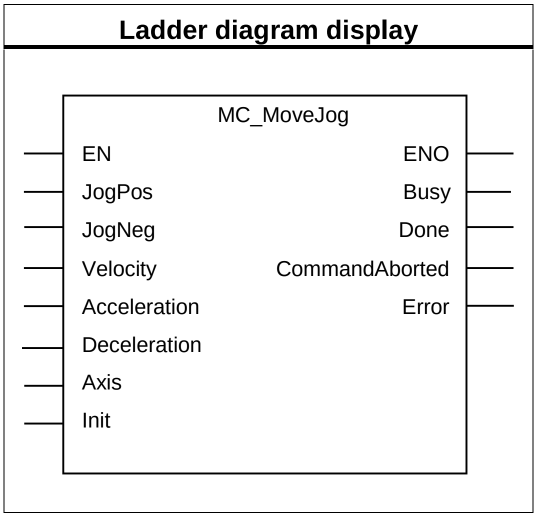
The control algorithm of FB MC_Control is a P-controller, which is configured via the parameter "controller gain" ("FactorP", see Position Controller Data).

4.10 FB MC_MoveJog (FB 3)

Function

FB MC_MoveJog allows you to move an axis using two level-controlled directional inputs "JogPos" and "JogNeg" (jogging).

Call



Parameter Description

P type	Parameter	Data type	Meaning
IN	JogPos	BOOL	1 = Movement in positive direction provided "JogPos" is set.
	JogNeg	BOOL	1 = Movement in negative direction provided "JogNeg" is set.
	Velocity	REAL	Axis velocity in [unit of length/s]
	Acceleration	REAL	Axis acceleration in [unit of length/ s^2]
	Deceleration	REAL	Axis deceleration in [unit of length/ s^2]
OUT	Busy	BOOL	1 = Job is busy.
	Done	BOOL	1 = Job completed without error. After reaching axis standstill, the bit is set for exactly one block call.
	CommandAborted	BOOL	1 = Job was aborted by another order. The bit remains active until "JogPos" or "JogNeg" is reset.
	Error	BOOL	1 = The job was aborted due to an axis error. The bit remains set until "JogPos" or "JogNeg" is reset.

Method of Operation

If you set a directional input, MC_MoveJog first accelerates the axis to the indicated velocity. Provided the directional input is set, the axis moves in this direction. If you reset the directional input again, the axis slows down.

The two directional inputs are evaluated with the following dependencies;

- Provided a direction is selected, setting the opposite direction has no effect. No error is produced.
- To change direction during travel, set the other directional input at TRUE and at the same time the originally selected directional input at FALSE.
- If both inputs are set at the start of a motion, this produces a start error ("Err.StartErr", see Error with Soft Stop).

Special features with a linear axis:

- If the maximum travel distance during travel with a linear axis is exceeded, FB MC_MoveJog slows down the axis to a standstill with an acknowledgeable error message (see Sequence of Travel Movements). Then the axis can be moved again after the error has been acknowledged.
- A **non-synchronized** linear axis is not monitored and can therefore travel right to the physical end of the axis without slowing down. You must therefore complete the motion in time.
- A **synchronized** linear axis is slowed down towards the end of the work range and stops at the software limit switch. The acknowledgeable error message "Err.DistanceErr" and the error message "Err.SWLimitMinExceeded" or "Err.SWLimitMaxExceeded" is output. After the error has been acknowledged, it is only possible to travel in the opposite direction.

Warning

Injury to persons and property may occur.

With a non-synchronized linear axis, FB MC_MoveJog may travel right up to the physical end of the axis.

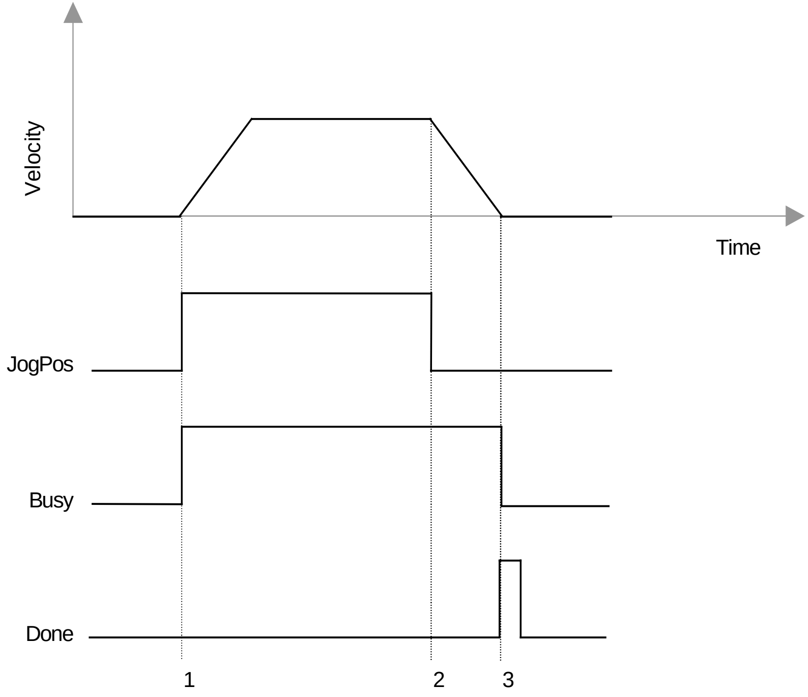
To avoid injury to persons and objects, take the following precautions:

- Install an EMERGENCY STOP switch in the vicinity of the computer. Only in this way can you ensure that the system is reliably switched off in the event of a computer or software failure.
- Install safety limit switches which have direct control over the power units of all drives.
- Ensure that nobody has access to the area of the system where there are moving parts.

Special features with a rotary axis:

- The travel distance of FB MC_MoveJog is not restricted with a rotary axis (continuous turning).

Signal flow diagram for FB MC_MoveJog:



1 Movement is started by setting "JogPos".

2 Movement is stopped by resetting "JogPos".

3 With axis standstill, "Busy" is reset and "Done" is set for a cycle.

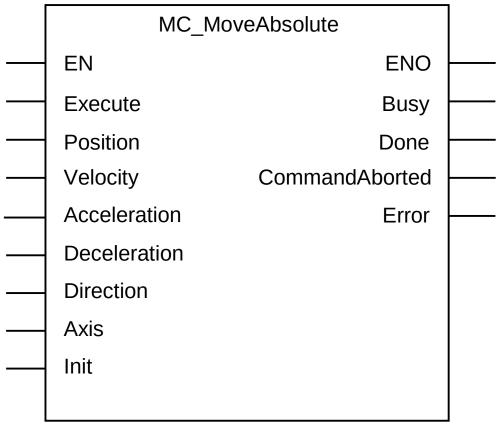
4.11 FB MC_MoveAbsolute (FB 1)

Function

FB MC_MoveAbsolute allows you to move a synchronized axis to the absolute target specified in "Position".

Call

Ladder diagram display



Parameter Description

P type	Parameter	Data type	Meaning
IN	Execute	BOOL	Start of positioning with a positive edge.
	Position	REAL	Absolute target position in [unit of length] With rotary axis: start of rotary axis <= position < end of rotary axis
	Velocity	REAL	Axis velocity in [unit of length/s]
	Acceleration	REAL	Axis acceleration in [unit of length/ s^2]
	Deceleration	REAL	Axis deceleration in [unit of length/ s^2]
	Direction	INT	Polarity selection for travel with rotary axis -1 = Negative direction 0 = Shortest distance 1 = Positive direction

P type	Parameter	Data type	Meaning
OUT	Busy	BOOL	1 = Order is running.
	Done	BOOL	1 = Order completed without error. The bit remains active until "Execute" is reset, but at least for one block call.
	CommandAborted	BOOL	1 = Order was interrupted by another order. The bit remains active until "Execute" is reset, but at least for one block call.
	Error	BOOL	1 = The job was aborted due to an axis error. The bit remains active until "Execute" is reset, but at least for one block call.

Method of Operation

On detecting a positive edge at the "Execute" input, FB MC_MoveAbsolute determines the direction of travel and accelerates the axis in this direction. On reaching the point of deceleration, the axis is slowed down. The specified speed does not have to be reached when traveling.

Special feature with a linear axis:

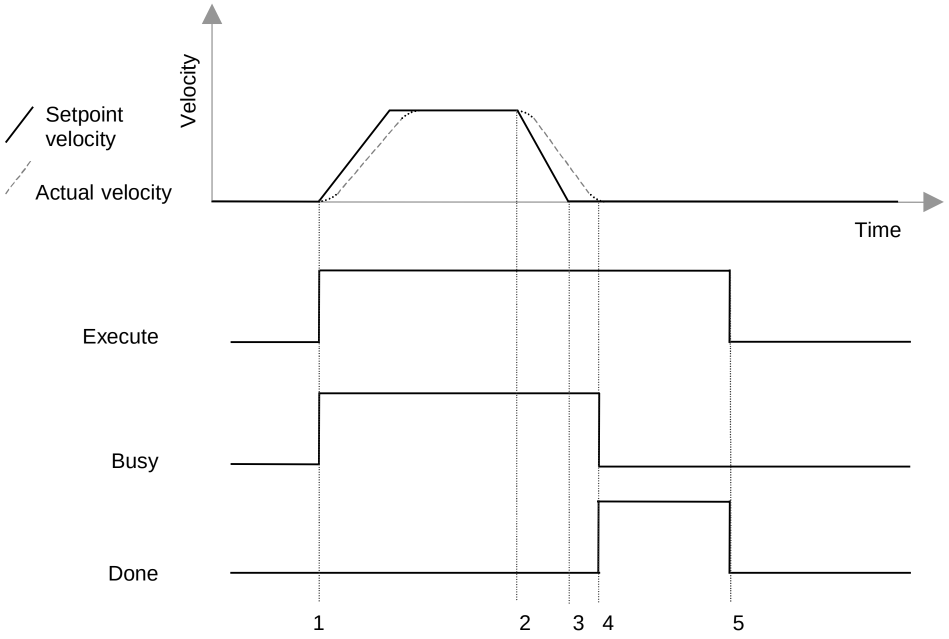
- Direction of travel is determined from the setpoint position at the start and the target.

Special features with a rotary axis:

- FB MC_MoveAbsolute only allows you to move one way from axis standstill, namely the distance that is less than one rotary axis revolution. For positioning over several rotary axis revolutions, use the FB MC_MoveRelative.
- Polarity selection when starting from axis standstill:
  - Positive direction: target approached in a positive direction
  - Negative direction: target approached in a negative direction
  - Shortest distance: target approached in the direction in which the distance between starting point and target is less or equal to half a rotary axis revolution.
- Polarity selection when overriding a motion
  - Positive direction: target approached in the positive direction. If the previous motion was in the negative direction, the axis is slowed down and the target approached in the positive direction. If the target is reached exactly when slowing down with a negative travel direction, the axis then travels one revolution in the positive direction.

- Negative direction: target approached in the negative direction. If the previous motion was in the positive direction, the axis is slowed down and the target approached in the negative direction. If the target is reached exactly when slowing down with a positive travel direction, the axis then travels one revolution in the negative direction.
- Shortest distance: target is approached in the direction in which the distance between starting point and target is less or equal to half a rotary axis revolution. If the braking distance is greater, the axis travels to the target in the original direction.
- If you have selected travel parameters so that the braking distance of the previous motion is greater than one rotary axis revolution, the target is approached directly after changing direction.
- The entire distance between starting point and target may be greater than one rotary axis revolution.

Signal flow diagram for FB MC_MoveAbsolute:



1 Start of the Travel FB

2 Brake location

3 Setpoint value has been reached.

4 Axis has reached the target range.

5 By resetting "Execute", "Done" will be reset. The procedure is completed.
If "Execute" is reset before the target position has been reached, "Done" will be reset for a cycle and then be reset again when the target position is reached.

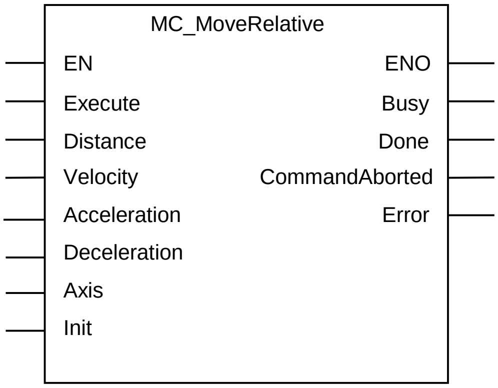
4.12 FB MC_MoveRelative (FB 2)

Function

FB MC_MoveRelative allows you to move an axis by a specified "distance" in relation to the preset value at the start of travel. The "distance" sign determines the direction.

Call

Ladder diagram display



Parameter Description

P type	Parameter	Data type	Meaning
IN	Execute	BOOL	Start of positioning with a positive edge.
	Distance	REAL	Distance to be traveled in [unit of length] in relation to the last position setpoint. The sign determines the direction of travel.
	Velocity	REAL	Axis velocity in [unit of length/s]
	Acceleration	REAL	Axis acceleration in [unit of length/ s^2]
	Deceleration	REAL	Axis deceleration in [unit of length/ s^2]
OUT	Busy	BOOL	1 = Order is running.
	Done	BOOL	1 = Order completed without error. The bit remains active until "Execute" is reset, but at least for one block call.
	CommandAborted	BOOL	1 = Order was interrupted by another order. The bit remains active until "Execute" is reset, but at least for one block call.
	Error	BOOL	1 = The job was aborted due to an axis error. The bit remains active until "Execute" is reset, but at least for one block call.

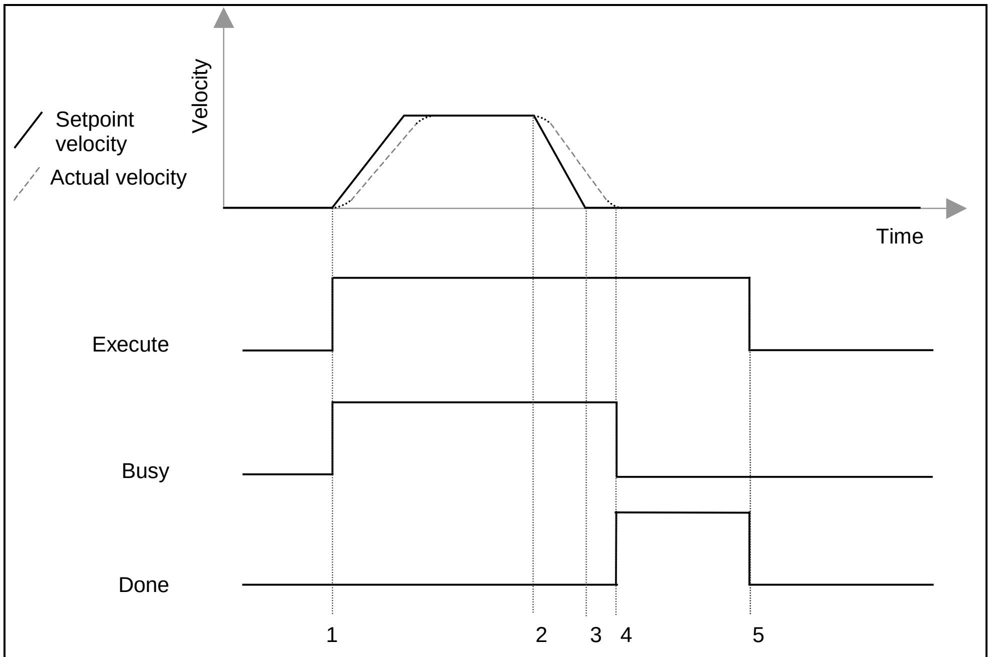
Method of Operation

On detecting a positive edge at the "Execute" input, FB MC_MoveRelative accelerates the axis in the direction indicated in "Distance". On reaching the point of deceleration, the axis is slowed down. The specified speed does not have to be reached during travel. The target is determined from the setpoint position at the start of travel and the distance to be traveled.

Special feature with rotary axes:

When values greater than one rotary axis revolution are specified in "distance", travel can be realized over several rotary axis revolutions.

Signal flow diagram for FB MC_MoveRelative:



1 Start of the Travel FB

2 Brake location

3 Setpoint value has reached the target position.

4 Axis has reached the target range.

5 By resetting "Execute", "Done" will be reset. The procedure is completed.
If "Execute" is reset before the target position has been reached, "Done" will be reset for a cycle and then be reset again when the target position is reached.

4.13 FB MC_Home (FB 4)

Function

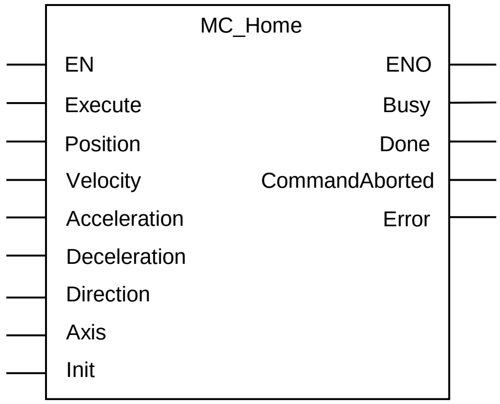
FB MC_Home is used

- to search for reference in the case of an incremental encoder "velocity" > 0) and
- to set a reference in the case of an incremental and absolute encoder ("Velocity" = 0).
  With an absolute encoder, reference setting is also known as absolute encoder adjustment.

Call

It is only permitted to start the reference search or reference setting when the axis is at a standstill.

Ladder diagram display



Parameter Description

P type	Parameter	Data type	Meaning
IN	Execute	BOOL	Start of reference search or reference setting with a positive edge.
	Position	REAL	**Incremental encoder** Reference point coordinates. **Absolute encoder** Reference point coordinates. The value must be within the work range: • **Linear axis** within the software limit switch • **Rotary axis** Axis start <= Position < Axis end
	Velocity	REAL	Axis velocity in [unit of length/s] Velocity > 0 Reference search Velocity = 0 Reference setting
	Acceleration	REAL	Axis acceleration in [unit of length/ s^2]
	Deceleration	REAL	Axis deceleration in [unit of length/ s^2]
	Direction	INT	Start direction for reference search. 1 = Reference search in positive direction -1 = Reference search in negative direction
OUT	Busy	BOOL	1 = Order is running.
	Done	BOOL	1 = Order completed without error. The bit remains active until "Execute" is reset, but at least for one block call.
	CommandAborted	BOOL	1 = Order was interrupted by another order. The bit remains active until "Execute" is reset, but at least for one block call.
	Error	BOOL	1 = The job was aborted due to an axis error. The bit remains active until "Execute" is reset, but at least for one block call.

Method of Operation

Reference Search ("velocity" > 0)

After "Execute" is set, the axis travels at the selected velocity and direction until the distance measurement module and the input driver recognize that the reference point is reached. When the reference point is reached, the actual position is set at the "position" value and the axis is slowed down. The axis is now synchronized ("Sync" = TRUE), but does not stand on the reference point.

If the maximum travel distance is exceeded in the reference search, FB MC_Home slows down the axis to a standstill (see Sequence of Travel Movements) and the error displays "Error" and "Err.DistanceErr" are set (see Error with Soft Stop).

If the reference search is stopped before the reference point is reached, the axis is unsynchronized ("Sync" = FALSE).

FB MC_Home neither stops at the axis end, nor does it automatically reverse the direction of travel.

Danger

In the event that no reference is found, you must take suitable measures to ensure that the axis comes to a standstill before the physical end (e.g. EMERGENCY STOP switch, safety limit switch).

Reference setting ("velocity" = 0)

Set "velocity" at 0. After "Execute" is set, the actual position is set at the "position" value, at the same time in the axis DB the status bit "Sync" = TRUE.

After exchanging the absolute encoder or any mechanical changes, you must reset the reference point, even if the axis is still synchronized.

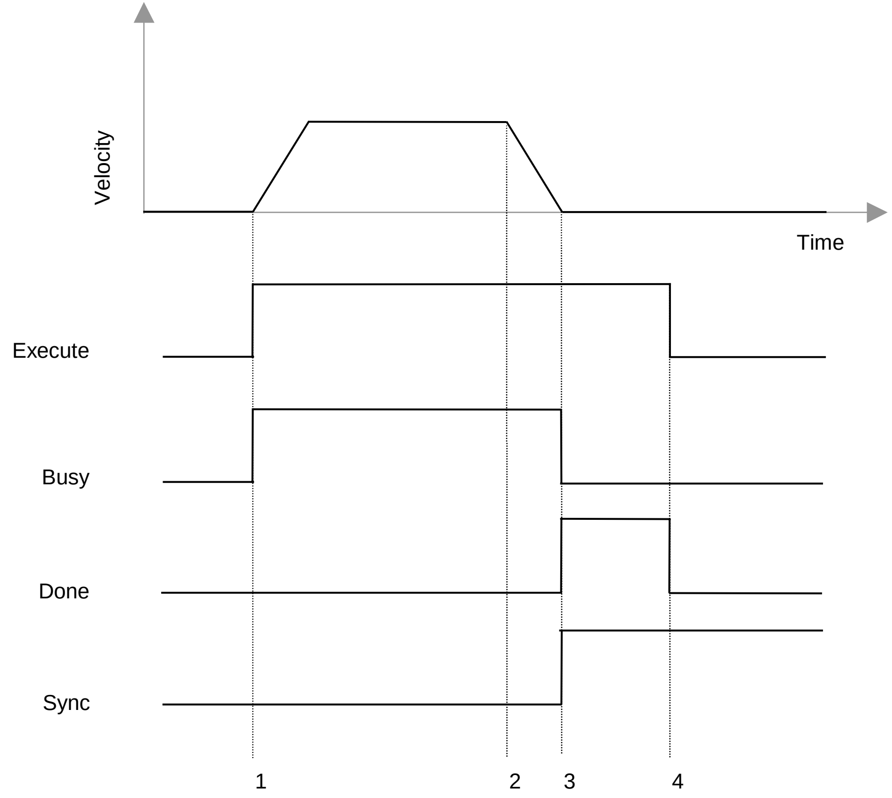
Note

With an absolute encoder the following applies:

The encoder value at the reference point is saved in the axis DB. For this reason, after setting the reference point, you should transfer the online axis DB to your offline database organization. Otherwise the axis is no longer synchronized after loading the DB into the CPU.

Please observe the information on Loading the Axis DB.

Signal flow diagram for reference search:



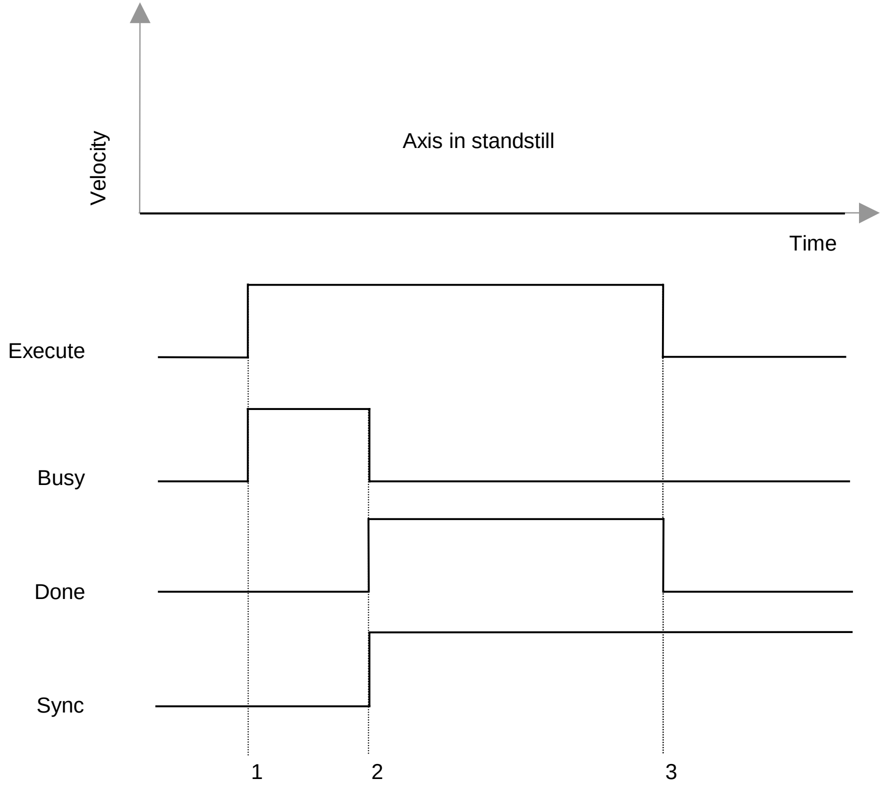
1 Start of the Reference-point approach with "Execute".

2 Reference-point approach found, the axis brakes.

3 Axis comes to a standstill, "Busy" is cancelled and "Done" is set. The axis is synchronized.

4 By resetting "Execute", "Done" is reset.

Signal flow diagram for reference setting:



1 Start of the setting of reference point or the absolute encoder adjustment with "Execute".

2 The axis is synchronized.

3 By resetting "Execute", "Done" is reset. The procedure is completed.

4.14 FB MC_StopMotion (FB 5)

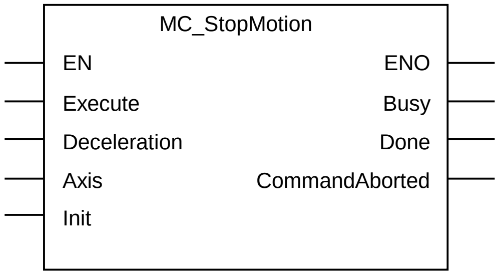
Function

FB MC_StopMotion allows you to stop any travel FB so that the axis comes to a standstill with the set deceleration.

Typical application: aborting positioning at any time due to an external event (e.g. safety door).

Call

Ladder diagram display



Parameter Description

P type	Parameter	Data type	Meaning
IN	Execute	BOOL	Start of braking process with a positive edge.
	Deceleration	REAL	Axis deceleration for deceleration slope in [unit of length/ s^2] When values <= 0.0 are entered, the maximum deceleration value is used from the axis DB.
OUT	Busy	BOOL	1 = Job is busy.
	Done	BOOL	1 = Job completed without error. The bit remains active until "Execute" is reset, but at least for one block call.
	CommandAborted	BOOL	1 = Job was cancelled due to manual intervention at the controller.
	Error	BOOL	1 = Job was aborted due to an axis error. The bit remains active until "Execute" is reset, but at least for one block call.

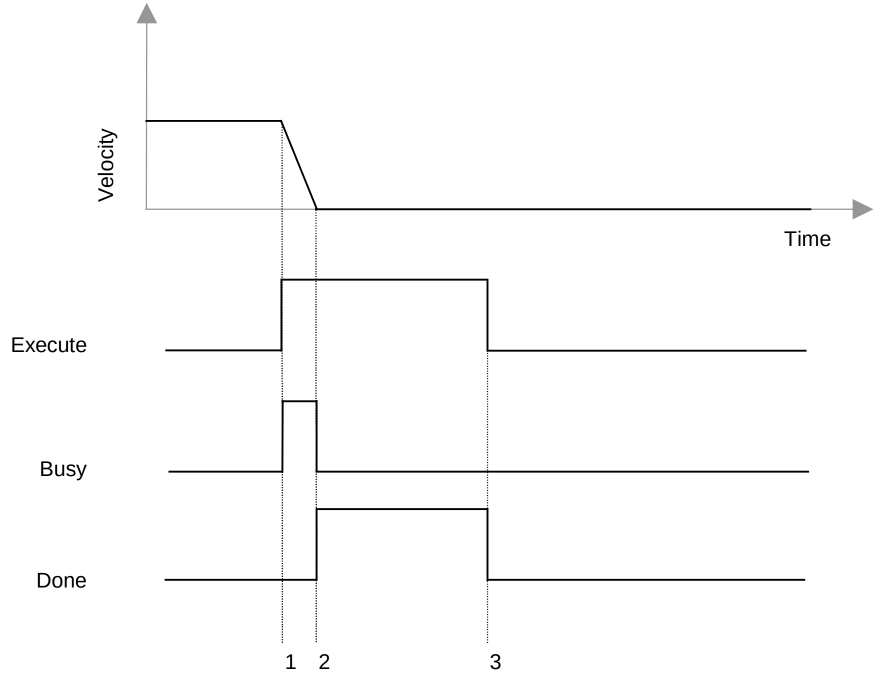
Method of Operation

On detecting the positive edge at the "Execute" input, FB MC_StopMotion takes over control of the axis and slows it to a standstill. FB MC_StopMotion can only be interrupted by hard errors and not by a travel FB. On reaching standstill, a new travel movement can be started.

The deceleration at the FB MC_StopMotion can be selected arbitrarily. If it is less than the deceleration of the interrupted block, the axis is slowed down with the deceleration of the interrupted block. This ensures that the axis does not travel beyond the target of the interrupted block.

If you interrupt a motion with target (FB MC_MoveAbsolute, FB MC_MoveRelative) using FB MC_StopMotion , the latter continues to calculate the residual distance. On termination of FB MC_StopMotion , the residual distance indicates the distance of the immediate setpoint position from the target of the interrupted block. You can assign this value to an FB MC_MoveRelative in order to move the axis to the original target after the interruption.

Signal flow diagram for FB MC_StopMotion :



1 During a movement in process, the Stop module is started.

2 By reaching the standstill, the FB MoveStop sets the output "Done".

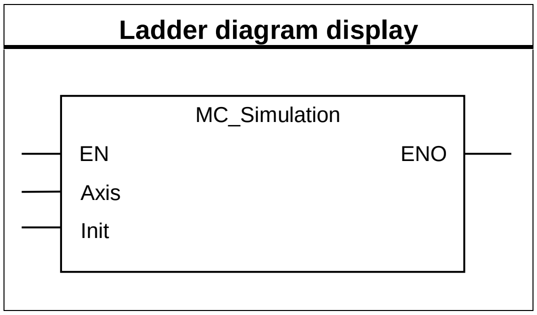
3 By resetting "Execute", the Stop module is terminated.

4.15 FB MC_Simulation (FB 12)

Function

FB MC_Simulation allows you to test your travel program including driver blocks. You do not require any I/Os and your axis is not moved.

Call



Method of Operation

FB MC_Simulation loops back the controller output value onto the input driver, by simulating an encoder value corresponding to the encoder type selected in axis DB.

Engage the simulation mode by setting the bit "Sim" = TRUE in the axis DB.

When simulation is engaged, a reference search is not possible. During simulation, parameterize the "Velocity" input of FB MC_Home with 0.0. This allows a reference to be set instead of the reference search.

If you set a reference point during simulation mode, the reference point of your real axis is altered. It is therefore necessary to resynchronize your axis after switching off simulation operation.

On switching simulation on and off, the axis is brought into the stop status requiring acknowledgement (see Errors with Hard Stop) and all the bits in the bit field "Init.Ix" in axis DB are also set.

If simulation mode is engaged

- the output driver does not output any value to the I/Os and
- the input driver does not read in any encoder value from the I/Os.

Danger

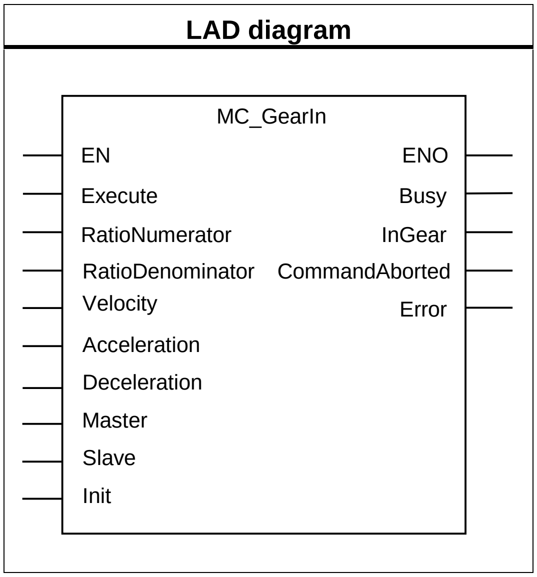
The axis must be at a standstill when simulation mode is engaged.

4.16 FB MC_GearIn (FB 41)

Function

FB MC_GearIn can be used to interconnect the speed setpoint of a slave axis via a gear factor to the speed setpoint of a master axis (setpoint coupling).

Call



Parameter description

P type	Parameter	Data type	Meaning
IN	Execute	BOOL	Start of the geared run at a positive edge
	RatioNumerator	REAL	Gear ratio counter Gear ratio = "RatioNumerator" / "RatioDenominator" If "RatioNumerator" < 0, the motion of the master axis reverses the motion of the slave axis. "RatioNumerator" = 0 is rejected with "DataErr" = 1. You can modify the effective gear ratio of the slave axis by overriding the active gear block with another instance that contains a modified gear ratio.
	RatioDenominator	INT	The gear ratio denominator must be an integer and = 1 You can modify the effective gear ratio of the slave axis by overriding the active gear block with another instance that contains a modified gear ratio.
	Velocity	REAL	Axis velocity in [length unit/s] The maximum geared velocity of the slave axis is limited to this value.
	Acceleration	REAL	Axis acceleration in [length units/s^2] The maximum geared acceleration of the slave axis is limited to this value. This parameter is used if: • The slave axis is coupled to a faster master axis • The acceleration of the master axis exceeds the capability of the slave axis
	Deceleration	REAL	Deceleration in [length units/s^2] The maximum geared deceleration of the slave axis is limited to this value. This parameter is used if: • A moving slave axis is coupled to a slower master axis • The slave axis is brought to a stop due to an error • The brake torque exceeds the capability of the slave axis
INOUT	Master	AXIS_REF	Reference to the master axis data
	Slave	AXIS_REF	Reference to the slave axis data
	Init	BOOL	When Init = TRUE, the block executes its initialization and then resets the bit.

P type	Parameter	Data type	Meaning
OUT	Busy	BOOL	1 = Job is busy.
	InGear	BOOL	1 = Gear speed reached Set the first time the gear speed is reached. If Execute is already reset at this point, InGear will still be active in one block cycle. The bit is reset at the negative edge at Execute, when errors have occurred or when the run is overridden.
	Coupled	BOOL	1 = Axis is currently coupled to the master axis Reports in each processing cycle whether the gear speed was reached.
	CommandAborted	BOOL	1 = The job was cancelled by another.
	Error	BOOL	1 = Job was aborted due to an axis error. The bit remains active until "Execute" is reset, but at least for one block call.

Function principle

When a positive edge at the "Execute" input is detected, FB MC_GearIn couples a slave axis ("Slave" parameter) via the set gear ratio to the speed and position setpoint of a master axis ("Master" parameter). The block therefore represents a motion block of the slave axis.

The values in each cycle of a coupled run are:

- The distance traveled by the slave axis, starting at the coupling position =
  the distance traveled by the master axis, starting at the coupling position * Gear ratio
- Velocity of the slave axis = Gear speed =
  Velocity of the master axis * Gear ratio
- Acceleration of the slave axis =
  Acceleration of the master axis * Gear ratio
- Deceleration of the slave axis =
  Deceleration of the master axis * Gear ratio

To allow the slave axis to follow the master axis without restriction, set the following values (with reserve factor 1.1) at the slave axis. These values must be valid and attainable:

- Velocity(slave axis) =
  Velocity(master axis) * ABS(gear ratio) * 1.1
- Acceleration(slave axis) =
  Acceleration(master axis) * ABS(gear ratio) * 1.1
- Deceleration(slave axis) =
  Deceleration(master axis) * ABS(gear ratio) * 1.1

To cancel the coupling, call the stop block or any other motion block for the slave axis.

Same as all other motion blocks, the FB MC_GearIn can override a run and can also be overridden.

The slave axis can be coupled to a moving master axis by automatically accelerating or decelerating it using the ramps configured at the gear block until it is adapted to the gear speed.

Output "InGear" is set when the gear has reached its set speed in its initial run. If "Execute" is not set at this point of time, "InGear" is executed for the duration of one block call. "InGear" is reset with the negative edge at Execute, if an error occurs and when the run is overridden.

As long as the gear speed can be reached, the slave axis runs in speed and position synchronism. The block output "Coupled" is set.

If the slave axis cannot reach the gear speed, it is driven using the motion parameters set at the gear block. The block output "Coupled" is reset in this case. After the gear speed can be reached again, the slave axis follows this speed and the block output "Coupled" will be set.

As long as FB MC_GearIn is active at the slave axis, you can call any motion block for the master axis, and also override the motion. Exception: During a reference point run of the master axis the coupling is cancelled with error ("Err.MasterErr").

The use of the axis parameter "Override" for the slave axes is not advisable, and is thus not evaluated.

Special features when operating a slave axis as linear axis:

- If the maximum distance of travel is exceeded during a gear run, FB MC_GearIn brings the slave axis to a standstill with the error message "Err.DistanceErr" (see motion sequences). The coupling will be released.
- A linear axis which is **not synchronized** is not monitored and can thus travel to its physical limits without braking down. Hence, terminate the run in due time.
- A **synchronized** linear axis is decelerated until it comes to a standstill on the software limit switch at the end of its operational range. The error messages "DistanceErr" and "Err.SWLimitMinExceeded" or "Err.SWLimitMaxExceeded" are output and the coupling will be released.

Warning

Risk of injury and material damage.

FB MC_GearIn may drive a linear axis which is not synchronized into its physical limits.

To prevent the risk of injury and material damage, provide the following measures:

- Install an EMERGENCY-OFF switch close to the computer. This is the only means by which you can ensure that the plant is safely switched off in the event of a computer or software crash.
- Install safety limit switches for direct control of the power units of the drives.
- Ensure that no persons can access plant areas in which moving parts are present.

Special feature when using a slave axis as rotary axis:

- FB MC_GearIn does not limit the travel distance (infinite rotation).

Error handling

Master axis errors

- The master axis is brought to a stop when it detects an error which results in an soft stop. The coupled slave axis follows this motion and does not release the coupling. The master and slave axes resume the motion after the error has been acknowledged.
- When the master axis detects an error which results in a hard stop, or if it is operated in manual mode, the gear block brings the slave axis to a halt, disengages the coupling and outputs the error message "Err.MasterErr".

Slave axis errors:

- After the slave axis detects an error which leads to a soft stop, the gear block brings this axis to a halt and disengages the coupling.
- The slave axis is brought to a stop and the coupling is disengaged when this axis detects an error which leads to a hard stop.
- The maximum travel distance of the coupling run of a linear axis is limited to 2^{24} increments or 2^{24} [length unit]. When this limit is exceeded, FB MC_GearIn brakes the slave axis down to a halt, outputs the acknowledgeable message "DistanceErr" and disengages the coupling.

Coupling errors:

- The slave axis is driven using the motion parameters set at the block if it cannot reach the gear speed in the current cycle. This applies particularly when the axis is coupled to a moving master axis and during a braking operation triggered by an error.

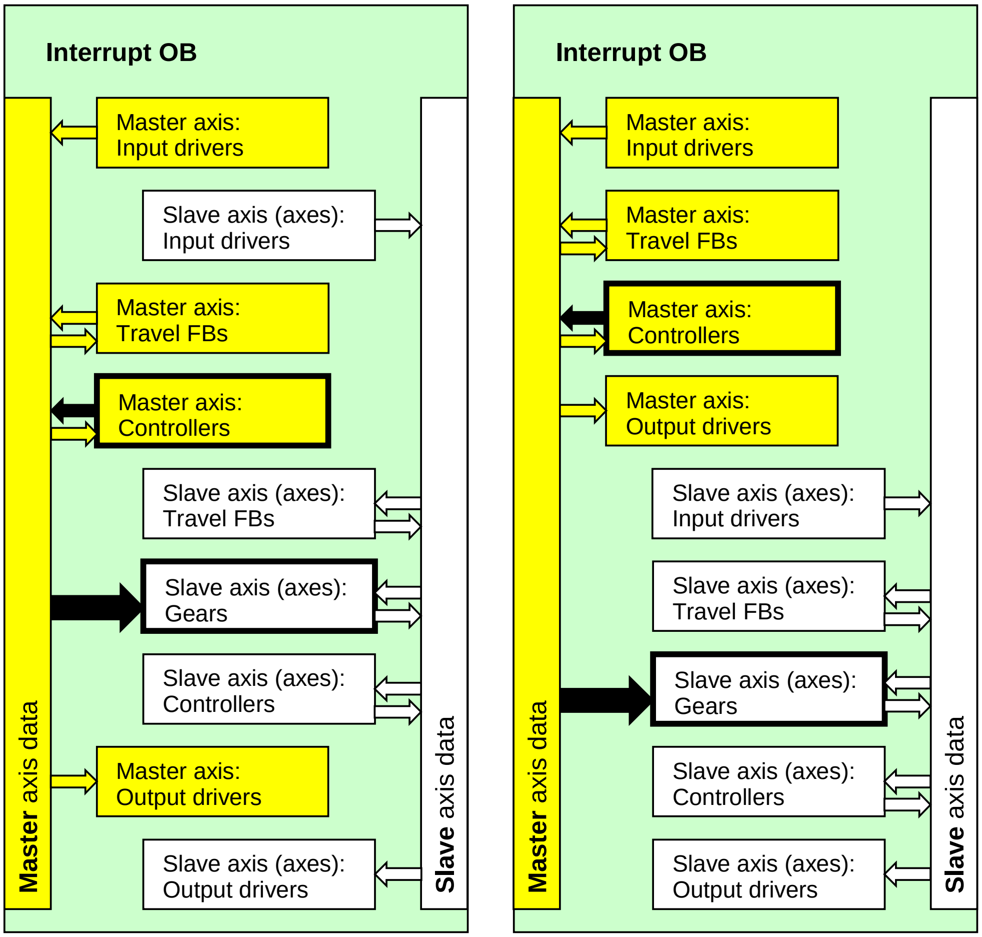
  The "Coupled" parameter is reset, but the coupling is not disengaged. As soon as the slave axis is capable of reaching the necessary values again, it is driven using these values and "Coupled" is set again.

Interaction of the blocks at the gear

In addition to the recommendations regarding the program structure in chapter 4, you should also note the following when using the gear block:

- The master and slave axes must be called in **the same** run level.
- In each processing cycle, the slave axis adapts itself again to the default values of the master axis. The slave axis blocks should therefore be called after the master axis blocks. Particularly the gear block of the slave axis should always be called **after** the controller for the master axis.
  Other call sequences, in particular the nested call of master and slave axis blocks, may cause great following distances between both axes.
- IN isochrone applications the I/O data of the master **and** of the slave axis must be placed into the same process image partition.

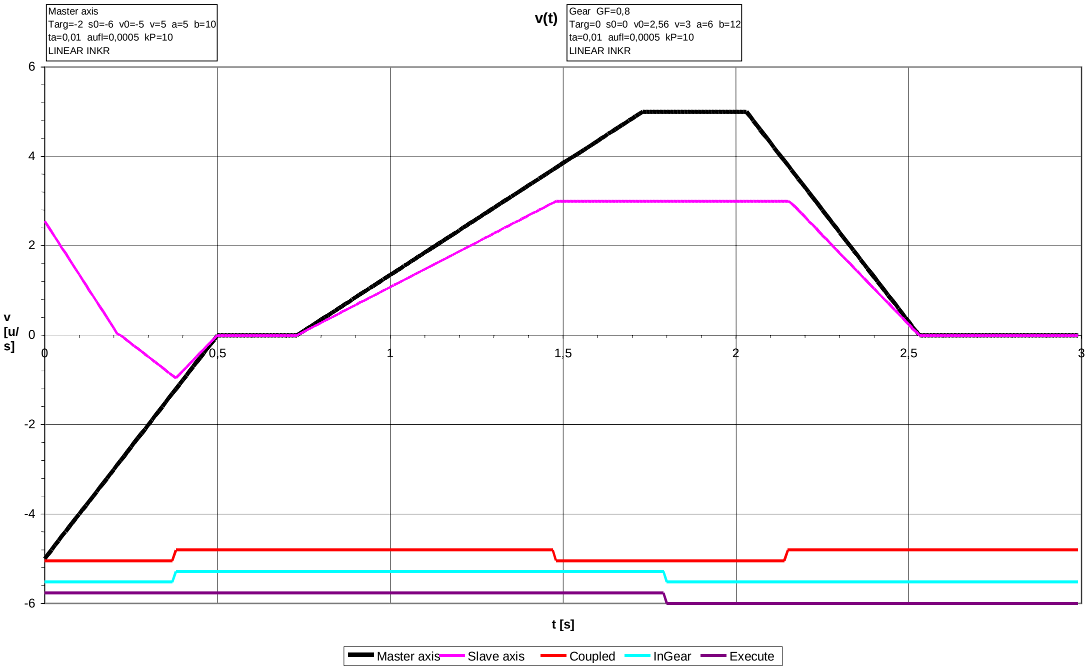
Hence, the result is one of the following pictures, depending on the selected structure:


Interrupt OB
Master axis: Input drivers
Slave axis (axes): Input drivers
Master axis: Travel FBs
Master axis: Controllers
Slave axis (axes): Travel FBs
Slave axis (axes): Gears
Slave axis (axes): Controllers
Master axis: Output drivers
Slave axis (axes): Output drivers
Master axis data
Slave axis data
Interrupt OB
Master axis: Input drivers
Master axis: Travel FBs
Master axis: Controllers
Master axis: Output drivers
Slave axis (axes): Input drivers
Slave axis (axes): Travel FBs
Slave axis (axes): Gears
Slave axis (axes): Controllers
Slave axis (axes): Output drivers
Master axis data
Slave axis data

Signal flow chart

The chart below shows how the gear block takes over control of the axis motion and how it couples to a moving master axis. In this example the gear ratio is 0.8. The slave axis cannot reach the sped required in order to maintain this ratio.

Between t = 0.5 and t = 0.72 the axis is at a standstill. The coupling is maintained.



4.17 User Program as Multi-instance FB

If you want to file all parameters of your travel FBs in one DB or use the same program for several axes, generate your user program in a multi-instance FB.

Put the axis data also in this multi-instance DB. To do this, in your multi-instance FB, define a static variable of type UDT AXIS_REF from the library "EMC Easy Motion Control".
Interconnect the Axis and Init parameters of the FBs of Easy Motion Control with these variables.

Conditionally launch FC MC_Init in your multi-instance block as the first block. The condition must be set in the startup OB (e.g. OB100) and reset after launching FC MC_Init.

You can only create new multi-instance DBs with the SIMATIC Manager or the LAD/STL/FBD Editor.

5 Axis Data-Block Structure

5.1 Function

The Axis DB is the central organization data block of Easy Motion Control. It contains

- Parameters which basically describe the axis (e.g. I/O addresses)
- Position control parameters which are calculated from these basic parameters on initialization of the EMC blocks
- Current position control values (e.g. position, error status …)
- Initialization bits for coordinating block startup

Interaction of these parameter types is ensured if you

- Process and transfer the axis DB with the configuration software or
- Ensure that, after changing certain parameters, FC MC_Init is launched.

In both cases, the plausibility of the entered parameters is checked wherever possible and the values internally required by Easy Motion Control are computed.

The configuration software also analyzes the type of changes and controls the loading processes over these.

5.2 Saving and Loading the Axis Data Block

⚠ **Warning**

Overwriting the complete axis DB is only permitted when the axis is at a standstill.

If you load the axis DB into the module without observing the advice in the following sections, correct movement of the axis is not guaranteed.

Offline

After changing specific parameters, all EMC blocks must run an initialization.

In the following description of the axis DB, these parameters are identified by **I**.

It is only possible to change and load these parameters

- when the axis is at a standstill and
- it must include setting the initialization bits (see Initialization and Parameter Changes).

The configuration software will assist you in this:

Load into the CPU with			
the configuration software of Easy Motion Control	Before loading, the configuration software compares the parameters in the open axis DB with the parameters in the axis DB of the CPU and differentiates between the following cases:		
	I **changed**	**axis** **is at a standstill**	
	NO	---	Parameters are loaded.
	YES	YES	The entire axis DB is loaded with set initialization bits.
		NO	Loading is **NOT** possible and is therefore prevented.
the SIMATIC Manager	You can only ever load the entire axis DB		
	CPU status		
	STOP		On condition that FC MC_Init is launched on starting up the CPU: no restrictions when loading.
	RUN		You may **NOT** load the axis DB!

Online

If you have changed parameters online and want to transfer these to your offline database organization (e.g. reference point coordinates in the case of absolute encoders), you should observe the following handling information:

Loading into the programming device with	
the configuration software of Easy Motion Control	no restrictions.
with the SIMATIC Manager	The initialization functions of Easy Motion Control blocks reset the initialization bits that are interconnected with them. These remain reset when you subsequently load the axis DB into the programming device. If this DB is later loaded into the CPU in RUN status, the blocks do not perform any initialization. Correct positioning of the axis is no longer guaranteed.

5.3 Axis Parameters

All information on length, velocity or deceleration relates to an arbitrary but uniform unit of length. All time information is in seconds.

In the following tables, the parameter name from the axis DB is indicated in the "Name" column and, in the "Comments" column, the parameter name used in the configuration software.

Warning

Parameters not listed in this chapter are reserved for internal use by the Easy Motion Control and should not be altered.

5.3.1 Scan Time of Position Control

<table>
<tr><th>Add.</th><th>Name</th><th>Type</th><th>Initial value</th><th colspan="2">Comment</th></tr>
<tr><td>0.0</td><td>Sample_T</td><td>REAL</td><td>0.01</td><td><b>Scan time</b> of axis FBs.
Suggestion:
at S7 300 to CPU 316 >= 0.020 [s]
at S7 400 and CPU 318 >= 0.004 [s]</td><td>I</td></tr>
<tr><td colspan="6">To ensure correct functioning of position control, it is important that all functions of one axis are processed in a fixed <b>time frame</b>. This time frame is realized, for example, by launching all blocks of one axis in the samecyclic or synchronous cycle interrupt OB. Select the time frame of the time interrupt in HW Config of the CPU.
As "scan time", enter the time frame of the OB in which you launch all blocks of one axis.

Example:
You launch the program with the Easy Motion Control blocks in OB35. In HW Config, you have set the cyclic interrupt time frame of OB35 at 10 ms.
Enter 0.010 as "scan time".</td></tr>
</table>

5.3.2 Axis Data

Add.	Name	Type	Initial value	Comment	
4.0	AxisType	BOOL	FALSE	**Axis type** FALSE = Linear axis TRUE = Rotary axis	I

Add.	Name	Type	Initial value	Comment	
4.1	EncoderType	BOOL	FALSE	**Encoder type** FALSE = Incremental encoder TRUE = Absolute encoder	I

Add.	Name	Type	Initial value	Comment
4.2	Sim	BOOL	FALSE	**Simulation mode** TRUE = switched on

Add.	Name	Type	Initial value	Comment	
6.0	AxisLimitMax	REAL	$1.0e^{+6}$	With linear axis: **Software limit switch end** With rotary axis: **Rotary axis end** Range: -2^{24} <= AxisLimitMin < AxisLimitMax <= $+2^{24}$ [unit of length]	I
10.0	AxisLimitMin	REAL	$-1.0e^{+6}$	With linear axis: **Software limit switch start** With rotary axis: **Rotary axis start** Range: -2^{24} <= AxisLimitMin < AxisLimitMax <= $+2^{24}$ [unit of length]	I

Add.	Name	Type	Initial value	Comment

<u>With linear axis:</u> Software limit switch start/end

Software limit switches are installed only for linear axes.

Monitoring of the software limit switches starts after they are enabled and when the axis is synchronized (see "Software Limit Switch Monitoring ", axis DB address 42.1). The work range is limited by the software limit switches.

The "Software limit switch start" address must always be lower than the "Software limit switch end" address.

Incremental encoder

The axis is not synchronized during the CPU startup. This is achieved by means of a subsequent reference point run, or by setting the reference point, after which software limit switch monitoring is enabled.

Absolute encoder

The set reference point will be retained during the next CPU startup. Software limit switches can be monitored.

The absolute encoder range must at least cover the work range.

With rotary axis: rotary axis start/rotary axis end

At the "rotary axis start" and "rotary axis end" parameters you determine in which value range the axis position is displayed during its rotation.

In theory the value "rotary axis end" represents the highest possible process variable the axis can reach.

$$\text{highest theoretical value} = \text{rotary axis end} - \frac{\text{Distance per encoder revolution}}{\text{Steps per encoder revolution}}$$

However, the highest theoretical value is never displayed, since it physically it marks the same position as the rotary axis start.

Example:

AxisLimitMin	= 0.0°
AxisLimitMax	= 360.0°

In a positive direction, the process variable counts ...359.8, 359.9, 0.0, 0.1,...

When using absolute value encoders, please note the "Axis Travel per Encoder Revolution ", axis DB address 68.0.

Add.	Name	Type	Initial value	Comment	
14.0	MaxVelocity	REAL	0.1	**Maximum axis velocity** Range: > 0.0 [unit of length/s]	I

Here you configure the maximum axis velocity if the motor is controlled with the value entered in the "Reference value at maximum axis velocity" parameter (Axis DB address 88.0).

The "Maximum axis velocity" parameter also forms the factor for calculating the value of the signal output to the drive as well as for limiting the axis velocity.

Special features of rotary axes with absolute encoders:

To be able to record the rotary axis position, the maximum axis velocity must be limited so that the traveled distance covered in each scan cycle is less than 50% of the range of the absolute value encoder.

Example of a single-turn encoder:

Prerequisite => One rotation of the axis is proportional to one encoder rotation

Travel distance/encoder range	= 360.0°
Travel distance/half the encoder range	= 180.0°
Scan cycle time	= 0.01 s

$$=> \text{"Maximum velocity"} < \frac{\text{Travel distance /half the encoder range}}{\text{Scan time}}$$

=> "Maximum velocity" < 180.0° / 0.01s = 18000°/s is proportional to 50 axis revolutions/s

Add.	Name	Type	Initial value	Comment	
18.0	MaxAcceleration	REAL	0.1	**Maximum axis acceleration** Range: > 0.0 [unit of length/s^2]	I

The slope of the acceleration ramp of the axis is restricted to this value in controlled mode.

The maximum possible acceleration can be determined empirically.

It should only be high enough so that, when accelerating to maximum velocity (under load), it is just below the current limit of the drive.

Add.	Name	Type	Initial value	Comment	
22.0	MaxDeceleration	REAL	0.1	**Maximum axis deceleration** Range: > 0.0 [unit of length/s^2]	I

The slope of the deceleration ramp of the axis is restricted to this value in controlled mode.

The maximum possible deceleration can be determined empirically.

It should only be high enough so that, when slowing down from maximum velocity (with load), it is just below the current limit of the drive.

5.3.3 Data for Monitoring

Add.	Name	Type	Initial value	Comment	
26.0	MonTimeTargetAppr	REAL	1.0	**Monitoring time for target approach** Range: > 2 x Sample_T (axis DB address 0.0, Scan time) [s]	I
30.0	TargetRange	REAL	1.0	**Target range** Range: > 0.0 [unit of length]	I

When the **position setpoint** has reached the target range after an approach, the "monitoring the time for target approach" starts if the "target approach monitoring" bit is set (axis DB address 42.0). If the actual **position value** is less than the value defined at the "target range" parameter (axis DB address 30.0) after this time has expired, the "target approach error" bit is set in the axis DB when the axis is reached (axis DB address 130.4).

The "target range" lies to the left and right of the target (see Concept Definitions).

Add.	Name	Type	Initial value	Comment	
34.0	StandstillRange	REAL	1.5	**Standstill range** Range: > 0.0 [unit of length]	I

When the axis has reached a standstill, the system monitors whether the axis retains or drifts off its current target position.

The "standstill range" lies to the left and right of the target (see Concept Definitions).

We recommend a "standstill range" that is greater than the "target range" (axis DB address 30.0).

Add.	Name	Type	Initial value	Comment	
38.0	MaxFollowingDist	REAL	5.0	**Maximum allowable following distance** Range: > 0.0 [unit of length]	I

The following distance monitoring function is used to check whether the actual position during a motion remains inadmissibly far behind the position setpoint..

The "following distance exceeded" error bit (axis DB address 130.2) is set if the maximum following distance is exceeded.

Recommended setting:

("Maximum velocity" / "Controller gain") x 1.1 (see axis DB address 14.0 and 44.0).

Add.	Name	Type	Initial value	Comment
42.0	MonitorTargetAppr	BOOL	TRUE	**Monitor target approach** TRUE = monitoring switched on
When target range monitoring is enabled, the motion is terminated when the actual position value reaches the "Target range" (axis DB address 30.0) within the "monitoring time for target approach" (axis DB address 26.0). When target range monitoring is disabled, the motion is terminated when the actual position value reaches the target (see Concept Definitions)				

Add.	Name	Type	Initial value	Comment
42.1	SWLimitEnable	BOOL	TRUE	**Monitor software limit switch** TRUE = monitoring switched on
There are only software limit switches with linear axes. When monitoring of the software limit switch is disengaged, the parameters "software limit switch start" and "software limit switch end" (axis DB addr. 6.0 and 10.0) are of no significance.				

5.3.4 Position Controller Data

Add.	Name	Type	Initial value	Comment
44.0	FactorP	REAL	1.0	**Controller gain** Range: > 0.0 [1/s]
The controller gain for the axis can be tuned by way of experiment. Increase controller gain in steps of 1.0 up to the point where the axis develops oscillation in its standstill position or during travel. Then reduce the controller gain until this oscillation is eliminated.				

Add.	Name	Type	Initial value	Comment
48.0	ManVelocity	REAL	0.0	**Manual setpoint velocity** Range: <= ± MaxVelocity [unit of length/s]
52.0	ManEnable	BOOL	FALSE	**Enable manual mode** TRUE = switched on
With manual control engaged, the controller outputs the value "Manual setpoint velocity " as a manipulated variable and limits it to "MaxVelocity". The position setpoint is corrected to the actual position.				

Add.	Name	Type	Initial value	Comment	
54.0	EmergencyDec	REAL	1.0	**Deceleration for hard stop** Range: > 0.0 [unit of length/s^2]	I
At this parameter you define the slope of the time-controlled emergency/hard stop deceleration ramp for the drive.					

5.3.5 Encoder Data

Add.	Name	Type	Initial value	Comment	
58.0	InputModuleInAddr	INT	0	**Start address for the position detection module inputs** Range: determined by HW Config	I
Enter the value assigned for this module in HW Config.					

Add.	Name	Type	Initial value	Comment	
60.0	InputModuleOutAddr	INT	0	**Start address for the position detection module outputs** Range: determined by HW Config	I
The parameter is only used with modules where the "start address for the position detection module outputs" can, in the HW Config, be set differently than the "start address or the position detection module inputs". Enter the value assigned for this module in HW Config.					

Add.	Name	Type	Initial value	Comment	
62.0	InputChannelNo	INT	0	**Channel number** Range: dependent on module	I
Enter "0" for the first channel and with single-channel modules.					

Add.	Name	Type	Initial value	Comment	
64.0	StepsPerRev	DINT	L#1	**Increments per encoder revolution** Range: >= 1 [steps/encoder revolution]	I

Enter the number of steps which your position detection module outputs per encoder revolution.

With this value and the parameter "axis distance per encoder revolution" (axis DB addr. 68.0), steps are converted according to unit of length.

Incremental encoder

If the incremental encoder is read in with a module which performs a twofold or fourfold evaluation of the impulses, this has to be considered when inputting the parameters.

Example:

Encoder pulses per revolution:	500
Fourfold evaluation	
You enter in "steps per encoder revolution":	2000

Absolute encoder

With absolute encoders the number of steps per encoder revolution is usually a power of two.

Common values are:

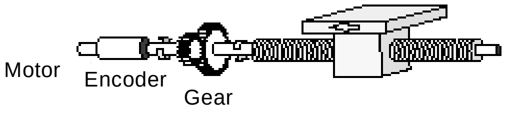
- with a single-turn encoder: 4096 and 8192
- with a multi-turn encoder (12x12 bits in the 25 bit telegram) 4096

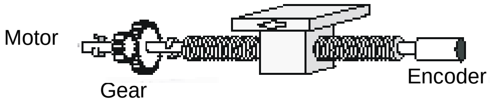
If there is any doubt, the correct value can be determined by moving the encoder one revolution and observing the change in encoder value in the configuration software.

With a **linear scale**, for "steps per encoder revolution" enter the entire number of steps corresponding to the length of your linear scale.

Add.	Name	Type	Initial value	Comment	
68.0	DisplacementPerRev	REAL	1.0	**Axis distance per encoder revolution** Range: > 0.0 [unit of length/encoder revolution]	I

Here you determine the axis travel distance per encoder revolution.
This value depends on the structure of the axis and on the encoder mounting position.
You must take all transmission units into consideration, e.g. couplings or gears.





Special feature of rotary axes
The encoder must perform an integer number of steps at each axis revolution.

$$\frac{(\text{Rotary axis end} - \text{Rotary axis start})}{\text{Axis distance per encoder revolution}} \text{ x steps per encoder revolution} = \text{integer}$$

Special feature of rotary axes with absolute value encoders
To ensure the reproducibility of all positions, the axis travel distance covered by the encoder must be a multiple integer of the distance the rotary axis covers in one rotation:

$$\frac{(\text{number of encoder revolutions}) \text{ x } (\text{axis distance per encoder revolution})}{(\text{Rotary axis end} - \text{Rotary axis start})} = \text{integer}$$

Example:
24-bit absolute value encoder

=> Number of encoder revs	= 4096
Rotary axis start	= 0.0°
Rotary axis end	= 360.0°
=> Rotary axis end - Rotary axis start	= 360.0°
Axis travel per encoder rev	= 45°

$$\text{inserted}: \frac{4096 \times 45^{\circ}}{360^{\circ}} = 512\,(\text{integer})$$

For a **linear scale**, enter the length of your linear scale for "Axis travel per encoder revolution".

<table>
<tr><th>Add.</th><th>Name</th><th>Type</th><th>Initial value</th><th colspan="2">Comment</th></tr>
<tr><td>72.0</td><td>NumberRevs</td><td>INT</td><td>1</td><td><b>Number of encoder revolutions</b> for an absolute encoder
Range:
&gt; 0</td><td>I</td></tr>
<tr><td colspan="6">Enter the number of encoder revolutions with an absolute encoder.
With a single-turn encoder, enter "1" here.
With an incremental encoder, this parameter is of no significance.
With a <b>linear scale</b>, enter "1" for "number of encoder revolutions".</td></tr>
</table>

<table>
<tr><th>Add.</th><th>Name</th><th>Type</th><th>Initial value</th><th colspan="2">Comment</th></tr>
<tr><td>74.0</td><td>PolarityEncoder</td><td>INT</td><td>1</td><td><b>Set encoder polarity</b>
Range:
+1 or -1</td><td>I</td></tr>
<tr><td colspan="6">Adjust the distance measurement direction to the movement direction of the axis.
+1 = ascending counting pulses (incremental encoder) or encoder values (absolute encoder) correspond to ascending actual positions.
-1 = ascending counting pulses (incremental encoder) or encoder values (absolute encoder) correspond to descending actual positions.</td></tr>
</table>

5.3.6 Data for Setpoint Output

Add.	Name	Type	Initial value	Comment	
80.0	OutputModuleOutAddr	INT	0	**Initial address for the output module outputs** Range: determined by HW Config	I
Enter the value assigned for this module in HW Config.					

Add.	Name	Type	Initial value	Comment	
82.0	OutputModuleInAddr	INT	0	**Initial address of the output module inputs** Range: determined by HW Config	I
The parameter is only used with modules where the "initial address of the output module inputs" is adjustable in HW Config separately to the "initial address of the output module outputs". Enter the value assigned for this module in HW Config.					

Add.	Name	Type	Initial value	Comment	
84.0	OutputChannelNo	INT	0	**Output module channel number** Range: dependent on module	I
Enter "0" for the first channel and with single-channel modules.					

Add.	Name	Type	Initial value	Comment	
86.0	PolarityDrive	INT	1	**Set drive polarity** Range: +1 or -1	I
Select the direction of drive movement. +1 = positive velocity setpoint values cause a movement in the positive direction. -1 = positive velocity setpoint values cause a movement in the negative direction. If you change "Set drive polarity", you must also change "Set encoder polarity" (axis DB addr. 74.0).					

Add.	Name	Type	Initial value	Comment	
88.0	DriveInputAtMaxVel	REAL	9.0	**Reference value at maximum axis velocity** Range: Depends on the power unit	I

If using analog-controlled power units, in this parameter you must enter the value of the voltage at which "maximum velocity" of the drive (axis DB addr. 14.0) is reached.

Example:

Nominal speed of motor:	= 3000 rpm
Motor is aligned to:	= 9V at 3000 rpm
You wish to drive at maximum:	= 1500 rpm
For "Reference value for maximum axis velocity", enter:	= 4.5V

When using the output driver FB OutputMM4_DP, use this parameter to declare an appropriate control value that is required to accelerate the drive to "maximum velocity" (axis DB address 14.0).

Example:

Rated rpm of the motor:	= 1380 rpm
The motor is calibrated to:	= 1380 rpm at 50Hz
The maximum travel required:	= 690 rpm
For "Reference value for maximum axis velocity", enter:	= 25Hz

Add.	Name	Type	Initial value	Comment
92.0	OffsetCompensation	REAL	0.0	**Offset compensation** Range: -10.0 to +10.0 [V]

This parameter allows you to perform an offset compensation if your drive does not offer this facility. The value configured here is added to the output value.
The output driver FB OutputMM4_DP ignores this parameter.

Add.	Name	Type	Start value	Comment	
96.0	DriveInputAt100	REAL	10.0	**Reference value for 100 % speed** Range: -10.0 to +10.0 [V]	I

This parameter is irrelevant when you are operating power units with a control voltage of ±10V.

If you are using the output driver FB OutputMM4_DP, use this parameter to declare an appropriate control value that is required to accelerate the drive to 100% of its speed.

Example:

Rated rpm of the motor (100% speed):	= 1380 rpm
The motor is calibrated to:	= 1380 rpm at 50Hz
For "Reference value for 100% speed", enter:	= 50Hz

5.4 Control and Current Value

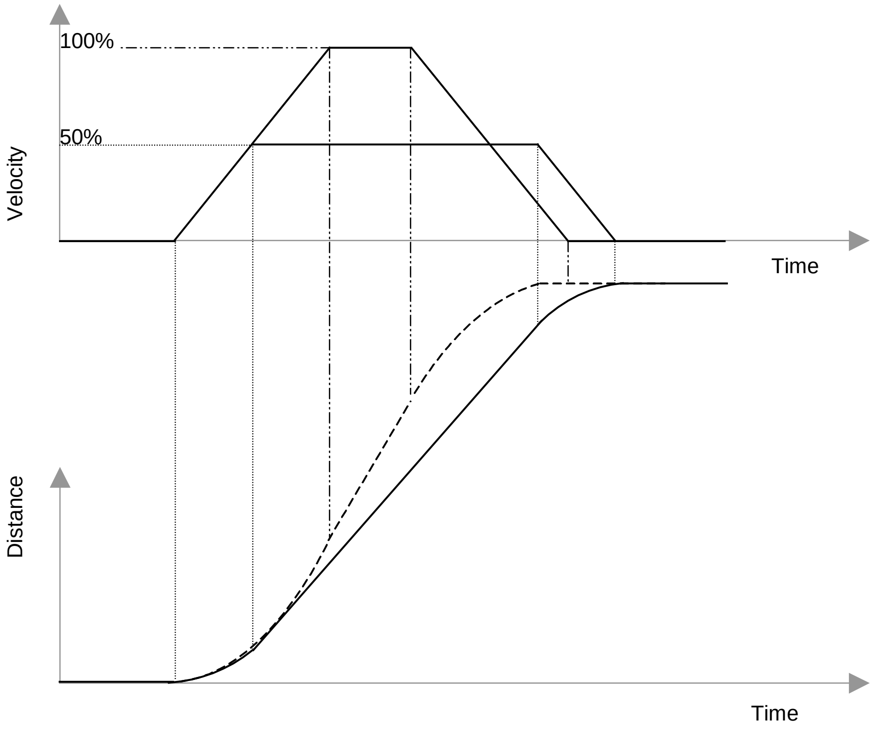
5.4.1 Velocity Override Control Value

Add.	Name	Type	Initial value	Comment
100.0	Override	REAL	100.0	**Velocity override** Range: 0.0 to +100.0 [%]

"Velocity override" has a continuous effect on the configured velocity of a motion FB. Acceleration and deceleration values are not affected.

This parameter has no effect on a slave axis coupled by FB MC_GearIn.

Example: Travel curves with override of 50% and 100%.



When switching from 50% to 100%, velocity is doubled. As acceleration and deceleration are not affected by parameter override, however, the positioning time is **not** halved.

5.4.2 Current Values

The following parameters show the current information on the status of your axis during travel. You should not change the contents of these parameters.

Add.	Name	Type	Initial value	Comment
104.0	ActPosition	REAL	0.0	**Actual position** Range: Floating point number [unit of length]
Here the actual encoder value is displayed.				

Add.	Name	Type	Initial value	Comment
108.0	FollowingDistance	INT	0	**Following distance** Range: Floating point number [unit of length]
Following distance = current position setpoint - actual position				

Add.	Name	Type	Initial value	Comment
112.0	RemainingDistance	REAL	0.0	**Remaining distance** to target Range: Floating point number [unit of length]
During travel with target specification (FB MC_MoveAbsolute, FB MC_MoveRelative), the residual distance to the target position is displayed. Travel without target (FB MC_MoveJog, FB MC_Home) as well as a hard error sets the residual distance at 0.				

Add.	Name	Type	Initial value	Comment
116.0	NomVelocity	REAL	0.0	**Setpoint velocity** Range: Floating point number [unit of length/s]
The setpoint velocity of the axis calculated by the travel FB is displayed here during travel.				

Add.	Name	Type	Initial value	Comment
120.0	ActVelocity	REAL	0.0	**Actual velocity** Range: Floating point number [unit of length/s]
The actual velocity of the axis is displayed here.				

Add.	**Name**	**Type**	**Initial value**	**Comment**
124.0	Sync	BOOL	FALSE	**Axis synchronized** FALSE = not synchronized TRUE = synchronized
	Incremental encoder			**Absolute encoder**
FALSE	After CPU startup or detecting an encoder error			Before initial reference setting
TRUE	After a reference search or reference setting			After reference setting

Add.	**Name**	**Type**	**Initial value**	**Comment**
136.0	EncoderValue	DINT	L#0	**Current encoder value** Range: Decimal [steps]
Here the current encoder value is displayed.				

5.5 Error Displays and Error Acknowledgement

For detailed information on error response, error display and error acknowledgement, see Error Displays and Error Acknowledgement in axis DB.

<table>
<tr><th>Add.</th><th>Name</th><th>Type</th><th>Initial value</th><th>Comment</th></tr>
<tr><td>124.1</td><td>Error</td><td>BOOL</td><td>FALSE</td><td><b>Group error</b>
FALSE = no error
TRUE = group error</td></tr>
<tr><td colspan="5">The group error is TRUE as long as there is at least one error.</td></tr>
</table>

<table>
<tr><th>Add.</th><th>Name</th><th>Type</th><th>Initial value</th><th>Comment</th></tr>
<tr><td>124.2</td><td>ErrorAck</td><td>BOOL</td><td>FALSE</td><td><b>Group acknowledgement</b> for all errors
FALSE = no acknowledgement
TRUE = acknowledge error</td></tr>
<tr><td colspan="5">When group acknowledgement is set, all acknowledgeable error displays are deleted.
Group acknowledgement is only effective when the axis is at a standstill.
Then group acknowledgement is automatically reset. You should not reset group acknowledgement yourself.
As long as input drivers EncoderIM178 and EncoderET200S1Count detect an encoder error, a set group acknowledgement will remain until the encoder error goes.</td></tr>
</table>

Add.	Name	Type	Initial value	Comment
128.0	Err	STRUCT	FALSE	Error with controlled deceleration slope, see Error with Soft Stop.
130.0				Error with time-controlled deceleration slope, see Error with Hard Stop.
132.0	Config	STRUCT	FALSE	Configuration error, see Configuration Error.

5.6 Bit Field for the Initialization Function

Add.	Name	Type	Initial value	Comment
278.0	Init	STRUCT	TRUE	Bit field for initializing function blocks

All Easy Motion Control function blocks must be initialized after every CPU startup and after changing specific parameters.

For this purpose, the bit field "Init.Ix" is defined in the axis DB and each block has the parameter "Init". The blocks run through their initialization routine if their input "Init" is set, and then reset the interconnected initialization bit.

With every FB call, you must interconnect the parameter "Init" with a bit from this bit field. This bit should not be used by any other FB (see Initialization and Parameter Changes).

Whenever FC MC_Init is called or you change a parameter in the configuration software which is only active after initialization of the blocks, all bits in this field are simultaneously set TRUE. This allows all function blocks to carry out an initialization with their next call.

6 Configuration

All blocks of one axis work together with one axis DB that contains all the axis data necessary for operating the axis. This axis DB can be generated and parameterized using Easy Motion Control configuration software.

You open the configuration software via the Windows start menu (**Start > Simatic > STEP7 > Easy Motion Control V2**).

Integrated Help

The configuration software has an integrated help which supports you during Configuration. You have the following possibilities of launching the integrated help:

- Using menu command Help > Contents… or
- Pressing F1

7 Commissioning

7.1 General

Important

Please note the points listed in the following warning.

Warning

Injury may occur to persons and property.

Uncontrolled travel motions during commissioning and operation can cause severe injury to persons and property.

To avoid injury to persons and objects, take the following precautions:

- Install an EMERGENCY STOP switch in the vicinity of the computer. This is the only way to ensure that the system is reliably switched off in the event of a hardware or software failure.
- Install safety limit switches with direct control over the power units of all drives.
- Ensure that nobody has access to the area of the system where there are moving parts.

Axis commissioning is described as a model in the primer *Easy Motion Control - Getting Started* for the CPU 314C with integral I/O (Call: **Start > Simatic > Documentation > English > Easy Motion Control V2 - First Steps**).

7.2 Error Pictures and Remedies during the Commissioning Phase

Following are various error pictures, possible causes and appropriate remedies which may be of assistance in commissioning an axis with Easy Motion Control blocks.

7.2.1 Axis does not Move Despite Travel Order

- Is drive wiring okay?
    - Is the "DriveEnabled" output of FB MC_Control continuously interconnected with the enable input of the power unit?
    - Is the "DriveEnabled" output of FB MC_Control interconnected with the "EnableDrive" input of the output driver?
- Are the I/O addresses and channel numbers correct?
  In the axis DB, the **initial** addresses of the modules are required. From the initial address and channel number (counting method: from channel 0), the driver blocks calculate the module-specific I/O addresses for setpoint and actual values.
- Is simulation mode engaged?
  If yes ("Sim" = TRUE), **no** I/O values are read or written.
- Is manual mode engaged?
  If yes ("ManEnable" = TRUE), a "setpoint velocity for manual mode" must also be specified ("ManVelocity" <> 0.0 [unit of length/s]).
- Is drive enabled?
  The input "EnableDrive" must be set at TRUE at FB MC_Control.
- Are there any axis errors present?
  The bit "group error" ("error") is then set in the axis DB and the cause of error is visible in the error display bit fields:
    - After launching FC MC_Init, the error "Stop status requiring acknowledgement" is always set ("Err.StoppedMotion") and must be acknowledged (set the "group acknowledgement" ("ErrorAck") = TRUE to do this).
    - After correction of erroneous values, Configuration errors require FC MC_Init to be rerun.
    - Other errors first have to be rectified and then acknowledged.
    - Encoder errors: Is the distance measurement module fully parameterized, if necessary with the help of the module configuration software?

- Is the "group acknowledgement" coming through?
  A "group acknowledgement" that you have set ("ErrorAck") is normally detected, processed and reset by Easy Motion Control.
  If it remains set, this may mean that
  - the Easy Motion Control blocks are staying in their initialization branch due to FC MC_Init being launched continuously or
  - distance measurement module is not even responding to the error acknowledgement
- Is "velocity override" = 0% ("Override")?
- Make sure that the desired travel order is also processed by the program ("Busy" of travel FB = TRUE).
- The error "Parameter of a travel FB inadmissible" (Err.DataErr") or "Target out of travel range " ("Err.TargetErr") is present.
  If you download the axis DB to the CPU without using the Easy Motion Control configuration software, you must call FC MC_Init that which writes the auxiliary values to the axis DB.

7.2.2 Axis Moves without Travel Order

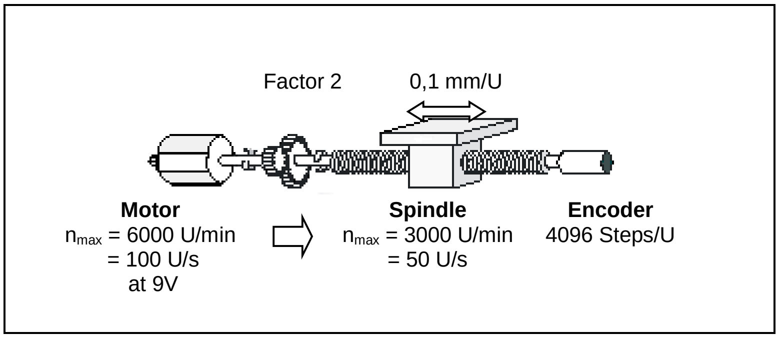
- The axis starts moving very quickly:
  - The direction of control action is probably not properly set. The parameters "Set encoder polarity" ("PolarityEncoder") and "Set drive polarity" ("PolarityDrive") are used for adjusting the blocks to your hardware installation. Adjust the parameters with the help of the "wiring test" wizard.
- The axis drifts slowly:
  - The drive power unit is enabled independently of the output "DriveEnabled" of FB MC_Control and the axis is <u>not</u> in control (i.e. FB MC_Control outputs "DriveEnabled" = FALSE).
  - The analog output is not correctly wired with the drive's power unit.
  - Either the input or the output driver does not access the module belonging to the drive, due to a faulty setting of the module address or channel number at the axis DB. Set these parameters by means of the "Wiring test" wizard.

7.2.3 Axis Velocity other than Expected or Axis Oscillates

Are the parameters describing the axis correct? There are certain connections between these parameters which should be explained by the EXAMPLE.

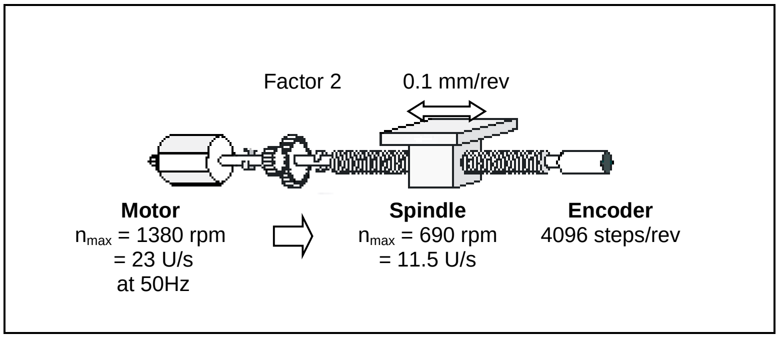
- "Controller gain" ("factorP"):
  If the selected "controller gain" (factorP) is too high, the axis may oscillate.
- "Maximum axis velocity" ("MaxVelocity")
  "Reference value for maximum axis velocity" ("DriveInputAtMaxVel"):

  EXAMPLE of a ± 10V control:



The following relationship holds:
If the output module outputs 9V, the drive turns at 6000 rpm. At the spindle, this is 3000 rpm or 50 revs/s. The slides moves at 0.1 mm per spindle revolution. This results in a maximum slide speed of 5 mm/s.
In other words, set:
"Maximum axis velocity" = 5 mm/s and "Reference value for maximum axis velocity" = 9V.

EXAMPLE for a Micromaster MM440 DP:



The following relationship holds:
The drive runs at a speed of 1380 rpm when the drive controller outputs a frequency of 50Hz. This is proportional to a spindle speed of 690 rpm or 11.5 rev/s. The slide moves by a distance of 0.1 mm per spindle revolution. The maximum velocity of the slide is thus 1.15 mm/s.

In other words, set:
"Maximum axis velocity " = 1.15 mm/s
"Reference value at maximum axis velocity" = 50Hz
"Reference value at 100% speed = 50Hz.

- "Steps per encoder revolution" (StepsPerRev"):
  You can check whether the number of steps produced by the encoder coincides with the configured value by observing the parameter "encoder value" with the configuration software, while you turn the **encoder once** by hand.

  EXAMPLE: the "encoder value" alters by 4096 steps with one spindle revolution.
- "Axis distance per encoder revolution" ("DisplacementPerRev"):
  This parameter assigns a specific distance to one revolution of the encoder. You can check whether this is correct by observing the parameter "actual position" with the configuration software, while you turn the **encoder once** by hand.

  EXAMPLE: the "actual position" alters by 0.1 mm with one spindle revolution.
- "Scan time" ("Sample_T"):
  Scan time is used together with the distance covered since the last scan in order to calculate axis velocity. It must correspond to the launch time of the OB in which the Easy Motion Control blocks are launched.
  "Sample_T" must be entered in seconds in the axis DB.

7.2.4 Positioning not Accurate Enough

- Check the details of the parameter "Steps per encoder revolution" ("StepsPerRev").
- Determine the parameter "Axis distance per encoder revolution" ("DisplacementPerRev") as accurately as possible (Wizard "Distance measurement").

7.2.5 Maximum Following Distance Exceeded

- In order for the "maximum axis velocity" ("MaxVelocity") to be reached at all, the following must apply:

$$\text{"MaxFollowingDistance"} >= \frac{\text{"MaxVelocity"}}{\text{"FactorP"}}$$

- Configured acceleration cannot be reached by the real axis
- The parameters "Maximum axis velocity" ("MaxVelocity") and "Reference voltage for maximum axis velocity" ("DriveInputAtMaxVel") or "Reference value for 100% speed" (DriveInputAt100")are not set correctly (see above).

7.2.6 The Gear Block Displays Neither "GearIn" Nor "Coupled"

- The slave axis is about to couple to a moving master axis without having (yet) reached the gear speed.
- The slave axis has disengaged the coupling due to an error.

7.2.7 The Gear Block Displays "GearIn", but not "Coupled"

- The slave axis may have reached the gear speed for (possibly only for one cycle), but has lost this speed again:
  - The acceleration or deceleration gain of the master axis exceeds the capability of the slave axis due to its configuration and the gear ratio.
  - The slave axis cannot reach the velocity that is determined by the velocity setpoint of the master axis and by the gear ratio.

8 Diagnostics

8.1 Error Displays and Error Acknowledgement in Axis DB

All errors detected by the blocks are displayed in the axis DB by

- setting the group error "Error" and
- an error-specific display "Err.xxxx".

With errors requiring acknowledgement, an error acknowledgement must be set after rectifying the error (parameter "ErrorAck" in axis DB, see below).

There are three types of errors with different error responses:

1. Errors where perfect functioning of position control is **no** longer guaranteed lead to a **hard stop**:
   FB MC_Control brings the axis to a standstill with the deceleration slope value entered in "EmergencyDec". Then FB MC_Control sets its "DriveEnabled" output at "0".
   After rectifying the error, set the error acknowledgement in axis DB. Then "DriveEnabled" is reset.
2. Errors where position control still functions lead to a **soft stop**:
   The axis is slowed down with the deceleration configured in the active travel FB. The axis remains in control, "DriveEnabled" is **not** reset.
   In this state the motion can only be overriden by means of MC_StopMotion.
   After rectifying the error, set error acknowledgement in axis DB.
3. Error in axis parameters (**Configuration error**):
   Insofar as possible, FC Init checks the parameters entered in the axis DB. In the event of an error, it sets the bit "Err._Config.Err" and enters the exact error cause in "Config.xxx".
   Configuration errors **cannot** be rectified by acknowledgement. Approach of the axis is prevented. After the incorrectly entered parameter is corrected, FC Init must be launched again. Configuration error displays are then reset once there is no longer any error.

The following applies to errors with hard and soft stop:

No travel order is possible as long as there are one or several errors present and these are not acknowledged.

Exception:

If only one software limit switch is exceeded, a travel order in the opposite direction is possible. This error is automatically acknowledged after exiting the software limit switch.

If the error was detected by a travel FB (e.g. FB MC_MoveJog or FB MC_MoveAbsolute), the "Error" output is set at this FB ("CommandAborted").

If the error was detected by a driver block, the exact cause of the error is indicated with this FB (see Input Driver (FB 21 to FB 29)).

Add.	Name	Type	Initial value	Comment
124.1	Error	BOOL	FALSE	Group error FALSE = no error TRUE = group error
The group error is TRUE as long as there is at least one error.				

Add.	Name	Type	Initial value	Comment
124.2	ErrorAck	BOOL	FALSE	**Group acknowledgement** for all errors FALSE = no acknowledgement TRUE = acknowledge error
When group acknowledgement is set, all acknowledgeable error displays are deleted. Group acknowledgement is only effective when the axis is at a standstill and after initialization. After evaluation of error acknowledgement, "ErrorAck" is automatically reset.				

8.1.1 Error with Soft Stop

The following applies to all errors described below:

TRUE = the error is present.

Add.	Name Err.	Meaning		Q = Acknowledgement necessary
128.0	SWLimitMin Exceeded	**Software limit switch start exceeded**		--
		Cause:	• The actual position is outside the working range. • An encoder exchange with an absolute encoder.	
		Remedy:	• Draw out the axis from the limit switch. • Turn off the software limit switch ("SWLimitEnable" = FALSE), synchronize the axis again and turn the software limit switch back on again.	
128.1	SWLimitMax Exceeded	**Software limit switch end exceeded**		--
		Cause:	• The actual position is outside the working range. • An encoder exchange with an absolute encoder.	
		Remedy:	• Draw out the axis from the limit switch. • Turn off the software limit switch ("SWLimitEnable" = FALSE), synchronize the axis again and turn the software limit switch back on again.	
128.2	TargetErr	**Target outside of permitted travel range**		Q
		Cause:	• FB **MC_MoveAbsolute**: • The target lies in a software limit switch • The target is < "rotary axis start" or >= "rotary axis end" • The calculated travel distance is greater than the distance corresponding to 2^{24} steps of your encoder. • Auxiliary variables were not calculated in the axis DB, because FC MC_Init was not called. • FB **MC_MoveRelative**: • The input travel distance leads to a target in a software limit switch. • The specified distance is greater than the distance corresponding to 2^{24} steps of your encoder. • Auxiliary variables were not calculated in the axis DB, because FC MC_Init was not called. • FB **MC_Home**: • The reference coordinate lies in a software limit switch • The reference coordinate is < "rotary axis start" or >= "rotary axis end" • Auxiliary variables were not calculated in the axis DB, because FC MC_Init was not called.	
		Remedy:	• Correct the parameters. • Call FC MC_Init	

Add.	**Name Err.**	**Meaning**		Q **= Acknowledgement necessary**
128.3	NoSync	**Axis not synchronized**		Q
		Cause:	FB MC_MoveAbsolute was started with an unsynchronized axis.	
		Remedy:	Synchronize the axis (reference search, reference setting).	
128.4	DirectionErr	**Invalid travel direction entered**		Q
		Cause:	One movement further into a software limit switch is inadmissible.	
		Remedy:	Start a travel order away from the software limit switch.	
128.5	DataErr	**Invalid travel-FB parameter**		Q
		Cause:	FB **MC_MoveHome**: • "Velocity" < 0 • "Velocity" > 0 and "Acceleration" <= 0 • "Velocity" > 0 and "Deceleration" <= 0 • "Velocity" > 0 with absolute value encoder • "Position" lies outside the numerical range at a linear axis. • Auxiliary variables were not calculated in the axis DB, because FC MC_Init was not called. FB **MC_MoveAbsolute**, **MC_MC_MoveRelative** and **MC_MoveJog**: • "Velocity" <= 0 • "Acceleration" <= 0 • "Deceleration" <= 0 • The input or calculated target coordinates are out of the numerical range (at MC_MoveAbsolute and MC_MoveRelative). • Gear ratio denominator (RatioDenominator") <= 0 (MC_GearIn) • Gear ratio counter ("RatioNumerator") = 0 (MC_GearIn) • A travel FB was interrupted and "deceleration" of the interrupting FB does not coincide with the deceleration configured on the interrupted block (see (see Overriding Travel). • Auxiliary variables were not calculated in the axis DB, because FC MC_Init was not called.	
		Remedy:	• Specify admissible values. • Initialize the blocks. • Call FC MC_Init.	

Add.	Name Err.	Meaning	Q = Acknowledgement necessary
128.6	StartErr	**Start from current axis status not possible**	Q
		Cause:	• The execute input at a motion FB or at FB MC_StopMotion was set at a time the axis was in error state or in manual mode ("ManEnable" = TRUE), or a FB MC_StopMotion was being executed. • FB MC_Home was not started when the axis was at a standstill. • Both directions are set at FB MC_MoveJog. • The master axis for FB MC_GearIn is in faulty state: • Error with hard stop • MC_Home is busy • Manual mode ("ManEnable" = TRUE).
		Remedy:	• Eliminate all errors. • Make sure that the master axis is in a permissible state.
128.7	DistanceErr	**Overtravel**	Q
		Cause:	A motion without target (FB MC_Home, MC_MoveJog, MC GearIn) has covered a travel distance that exceeds 2^{24} encoder steps. The axis is halted roughly at this point. A synchronized linear axis was brought to a halt at a software limit switch by means of MC_MoveJog or MC_GearIn. In most cases "SWLimitMin Exceeded" or "SWLimitMax Exceeded" is also set. With FB MC_Home, a reference search has covered a distance which corresponds to more than 2^{24} steps of the encoder.
		Remedy:	Make sure the travel distance does not exceed the limit specified above (see Maximum Travel Distance)..
129.0	MasterErr	**Master axis in faulty state**	Q
		Cause	The master axis was set to an impermissible state during gear coupling: • Error with hard stop • MC_Home is busy • Manual mode ("ManEnable" = TRUE)
		Remedy	Clear the faulty status of the master axis.

8.1.2 Error with Hard Stop

The following applies to all errors described below:

TRUE = the error is present.

Add.	Name Err.	Meaning		Q = Acknowledgement necessary
130.0	StoppedMotion	**Axis is in stop state which needs acknowledgment**		Q
		Cause:	The axis is in stop status requiring acknowledgement • after initialization by launching FC MC_Init or • as the result of an error listed in this table.	
		Remedy:	Rectify any errors that might possibly be present.	
130.1	EnableDriveErr	**Drive enable missing**		Q
		Cause:	"EnableDrive" is not set at FB MC_Control.	
		Remedy:	Set "EnableDrive".	
130.2	FollowingDistErr	**Following distance exceeded**		Q
		Cause:	The axis does not follow the setpoint values or is too slow in following.	
		Remedy:	• Switch on the drive. • Optimize "Controller Gain" (axis DB addr. 44.0). • Adjust Set Encoder Polarity (axis DB addr. 74.0) to Set Drive Polarity (axis DB addr. 86.0). • Increase the max. admissible following distance (axis DB addr. 38.0).	
130.3	StandstillErr	**Outside Standstill Range**		Q
		Cause:	• Axis has moved out of "standstill range" without travel order. • "Standstill range" is less than "target range" • "Set encoder polarity" or "Set drive polarity" is incorrectly parameterized.	
		Remedy:	• Check the connection to the setpoint input of the power unit. • Correct the parameters "standstill range"/"target range" (axis DB addr. 34.0, 30.0) • during initial startup: Run a wiring test (axis DB addr. 74.0, 86.0).	
130.4	Target ApproachErr	**Error on target approach**		Q
		Cause:	The actual value has not reached the "target range" within the "monitoring time for target approach".	
		Remedy:	• Optimize the "Target range" and the "Monitoring Time for Target Approach" (axis DB address 26.0, 30.0). • Check the drive and axis.	

130.5	EncoderErr	**Encoder error**		Q
		Cause:	• The input driver FB has detected an encoder error. The instance DB of the driver DB gives a precise indication of the cause of error. See Input Driver (FB 21 to FB 29). • The "EncErr" input was set at the input driver (see Input Driver, section on error handling)..	
		Remedy:	Rectify the error cause.	
130.6	OutputErr	**Error at output driver**		Q
		Cause:	At the output driver, input "OutErr" was set (see Output Driver, Section on Error Handling).	
		Remedy:	Rectify the error cause.	
130.7	ConfigErr	**Group error: Axis data incorrectly configured**		**Acknowledgement not possible**
		Cause:	Parameters in axis DB are defective. For exact cause of error, see Configuration Errors.	
		Remedy:	Change the defective parameter and make sure that FC MC_Init is launched.	
131.0	DriveErr	**Drive error**		**Acknowledgement not possible**
		Cause:	The output driver FB has detected an error at the power unit/drive. The instance DB of the driver DB gives a precise indication of the cause of error. See Output Driver (FB 31 to FB 37).	
		Remedy:	Eliminate the cause of error.	

8.1.3 Configuration Errors

The "Bit-Addr." column contains the address of the error display, the "Par.-Addr." column refers to the address of the defective parameter (see Axis Data block Setup).

Bit Addr.	Name Config.	Configuration Errors	Par. Addr.
132.0	Err_AxisLimit	"Software limit switch start" < -2^{24} [unit of length] "Software limit switch end" > $+2^{24}$ [unit of length] "Software limit switch end" <= "Software limit switch start"	6.0, 10.0
132.1	Err_MaxVelocity	"Maximum speed" <= 0.0 [unit of length/s]	14.0
132.2	Err_MaxAcceleration	"Maximum acceleration" <= 0.0 [unit of length/s^2]	18.0
132.3	Err_MaxDeceleration	"Maximum deceleration" <= 0.0 [unit of length/s^2]	22.0
132.4	Err_MonTimeTargetAppr	"Monitoring time for target approach" <= 0.0 [s]	26.0
132.5	Err_TargetRange	"Target range" <= 0.0 [unit of length]	30.0
132.6	Err_StandstillRange	"Standstill range" <= 0.0 [unit of length]	34.0
132.7	Err_MaxFollowingDist	"Maximum admissible following distance" <= 0.0 [unit of length]	38.0
133.0	Err_EmergencyDec	"Deceleration for hard stop" <= 0.0 [unit of length/s^2]	54.0
133.1	Err_StepsPerRev	"Steps per encoder revolution" < 1	64.0
133.2	Err_DisplacementPerRev	"Axis distance per encoder revolution" <= 0.0 [unit of length]	68.0
133.3	Err_NumberRevs	"Number of encoder revolutions" < 1 with absolute encoder	72.0
133.4	Err_PolarityEncoder	"Encoder polarity" not equal ±1	74.0
133.5	Err_PolarityDrive	"Drive polarity" not equal ±1	86.0
133.6	Err_DriveInputAtMaxVel	"Reference value for maximum axis velocity" <= 0.0 or > "DriveInputAt100"	88.0, 96
133.7	Err_AxisLength	Axis length > 2^{24} [unit of length] or Axis length > $2^{31}-1$ [steps]	6.0, 10.0
134.0	Err_EncoderRange	Encoder range does not match axis length (see Encoder Data, "Axis Distance per Encoder Revolution", Addr. 68.0)	6.0, 10.0 , 68.0, 72.0
134.1	Err_MaxVelRotaryAxis	Maximum velocity and scan time do not match rotary axis length (see Axis Data , "Maximum Velocity", Addr. 14.0)	14.0 , 0.0, 68.0, 72.0
134.2	Err_DriveInputAt100	"Reference value for 100% speed" <= 0	96

9 Sample Programs for Easy Motion Control

9.1 Introduction

The software package is also used to install a sample project which shows you typical applications of differing complexity and target direction based on various sample programs.

The English sample project is in the folder

...\STEP7\EXAMPLES\zEn20_02_EMC2.

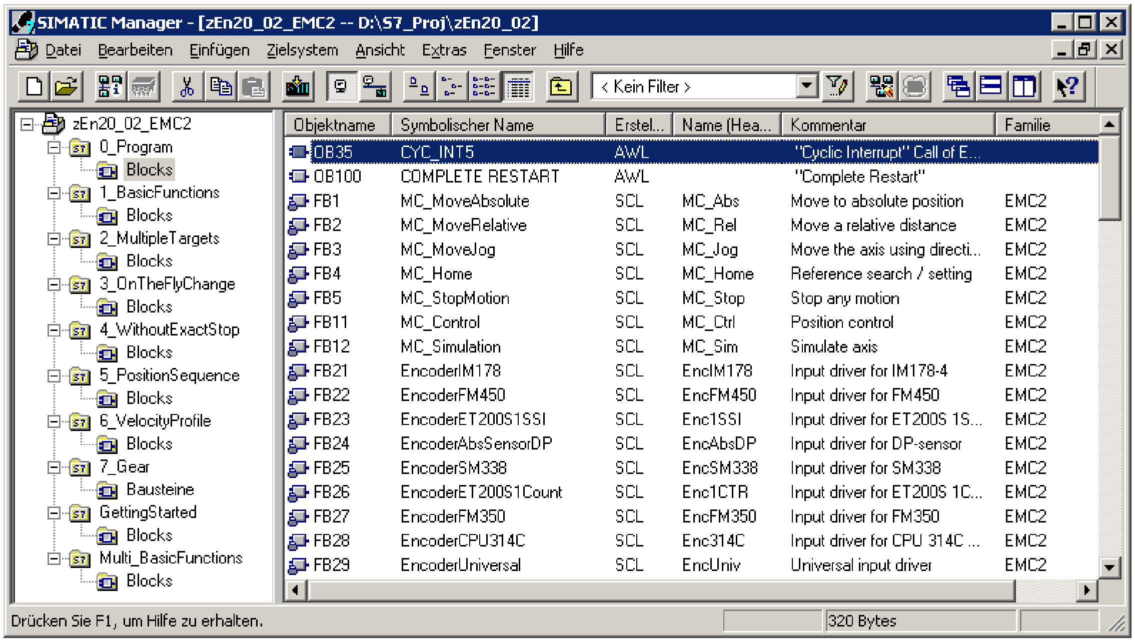
9.2 Requirements

- You have configured and wired an S7 station consisting of a power supply module, a CPU and the necessary I/O modules as well as a driver together with encoder.
- On your PC/PG, STEP 7 is installed as well as the configuration package for Easy Motion Control. The handling description is based on STEP 7 V5.0. With other versions, there may be divergences.
- The programming device is connected to the CPU.
- To ensure the safety of operating staff and the system itself, you have provided safety limit switches and EMERGENCY STOP switches.

9.3 Sample Project Structure

Easy Motion Control can be operated with different I/Os by means of appropriate driver blocks.

To enable you to adapt the examples easily to your existing hardware, the following structure was selected for the sample project:


SIMATIC Manager - [zEn20_02_EMC2 -- D:\S7_Proj\zEn20_02]
Datei Bearbeiten Einfügen Zielsystem Ansicht Extras Fenster Hilfe
< Kein Filter >

- zEn20_02_EMC2
  - 0_Program
    - Blocks
  - 1_BasicFunctions
    - Blocks
  - 2_MultipleTargets
    - Blocks
  - 3_OnTheFlyChange
    - Blocks
  - 4_WithoutExactStop
    - Blocks
  - 5_PositionSequence
    - Blocks
  - 6_VelocityProfile
    - Blocks
  - 7_Gear
    - Bausteine
  - GettingStarted
    - Blocks
  - Multi_BasicFunctions
    - Blocks

Objektname	Symbolischer Name	Erstel...	Name (Hea...	Kommentar	Familie
OB35	CYC_INT5	AWL		"Cyclic Interrupt" Call of E...	
OB100	COMPLETE RESTART	AWL		"Complete Restart"	
FB1	MC_MoveAbsolute	SCL	MC_Abs	Move to absolute position	EMC2
FB2	MC_MoveRelative	SCL	MC_Rel	Move a relative distance	EMC2
FB3	MC_MoveJog	SCL	MC_Jog	Move the axis using directi...	EMC2
FB4	MC_Home	SCL	MC_Home	Reference search / setting	EMC2
FB5	MC_StopMotion	SCL	MC_Stop	Stop any motion	EMC2
FB11	MC_Control	SCL	MC_Ctrl	Position control	EMC2
FB12	MC_Simulation	SCL	MC_Sim	Simulate axis	EMC2
FB21	EncoderIM178	SCL	EncIM178	Input driver for IM178-4	EMC2
FB22	EncoderFM450	SCL	EncFM450	Input driver for FM450	EMC2
FB23	EncoderET200S1SSI	SCL	Enc1SSI	Input driver for ET200S 1S...	EMC2
FB24	EncoderAbsSensorDP	SCL	EncAbsDP	Input driver for DP-sensor	EMC2
FB25	EncoderSM338	SCL	EncSM338	Input driver for SM338	EMC2
FB26	EncoderET200S1Count	SCL	Enc1CTR	Input driver for ET200S 1C...	EMC2
FB27	EncoderFM350	SCL	EncFM350	Input driver for FM350	EMC2
FB28	EncoderCPU314C	SCL	Enc314C	Input driver for CPU 314C ...	EMC2
FB29	EncoderUniversal	SCL	EncUniv	Universal input driver	EMC2

Drücken Sie F1, um Hilfe zu erhalten. 320 Bytes

"0_Program" and "1_......" to "7_......"

The "0_Program" is used as a framework for sample programs "1_......" to "7_......" (from now on referred to as "n_EXAMPLE").

You only have to adapt the hardware once in the "0_Program" by selecting the right drivers. You can then use the adapted "0_Program" in all examples.

An operable program consists of the blocks of the **"0_Program"** adapted to your hardware as well as the blocks from an **"n_EXAMPLE".**

"Multi_BasicFunctions"

Demonstrates the use of Easy Motion Control blocks in a multi-instance FB.

"GettingStarted"

With the Getting Started primer, you create an S7 program which you can compare with the "Getting Started" program of the sample project. This program is described in *Getting Started*.

9.4 Structure of Programs

Single instance

The **"0_Program"** contains:

- The Easy Motion Control blocks and corresponding instance DBs
- Two axis DBs
- UDT AXIS_REF
- OB100
  In OB100 a bit is set for recognizing restart.
- OB35
  OB35 starts the actual **sample program** by launching FB100 ("Example") with the latter's instance DB100 ("DB_Example").
  The Easy Motion control driver blocks necessary for adapting the hardware are also launched in OB35. As a commentary by way of example the drivers for an IM178-4).
  It is precisely here that you must make the abovementioned adjustment to your hardware.

The S7 programs **"n_EXAMPLE"** contain:

- FB100 ("Example"): program for controlling axis functions
- DB100 ("DB_Example"): instance DB of FB100
- Variable table: ("VAT_Example") for using the example.

"5_PositionSequence" and "6_VelocityProfile" in each case contain an additional data block which is used by the programs.

An operable program consists of the blocks of the **"0_Program"** adapted to your hardware as well as the blocks from an **"n_EXAMPLE".**

Multi-instance

The multi-instance example **"Multi_BasicFunctions"** differs from the single-instance examples in that, in multi-instance FB 100, the drivers and Init block are also started and thus only the FB is launched in OB35.

If you have already tested one of the single-instance examples and thus adapted it to your hardware, and then wish to execute the multi-instance example, you should save OB35 and axis DB of the single-instance example in "0_Program" before they are overwritten by OB35 from "Multi_BasicFunctions".

The S7 program **"Multi_BasicFunctions"** contains:

- OB35: Call of FB 111 ("Example_Multi")
- FB111 ("Example_Multi"): Program for controlling the axis functions
- Variable table: ("VAT_Example_Multi") for operator control of out sample.

Travel FB Parameters

Travel FBs of Easy Motion Control can be provided individually at every launch with a new velocity, acceleration and deceleration.

These parameters are dependent on your system and Configuration and cannot thus be indicated generally in the sample programs.

So that you do not have to adapt every individual block launch in the sample programs, these parameters are interconnected with the maximum values configured in Axis DB.

That means that the axis also travels with the maximum possible velocity. To prevent this, the parameter "Override" is set at 10% in the variable tables for using the examples and the velocity is thus restricted to 10% of the maximum.

9.5 Preparation of Examples

Single instance

1. In the SIMATIC Manager open sample project **zEn20_02_EMC2** in the folder **...\STEP7\EXAMPLES** and copy it into your project directory under a suitable name (**File > Save as**). This project remains open.
2. Insert a station into this project according to your hardware configuration and give it a name, e.g. **"ExampleStation"**.
3. Open **"ExampleStation"** and then "Hardware" and configure your system. Parameterize for Easy Motion Control
   - the cyclic interrupt of OB35 in the CPU with 10 ms (CPU 318 or S7-400) or 20 ms (S7-300 to CPU 316)and
   - the I/O according to recommendations in the chapters Input Driver (FB21 to FB29) and Output Driver (FB 31 to FB 37).
4. Save and compile the settings, close the HW Config and give a name to the S7 program of its CPU (**"Example"**).
5. Copy the symbol table and the block container contents from **"0_Program"** to **"Example".**
6. Open OB35 from **"Example"** in LAD/STL/FBD Editor, enter the driver block calls according to your hardware and save OB35.
   Please note that you must call two input and two output drivers for sample 7 "Basic functions".
7. Start the Easy Motion Control configuration interface and parameterize axis data in "DB_Axis" of **"Example"**.
   Also enter **0.01 s** or **0.02 s** here as scan time.
8. Copy the blocks of **"n_EXAMPLE"** that you have selected to **"Example"**.
9. If you wish to carry out further tests after testing this sample program, repeat Point 8.

Multiple instance

Go through Points 1 to 4 as described in the case of single-instance. Then proceed as follows:

1. Copy the blocks of the program **"Multi_BasicFunctions"** also to **"Example"**.
2. Open FB111 from **"Example"** in LAD/STL/FBD-Editor, enter the data declarations for the driver blocks that match your hardware (static variables #IN and #OUT as variables of the type of driver FBs.) and adjust the block calls accordingly.
   Then save FB111.
3. In SIMATIC Manager, delete DB111 from "Example" and regenerate this block as instance DB of FB111.
4. Start the Easy Motion Control configuration interface and parameterize axis data in "DB111 of **"Example"**.
   Also enter **0.01s** or **0.02s** as scan time here.

9.6 Executing Examples

After completing preparations, load **"ExampleStation"** into the CPU and bring these into RUN.

Please note:

- the input driver for CPU314C (FB28 "EncoderCPU314C") cannot be downloaded to a CPU3xx. You can ignore the corresponding error message during the download to the station.
- To avoid memory problems during the download, delete all driver blocks not required from the "Example" program, for example.

Testing Examples with Variable Tables

For testing the **sample programs**, you will in each case find a variable table "VAT_Example" Example 7, namely the "Gear functions", contains two VATs).

Open "VAT_Example" and establish a connection to the CPU. Select the view "Symbol" and "Symbol comments" and observe the variables cyclically. By entering control values and "Activate control values" you can change these values in the online data blocks.

When using variable tables, you should make sure that all control values that are **not** to be activated are empty or have a comment sign.

In the event of error ("DB_Axis.Ax.Error" is set) you can view detailed error information in the configuration software and acknowledge errors.

Testing Samples with OP 27

The sample project contains the object "OCM" of type "SIMATIC OP". This is a configuration for an operator panel OP 27 that is appropriate for the sample programs. If you have an OP 27, you can use the sample programs with the OP by means of "OCM".

To do this, load "OCM" into OP 27 with the ProTool configuration software.

For the necessary connections and operating stages, see the OP documentation.

The design of the OP project allows you to control **all** of the sample programs.

After power-up of the OP, a start picture appears for you to select the picture appropriate to your example. This contains the control elements necessary for executing the sample program as well as output and input fields.

The errors detected by Easy Motion Control are displayed on the OP as fault signals requiring acknowledgement. Acknowledging these messages is, however, not synonymous with the error acknowledgement required for Easy Motion Control. You can generate this using a special acknowledgement button in the OP pictures.

The OP uses so-called flags to trigger these error messages. These point directly to the axis error areas of the axis data used in the samples. The OP verifies these areas cyclically and reports an error if one of the addressed DBs is not loaded in the CPU. Even if you do not require this DB for the current sample program, you should nevertheless download it after you have received this message,.

Testing Examples in Simulation Mode

You can test the function of the examples even without axis I/Os by only using the variables according to the primer and, for example, observing axis position in order to observe the correct implementation of the program.

For this purpose, before the input driver, also launch FB MC_Simulation and set the bit "Sim" in the axis DB.

With single-instance examples, "before the input driver" means in the OB35 of the "Example", with multi-instance examples, in the FB100.

9.7 Example 1: Basic Functions

Target

In this example, the basic functions of jogging, reference setting, reference search, absolute/relative positioning and stopping travel are realized:

- With "jogging", you can move the axis arbitrarily within the work range with the help of two direction bits.
- With "reference setting" you adjust the actual position value of the axis at a standstill to the preset value.
- With "reference search" you move the axis until it reaches a reference point. The preset coordinate value is assigned to this position.
- With "absolute positioning", you move the axis onto a target within the work range.
- With "relative positioning", you move the axis within the work range by a specific distance in relation to the starting point.
- With "Stop" you interrupt current travel with the specified braking deceleration.

Operation

Open "VAT_Example", establish the connection to the configured CPU and observe the variables.

Set drive enable:
First the drive enable has to be issued. For this, set the bit "DB_MC_Control.EnableDrive" and activate the control value.

Acknowledge error:
After issuing the drive enable, "DB_Axis.Ax.Err.EnableDriveErr" must be acknowledged. For this, set the bit "DB_Axis.Ax.ErrorAck" and activate the control value or use the acknowledgement function of the configuration software.

Jogging:
Set and activate the required direction bit "DB_MC_MoveJog.DirPos" or "DB_MC_MoveJog.DirNeg".

To stop the movement, reset the direction bit to "0".

Reference setting:
Requirement:
The axis must be at a standstill.
In order to assign a coordinate value to the current axis position, enter its value in "DB_MC_Home.Position" and set "DB_MC_Home.Velocity" at 0.0.

By setting "DB_MC_Home.Execute" the function is executed: the current actual position "DB_Axis.Ax.ActPosition" takes over the reference point coordinate and the feedback "DB_Axis.Ax.Sync" is set.

Reference searching:
This function is **only possible with incremental encoders** and requires an initiator on the axis which serves as a reference point switch. (Please observe the wiring and configuration information applicable for your hardware, according to sections Input Drivers (FB21 to FB29) and Output Drivers (FB 31 to FB 36).)

This position should be assigned a coordinate value on overriding the initiator. Enter the coordinate value in "DB_MC_Home.Position" and set "DB_MC_Home.Velocity" at a value **over 0.0**.

Note

Set "DB_MC_Home.Direction" so that the axis actually moves in the direction of the reference point switch.

By setting "DB_MC_Home.Execute" the function is started: the axis travels until the reference point is notified, the reference point coordinate is accepted as the new actual value and the axis slows down.

The feedback "DB_Axis.Ax.Sync" is set.

Absolute Positioning:
In "DB_MoveAbs1.Position" select the target position to be reached and set "DB_MoveAbs1.Execute" in order to start travel: the axis accelerates, travels at constant velocity and then slows down so as to come to a standstill at the specified target (whereby you should observe "DB_Axis.Ax.ActPosition").

On reaching the target, you will observe that "DB_MoveAbs1.Execute" changes back to "0"; this is implemented in FB100 to allow FB MC_MoveAbsolute to detect the edge for the next start.

Relative Positioning:
In "DB_MoveRel.Distance" select the distance to be covered and set "DB_MoveRel.Execute" to start travel: the axis accelerates, travels at constant velocity and then slows down so as to come to a standstill after "Distance" (whereby you should observe "DB_Axis.Ax.ActPosition").

Interrupting Positioning:
If you wish to abort travel before its target has been reached, set the bit "DB_MC_StopMotion .Execute". The axis slows down with the deceleration which you have set at the parameter "DB_MC_StopMotion .Deceleration".

Error Evaluation:
If an error occurs while executing a travel block, the axis stops. (Axis error, "DB_Axis.Ax.Error" is set). You can view detailed error information in the configuration software and acknowledge the error.

Configuration errors indicate errors in the axis data; they are detected during initialization using FC MC_Init and cannot be acknowledged. After correcting the parameters, re-initialization is necessary (launch FC MC_Init).

9.7.1 Extension: Stopping and Axis Disconnection When Opening a Safety Door

Target

When a specific event occurs such as opening a safety door, every active travel movement should be specifically aborted and then the position controller input ("DB_MC_Control.EnableDrive") disconnected.

By **wiring** the position controller output "DB_MC_Control.DriveEnabled" with the enable input of the drive controller, the drive is also disconnected.

Extension

The extension uses the three bits "Safetydoor_open", "Safetydoor_Ack" and "Edge".

Open FB100 and cancel the jump command which jumps directly to Network 2.

Operation

In "VAT_Example" the bits "Safetydoor_open" and "Safetydoor_ack" are already entered.

Every time "Safetydoor_open" is set, the travel FB which is currently active is interrupted by starting FB MC_StopMotion .

If FB MC_StopMotion has finished its braking process and the door is still open, the enable input of the position controller is reset and thus, with appropriate wiring (see above) the drive is disconnected.

After closing the safety door ("Safetydoor_open" = FALSE) and setting "Safetydoor_ack", the axis error is acknowledged and the controller is enabled, the axis returns to controlled mode.

9.8 Example 2: Positioning between Two Target Points which can be Changed Arbitrarily

Target

You can select between two targets and position on these by specifying two target positions and control from two different start bits.

If, during positioning on target 1, a start command is issued for target 2, there is a smooth switchover to positioning onto target 2. If target 2 requires the axis to change direction, it is first of all brought to a standstill and then positioned in the opposite direction on target 2.

The same applies to overriding travel to target 2 with a journey to target 1.

Operation

Open the variable table "VAT_Example", establish the connection to the configured CPU and observe the variables.

Set drive enable:
First the drive enable has to be issued. For this, set the bit "DB_MC_Control.EnableDrive" and activate the control value.

Acknowledge error:
After issuing the drive enable, "DB_Axis.Ax.Err.EnableDriveErr" must be acknowledged. For this, set the bit "DB_Axis.Ax.ErrorAck" and activate the control value or use the acknowledgement function of the configuration software.

Reference setting:
Requirement:
The axis must be at a standstill.
In order to assign a coordinate value to the current axis position, enter its value in "DB_MC_Home.Position" and set "DB_MC_Home.Velocity" at 0.0.

By setting "DB_MC_Home.Execute" the function is executed: the current actual position "DB_Axis.Ax.ActPosition" takes over the reference point coordinate and the acknowledgement "DB_Axis.Ax.Sync" is set.

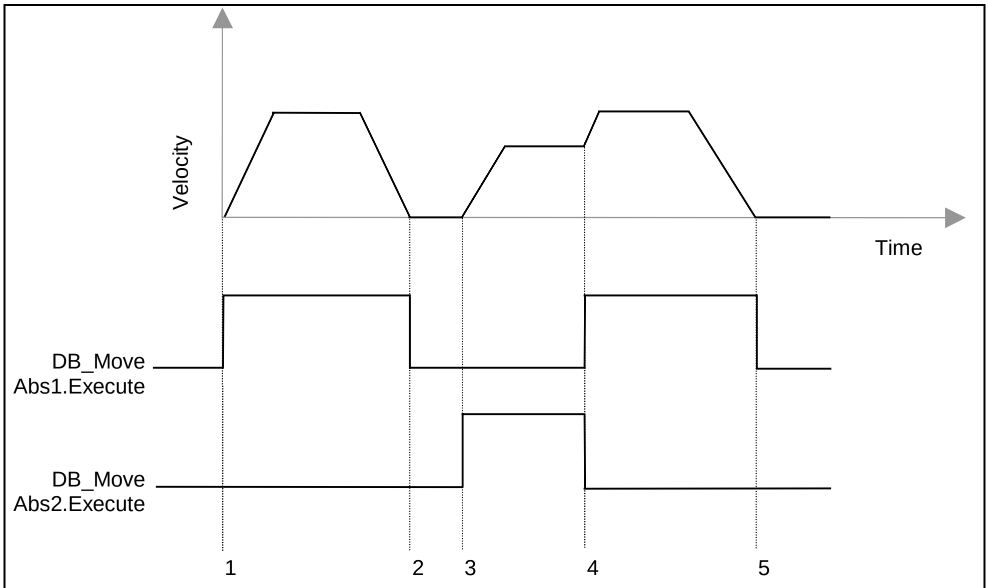
Positioning on Target 1:
To position the axis on the first target, select a target position in "DB_MoveAbs1.Position" and set the bit "DB_MoveAbs1.Execute". Transfer these values in order to start travel.

Positioning on Target 2:
To position the axis on the second target, select a target position in "DB_MoveAbs2.Position" and set the bit "DB_MoveAbs2.Execute". Transfer these values in order to start travel.

In order to allow better detection of override, a lower value was selected for velocity when traveling to target 2 than with target 1 (see diagram).

Interrupting Positioning:
If you wish to abort travel before target has been reached, set the bit "DB_MC_StopMotion .Execute". The axis slows down with the deceleration which you have set at the parameter "DB_MC_StopMotion .Deceleration".

Flow Diagram



1 Select Target 1 in "DB_MoveAbs1.Position" and set "DB_MoveAbs1.Execute". The axis starts in direction Target.

2 "DB_MoveAbs1.Position" is reached. "DB_MoveAbs1.Execute" will be reset by the sample program.

3 Select Target 2 in "DB_MoveAbs2.Position" and set "DB_MoveAbs2.Execute". The axis starts in direction Target.

4 During the movement to Target 2, provide "DB_MoveAbs1.Position" with a **new** Target and set "DB_MoveAbs1.Execute". As a result, it will move to this new target immediately and "DB_MoveAbs2.Execute" will be reset.

5 "DB_MoveAbs1.Position" is reached. "DB_MoveAbs1.Execute" will be reset by the sample program.

9.9 Example 3: On-the-Fly Change from Movement to Positioning

Target

This example allows you to change from a "jogging" movement into a positioning motion "on-the-fly". The axis does not stop during the transition, unless the positioning target requires the axis to change direction.

Overriding of positioning motion by a jogging order is prevented in the program.

Operation

Open the variable table "VAT_Example", establish the connection to the configured CPU and observe the variables.

Set drive enable:
First the drive enable has to be issued. For this, set the bit "DB_MC_Control.EnableDrive" and activate the control value.

Acknowledge error:
After issuing the drive enable, "DB_Axis.Ax.Err.EnableDriveErr" must be acknowledged. For this, set the bit "DB_Axis.Ax.ErrorAck" and activate the control value or use the acknowledgement function of the configuration software.

Reference setting:
Requirement:
The axis must be at a standstill.
In order to assign a coordinate value to the current axis position, enter its value in "DB_MC_Home.Position" and set "DB_MC_Home.Velocity" at 0.0.

By setting "DB_MC_Home.Execute" the function is executed: the current actual position "DB_Axis.Ax.ActPosition" takes over the reference point coordinate and the acknowledgement "DB_Axis.Ax.Sync" is set.

Jogging:
Set and activate the required direction bit "DB_MC_MoveJog.DirPos" or "DB_MC_MoveJog.DirNeg".

(To stop the movement, you can reset the direction bit to "0" but then you will not see any flying change.)

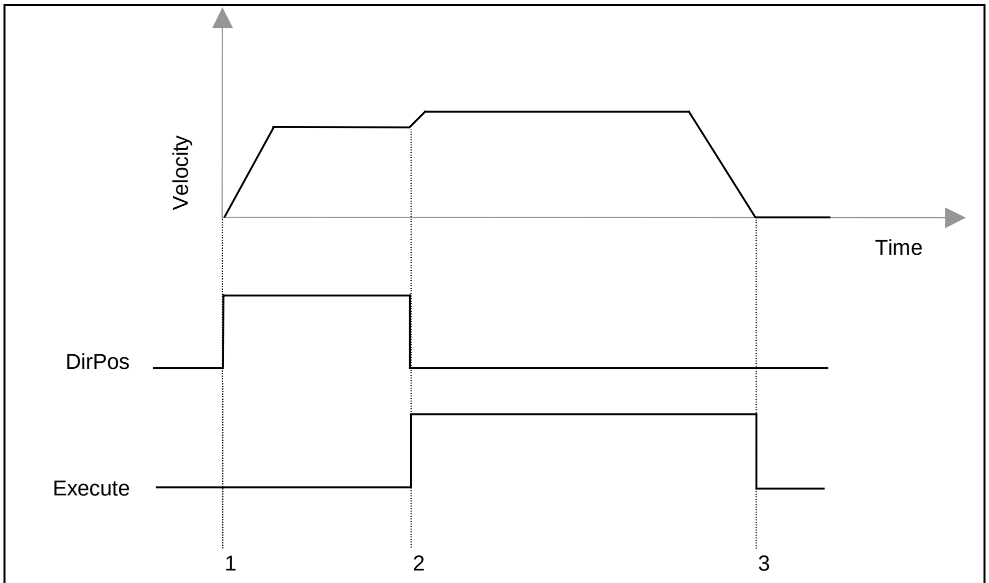
Executing the Positioning Motion:
In "DB_MoveAbs1.Position" select the target position to be reached and set "DB_MoveAbs1.Execute" in order to start travel: the axis accelerates, travels at constant velocity and then slows down so as to come to a standstill at the specified target (see "DB_Axis.Ax.ActPosition").

Any active jogging motion is immediately interrupted.

In order to allow better detection of override, a higher value was selected for positioning velocity than with jogging (see diagram).

Interrupting a Motion:
If you wish to abort travel before the target has been reached, set the bit "DB_MC_StopMotion .Execute". The axis slows down with the deceleration which you have set at the parameter "DB_MC_StopMotion .Deceleration".

Flow Diagram



1 Start the movement by setting the "DB_MoveJog.DirPos".

2 Select a target in "DB_MoveJog.DirPos" and set "DB_MoveAbs.Execute". The axis accelerates and moves to the direction of the target, "DB_MoveJog.DirPos" will be reset by the sample program.

3 The target is reached. "DB_MoveAbs.Execute" will be reset by the sample program.

9.10 Example 4: Positioning without Exact Stop

Target

In the first step, start a positioning. If you set the start for the second position before reaching the target, the current process is continued until a configured distance is reached to the first target. The axis then travels towards the new target without coming to a standstill in the meantime, unless the new target requires the axis to change direction.

Operation

Open the variable table "VAT_Example", establish the connection to the configured CPU and observe the variables.

Set drive enable:
First the drive enable has to be issued. For this, set the bit "DB_MC_Control.EnableDrive" and activate the control value.

Acknowledge error:
After issuing the drive enable, "DB_Axis.Ax.Err.EnableDriveErr" must be acknowledged. For this, set the bit "DB_Axis.Ax.ErrorAck" and activate the control value or use the acknowledgement function of the configuration software.

Reference setting:
Requirement:
The axis is at a standstill
In order to assign a coordinate value to the current axis position, enter its value in "DB_MC_Home.Position" and set "DB_MC_Home.Velocity" at 0.0.

By setting "DB_MC_Home.Execute" the function is executed: the current actual position "DB_Axis.Ax.ActPosition" accepts the reference point coordinate and the acknowledgement "DB_Axis.Ax.Sync" is set.

Select Switching Value:
In parameter "DB_Example.Distance" enter the distance to the current target at which switching to the next position should occur if a new start is to be performed.

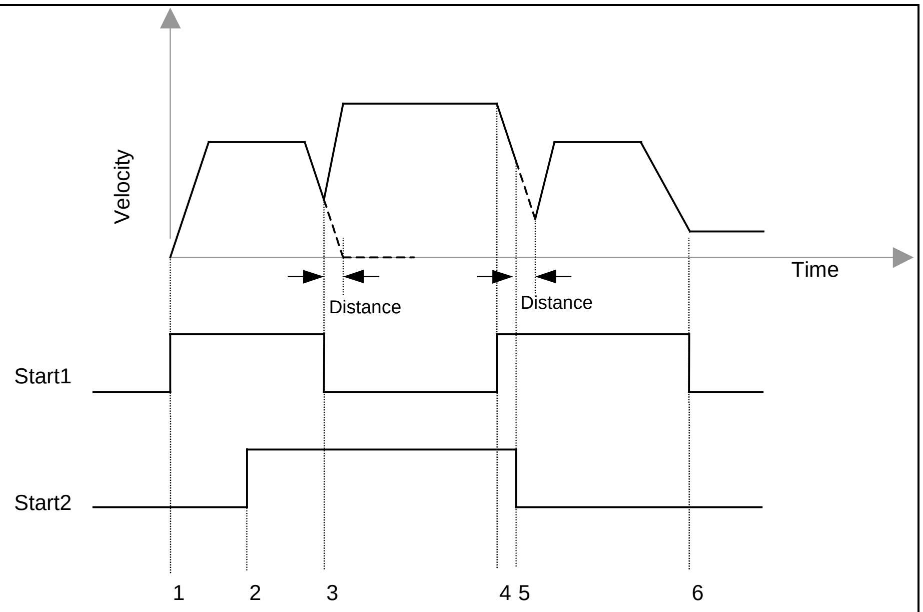
Positioning with MoveAbs1:
To position the axis on the first target, select a target position in "DB_MoveAbs1.Position" and transfer this value. Now set the bit "DB_Example.Start1" to start the motion.

Positioning with MoveAbs2:
To position the axis on the second target, select a target position in "DB_MoveAbs2.Position" and transfer this value. Now set the bit "DB_Example.Start2" to start the motion.

In order to allow better detection of override, a higher value was selected for velocity when positioning on target 2 (see diagram).

Interrupting a Motion:
If you wish to abort travel before its target has been reached, set the bit "DB_MC_StopMotion .Execute". The axis slows down with the deceleration which you have set at the parameter "DB_MC_StopMotion .Deceleration".

Flow Diagram



1 Select Target 1 in "DB_MoveAbs1.Position" and set "DB_Example.Start1". The axis starts in direction Target 1.

2 Select the new target in "DB_MoveAbs2.Position" and set "DB_ Example.Start2". The axis moves temporarily further in direction Target 1.

3 The selected distance ("DB_ Example.Distance") is reached, the program resets "DB_ Example.Start1". Now "DB_ Example.Start2" is active with Target 2.

4 Provide "DB_MoveAbs1.Position" with a new target and set "DB_ Example.Start1".

5 The selected reversing point ("DB_ Example.Distance") is reached, the program resets "DB_ Example.Start2", "DB_ Example.Start1" is active with "DB_MoveAbs1.Position".

6 "DB_MoveAbs1.Position" is reached. "DB_ Example.Start1" will be reset by the sample program.

9.11 Example 5: Sequential Travel to 20 Different Positions

Target

This example shows how a list of target positions and travel parameters filed in one data block is processed sequentially. Each start positions on the position number filed in a block parameter and increases this position number.

Operation

Provide DB_Pos with Target Positions:
In "DB_POS_05", enter the target positions and velocities at which the respective target should be reached. The data block example "DB_POS_05" contains 20 target positions and has the following configuration:

```
STRUCT
   MaxPosNo    : INT  := 20;  //Maximum number of positions
   POSITION_01 : REAL ;       //Position 1
   VELOCITY_01 : REAL ;       //Velocity 1
   POSITION_02 : REAL ;       //Position 2
   VELOCITY_02 : REAL ;       //Velocity 2
   POSITION_03 : REAL ;       //Position 3
   VELOCITY_03 : REAL ;       //Velocity 3
   POSITION_04 : REAL ;       //Position 4
```

etc.

Open "VAT_Example", establish the connection to the configured CPU and observe the variables.

Set drive enable:
First the drive must be enabled. For this, set the bit "DB_MC_Control.EnableDrive" and activate the control value.

Acknowledge error:
After issuing the drive enable, "DB_Axis.Ax.Err.EnableDriveErr" must be acknowledged. For this, set the bit "DB_Axis.Ax.ErrorAck" and activate the control value or use the acknowledgement function of the configuration software.

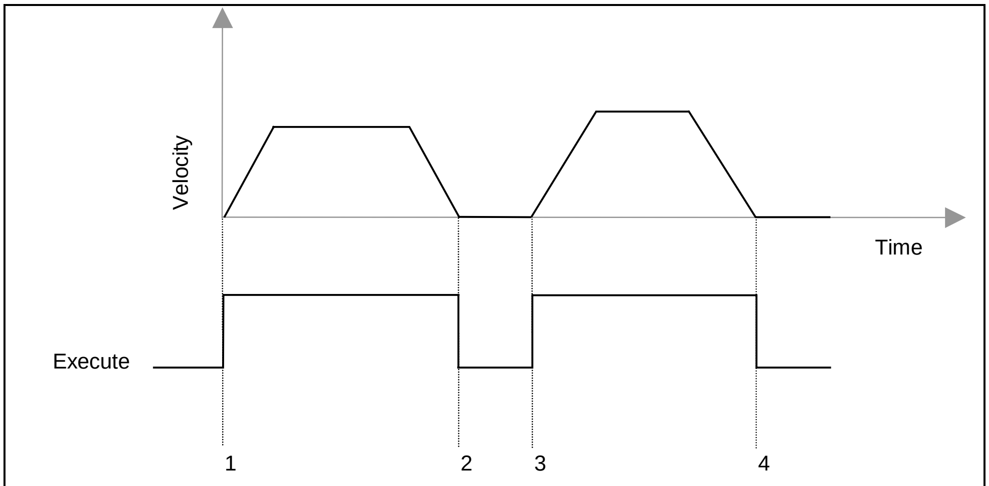
Reference setting:
Requirement:
The axis must be at a standstill.
In order to assign a coordinate value to the current axis position, enter its value in "DB_MC_Home.Position" and set "DB_MC_Home.Velocity" at 0.0.

By setting "DB_MC_Home.Execute" the function is executed: the current actual position "DB_Axis.Ax.ActPosition" accepts the reference point coordinate and the feedback "DB_Axis.Ax.Sync" is set.

Executing Positioning:
At parameter "DB_Example.PosNo" enter the position number of the target which is to be reached with the next start. Start positioning by setting the bit "DB_MoveAbs1.Execute". On reaching the target, the sample program resets the bit "DB_MoveAbs1.Execute" at "0", at the same time the value in "DB_Example.PosNo" is increased by 1. When "DB_MoveAbs1.Execute" is next set, the process is repeated and the list in the DB is processed.

Interrupting Positioning:
If you wish to abort travel before the target has been reached, set the bit "DB_MC_StopMotion .Execute". The axis slows down with the deceleration which you have set at the parameter "DB_MC_StopMotion .Deceleration".

Flow Diagram



1 Set "DB_MoveAbs1.Execute". The axis starts in direction of the target, of which the position number can be selected in "DB_ Example.PosNo".

2 The position is reached. "DB_MoveAbs1.Execute" will be reset by the sample program, "DB_ Example.PosNo" is increased to 1.

3 Reset "DB_MoveAbs1.Execute". The axis starts in direction of the next position.

4 The position is reached. "DB_MoveAbs1.Execute" will be reset by the sample program, "DB_Prog.PosNo" is increased to 1.

9.12 Example 6: Traveling to a Position with Distance-dependent Velocity Profile

Target

With this example, while traveling to a target position, you can travel on freely selectable sections of the stretch at 5 different velocities.
In the data block "DB_POS_06" 4, switching positions are filed with the corresponding velocities. The actual target is in Position 5.

During travel, after reaching each switching position, there is a switch to the new velocity without the axis stopping.

The switching positions filed in "DB_POS_06" must lead to the target position in ascending or descending order from the current actual position. If this is not the case, the program searches out a place in the table that is suitable for the target position.

Operation

Provide "DB_POS_06" with Target Positions:
In "DB_POS_06", enter the 4 required switching positions, the target and velocities at which to travel.

"DB_POS_06" has the following configuration:

```
STRUCT
   ActPosNo   : INT  ;       //Number of actual switching position
   POSITION_1 : REAL ;       //Switching position 1
   VELOCITY_1 : REAL ;       //Velocity 1
   POSITION_2 : REAL ;       //Switching position 2
   VELOCITY_2 : REAL ;       //Velocity 2
   POSITION_3 : REAL ;       //Switching position 3
   VELOCITY_3 : REAL ;       //Velocity 3
   POSITION_4 : REAL ;       //Switching position 4
   VELOCITY_4 : REAL ;       //Velocity 4
   POSITION_5 : REAL ;       //Target position
   VELOCITY_5 : REAL ;       //Velocity 5
```

Open "VAT_Example", establish the connection to the configured CPU and observe the variables.

Note

The switching positions should form an ascending or descending sequence between start position (current actual position) and target position ("Position_5"). Only in this way is it possible to travel the full profile.

Set Drive Enable:
First the drive enable has to be issued. For this, set the bit "DB_MC_Control.EnableDrive" and activate the control value.

Acknowledge Error:
After issuing the drive enable, "DB_Axis.Ax.Err.EnableDriveErr" must be acknowledged. For this, set the bit "DB_Axis.Ax.ErrorAck" and activate the control value or use the acknowledgement function of the configuration software.

Reference Setting:
Requirement:
The axis must be at a standstill.
In order to assign a coordinate value to the current axis position, enter its value in "DB_MC_Home.Position" and set "DB_MC_Home.Velocity" at 0.0.

By setting "DB_MC_Home.Execute" the function is executed: the current actual position "DB_Axis.Ax.ActPosition" accepts the reference point coordinate and the feedback "DB_Axis.Ax.Sync" is set.

Executing Positioning:
By setting the bit "DB_Example.Start" you start the positioning. During travel, the current switching position is displayed in "DB_POS_06.ActPosNo".

Also watch the current setpoint and actual velocity (observing the velocity override) in "DB_Example.NewSpeed" or "DB_Axis.Ax.ActVelocity".

These values change according to the current position number. On reaching the target, the sample program resets the bit "DB_Example.Start" to "0".

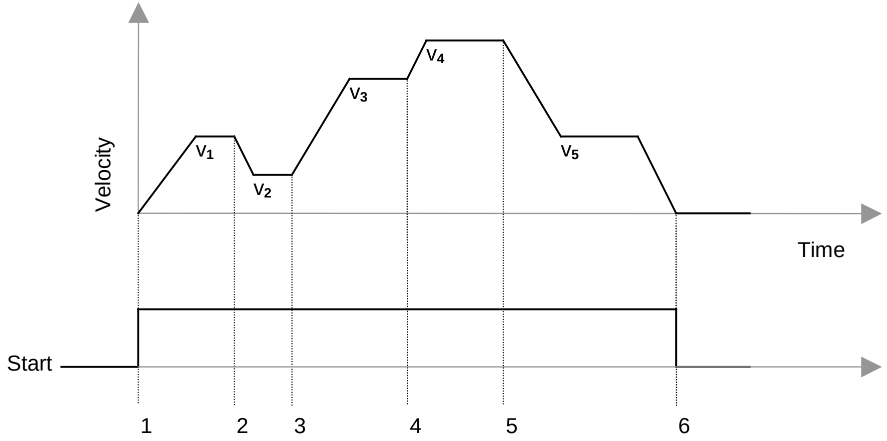
Interrupting Positioning:
If you wish to abort travel before the target has been reached, set the bit "DB_MC_StopMotion .Execute". The axis slows down with the deceleration which you have set at the parameter "DB_MC_StopMotion .Deceleration".

Repeating the Example

If you wish to repeat the example with the same data record, you can for example reset the current actual position value to the start value using the reference setting function (if your mechanical configuration permits this).

If you enter the original start value as the new target position in "DB_POS_06", the axis travels back to the start at constant velocity and you can repeat travel with a new target position.

Flow Diagram



1 Set "DB_ Example.Start". The axis starts in direction of the target which you have entered in "DB_POS_06.Position_5".

2 Reversing point_1 is reached. As in the example v_2 is smaller than v_1 , the axis will be slowed down and moves with v_2 till the reversing point_2.
This procedure repeats itself accordingly at other reversing points.

3 ,4 ,5 .....

6 The target position is reached. "DB_ Example.Start" will be reset by the sample program.

9.13 Example 7: Gear Functions

Target

This sample program provides functions for demonstrating the gear functionality of Easy Motion Control. After having commissioned both axes separately, you can call a gear block in order to set master mode for axis 1 and slave mode for axis 2.

The following functions are available for commissioning both the master and the slave axes:

- In "jog" mode, you can move the axis arbitrarily within the work range with the help of two direction bits.
- With "reference setting", you calibrate the actual position value of the axis at a standstill to a preset value.
- With "reference search" you move the axes until they reach a reference point. This position is assigned to the preset coordinate value.

Additionally, the gear functionality is available:

- At the "Gear ratio 1" and "Gear ratio 2" parameters, you set the various coupling ratios between the **master** and **slave** axis and enable either gear ratio 1 or 2.
- In "Jog" mode, move the **master** axis into positive or negative direction of travel. The **slave** axis follows this motion at the set ratio if a gear is enabled.
- Use the "Absolute Positioning" function to run the **master** axis into a presettable position. Also applicable is: The **slave** axis follows this motion at the set ratio if a gear is enabled.
- Use the "Stop" command to disengage the axis coupling.

Note

In order to be able to operate two axes you need to:

- Configure the axis data (Easy Motion Control configuration interface)

and in OB35, call

- an input driver
- an output driver
- MC_Init

for the second axis also!

(You will find the syntax of these calls in a comment in OB35)

Operator control of the basic functions

The same basic functions are available for the master and for the slave axis. In the program, the primary difference between both of them is found in the axis and instance DBs. The name of the DBs reflects this difference.

Example:	Axis DB of the master axis:	DB_M_Axis
	Axis DB of the slave axis:	DB_S_Axis

In this text this is indicated with an (X).

Commission both axes by means of the basic functions and then test the gear functions.

Open "VAT_Example", log on to configured CPU and monitor the variables.

Enabling the drives:
You first need to enable the drives by setting the "DB_(X)_Control.DriveEnable" bit and activating the control value.

Error acknowledgement:
After you have enable the drives, you need to acknowledge "DB_(X)_Axis.Ax.Err.EnableDriveErr". To do so, set the "DB_(X)_Axis.Ax.ErrorAck" bit and activate the control value, or use the acknowledgement function of the configuration software.

Jog mode:
Set and enable the desired direction bit "DB_(X)_Jog.DirPos" or "DB_(X)_Jog.DirNeg".

To Stop the motion, reset the direction bit to "0".

Setting the reference point:
Prerequisite: The axis is at a standstill.

To assign a coordinate value to the current axis position, enter this value in "DB_(X)_Home.Position" and set "DB_(M)_Home.Velocity" to 0.0.

The function is executed by setting "DB_(X)_Home.Execute": the process variable "DB_(X)_Axis.Ax.ActPosition" accepts the value of the reference point coordinate, and the "DB_(X)_Axis.Ax.Sync" feedback is set.

Reference point approach:

This function is available **only with incremental encoders** and requires a **proximity switch on the axis**, which is used as reference point switch Please note the wiring and configuration instructions for your hardware.

This point at which the axis travels over the proximity switch should be assigned a coordinate value. Enter this coordinate value in "DB_(X)_Home.Position" and set a value > 0.0 at "DB_(X)_Home.Velocity".

Set "DB_(X)_Home.Direction" in such a way so as to ensure that the axis approaches the reference point switch.

The function is executed by setting "DB_(X)_Home.Execute": The axis is driven up to the reference point until the feedback signal is issued, the reference point coordinate is then input as new process variable and the axis is brought to a halt.

The feedback signal "DB_(X)_Axis.Ax.Sync" is set.

Error evaluation:
The axis stops if an error occurs during the execution of a motion block (axis error "DB_(X)_Axis.Ax.Error" is set). You can open the configuration software to view detailed error information and to acknowledge the error.

Parameter errors are an indication of faulty axis data; FC MC_Init detects these in the initialization process, and you cannot acknowledge them. You need to restart initialization by calling FC MC_Init after you have corrected the parameters.

Operator control of the gear functions

The slave axis is coupled to the master axis by means of a gear block and follows the motion of the master axis according to the set gear ratio.

A **slave** axis may reach its physical limits ("MaxVelocity") if a high gear ratio is set and the speed of the master axis is too high. In this situation, the required gear ratio between the two axes can not be maintained (the "Coupled" parameter is set FALSE)

You can avoid this situation, for example, by setting the master axis to a value that allows the slave axis to travel at its a maximum velocity, i.e. "MaxVelocity". The sample program demonstrates this at MC_MoveJog for the **master** axis: Its velocity, acceleration and deceleration is limited in such a way that, by taking the active gear ratio and a 10% reserve into account, the **slave** axis does not violate its limit values and thus remains continuously "Coupled".

A gear block can be called either when the master axis is in motion or when it is at a standstill. The start of a new gear block overrides any axis motion, i.e. including a gear motion. This functionality is used in the sample program to set and enable a different gear ratio.

The coupling operation is terminated either by the call of the stop block or by the start of another motion block.

Open "VAT_Example_G", log on to the CPU and monitor the variables.

Set gear ratio 1:
Set gear ratio 1 by selecting the corresponding "DB_S_Gear1.RatioNumerator" and "DB_S_Gear1.RatioDenominator".

Set gear ratio 2:
Set gear ratio 1 by selecting the corresponding "DB_S_Gear2.RatioNumerator" and "DB_S_Gear2.RatioDenominator".

Enable gear 1 or 2:
Enable one of the two gears by setting "DB_S_Gear1.Execute" or "DB_S_Gear2.Execute". (ratio 2 will take priority over 1 if you set "Execute" at both, because the corresponding gear block in the FB "Example" is called in second place and thus immediately overrides the first block.)

Disable the gears:
Disable the gears by setting "DB_S_Stop.Execute".

Jog mode:
Set and enable the desired direction bit "DB_M_Jog.DirPos" or "DB_M_Jog.DirNeg" to move the master axis.

To stop the axis, reset the direction bit to "0".

If one of the two gears is enabled, the slave axis follows with the set gear factor as long as you run the master axis in "Jog mode".

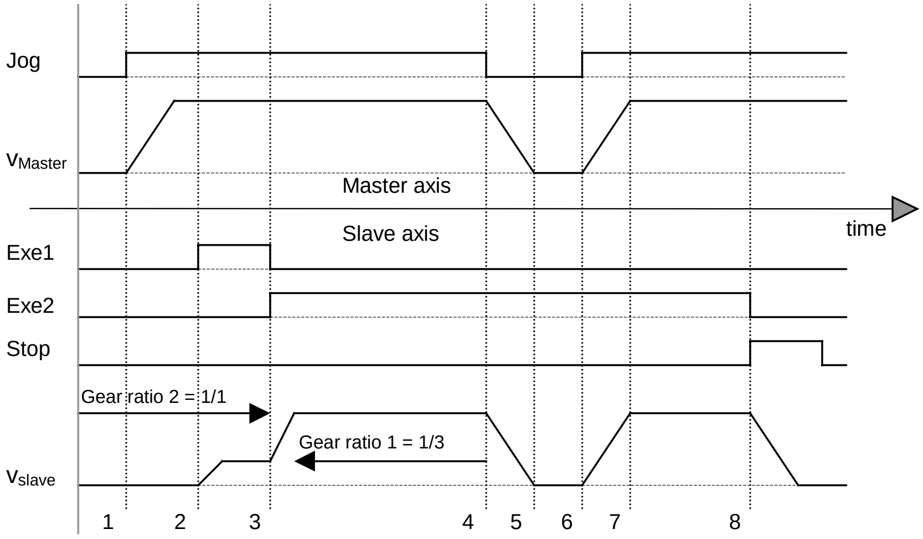
Absolute positioning:
In "DB_M_Abs1.Position", select the target position and then set "DB_M_Abs1.Execute", to execute the motion: The axis accelerates, is driven at a constant speed and decelerate to a standstill at the target position.

If one of the two gears is enabled, the slave axis follows with the set gear factor as long as you run the master axis.

Modifying the gear factor:
To set a new gear factor, open an inactive gear block, edit the values of "DB_S_Gear?.RatioNumerator" and/or "DB_S_Gear?.RatioDenominator" and then set "DB_S_Gear?.Execute". monitor the axis position and the travel velocity during the motion at "DB_(X)_Axis.Ax.ActPosition" and "DB_(X)_Axis.Ax.ActVelocity", as well as the status bit of the coupling at "DB_S_Gear?.Busy" and "DB_S_Gear?.Coupled".

"DB_S_Gear?.Busy" indicates that the gear block is being executed. If the slave axis cannot follow the master axis due to its set motion parameters, "DB_S_Gear?.Coupled" is set. The speed ratio between the master and slave axes reflects the gear ratio.

Flow Diagram



1 Start the master axis by setting "DB_M_Jog.DirPos". The slave axis is not in motion, because neither a run command nor a gear has been set for it.

2 by setting "DB_S_Gear1.Execute" = TRUE, the slave axis is coupled to the master axis using gear ratio 1 **(=1/3)**. The slave axis is immediately accelerate to the velocity required, since the master axis is already in motion.

3 By setting "DB_S_Gear2.Execute" = TRUE, the slave axis is coupled to the master axis using gear ratio 2 **(=1/1)**. (Gear block 1 is overridden and its Execute is reset in the sample program). The slave axis is accelerated, since gear ratio 2 is higher than gear ratio 1.

4 The master axis is halted, because "DB_M_Jog.DirPos" = FALSE is set. The slave axis follows, but the coupling is maintained during the period of standstill.

5 The master and slave axes are halted.

6 The master and slave axes accelerate, because "DB_M_Jog.DirPos" is reset to TRUE and gear ratio tow is active.

7 Both axes reach their speed setpoint.

8 "DB_S_Stop.Execute" = TRUE cancels the coupling. The slave axis is brought to a halt, while the master axis continues its motion (because of "DB_M_Jog.DirPos" = TRUE). The Execute of gear block 2 is reset in the program.

9.14 Example Multi: Basic Functions as Multi-instance FB

Target

This sample is similar to sample 1 and demonstrates the basic functions, i.e. jog mode, reference setting , absolute/relative positioning and stopping a run, but uses a **multiple instance FB** (frame FB) instead.

The frame FB transfers the parameters required for operator control and monitoring between its sheet bar and the parameters of the EMC blocks or axis data.

The result is a "trim" interface to Easy Motion Control, which is easy to control but does not offer the full functionality of the blocks. Should you require further functionalities, do so by adding appropriate expansions to your program and to the parameter bar.

Of course, you can also access instance data without having to work your way around via the parameter bar.

Program structure

The user program, including all Easy Motion Control block calls, axis data as well as the I/O drivers should be integrated into a **single** frame FB ("Example_Multi").

The handling of the current sample program differs from that of others precisely for this reason: Since you are the only one who knows which hardware will be used to run the program, it is up to you to integrate the appropriate I/O drivers in the FB.

The prepared frame FB contains the runtime program, including all Easy Motion Control block calls, their instance data, the axis data and, as representative, one call respectively for the I/O drivers.

You implement suitable I/O drivers for your plant, by declaring the static variables **#In** and **#Out** of the type of your driver blocks **and** entering the parameters of these drivers (see Preparing the Samples).

Parameters

Specify the velocity as well as the acceleration and deceleration ramp at the **inputs**. Specify the distance or the target for relative or absolute positioning. Within the frame FB, these parameter are interconnected permanently with the parameters of the motion block.

The start signals for the motion blocks must be interconnectable in the user program, but must be able of being reset in the frame FB. They are therefore declared as **in/out parameters**.

The synchronization/error status as well as the current position and velocity values of the axis are interconnected at the **outputs**. The error signals of the various motion block are also indicated.

You may, of course customize the parameter bar to suit your requirements.

Program structure

This sample program utilizes a significant feature of the EMC blocks, namely the automatic coordination between the blocks: the start of a new run automatically overrides a current run, with the user only having to issue a new start signal.

The start signals of the frame FB are interconnected directly with the corresponding parameters of the EMC blocks. The start signal is reset at the end of a run (i.e. the block has terminated the motion or was overridden). The block is thus in a position to recognize and execute a new run job.

In the program, the control and feedback information is exchanged between the EMC blocks and the axis data on the one side and the interface of the frame FB on the other.

Error handling

Axis errors are displayed at the output bar of the multiple instance FB. This is always displayed as group error ("DB_Example_Multi.AxisError" = axis error) and in addition as block-relevant error ("DB_Example_Multi.MCXxxErr" = motion block error).

Easy Motion Control records the precise cause of an error in the error bits of the axis data. These error bits are not output to external applications in order to avoid unnecessary load on the interface of the frame FB. To view these errors, you must access the axis data in the instance DB directly, for example by means of the supplied VAT's.

You can determine the **type** of error by evaluating the error bits in the axis data. The output parameter "DB_Example_Multi.MCXxxErr" shows at which module the error was detected. ("Xxx" is representative for one of the blocks.)

The error, and hence the start signal of the block, will be retained until it is rectified and acknowledged by setting "DB_Example_Multi.ErrorAck". Provided the acknowledgement can be accepted, the EMC resets the error "DB_Example_Multi.AxisError" as well as the acknowledgement "DB_Example_Multi.ErrorAck". The negative edge of this acknowledgement is recognized and leads to a reset of the start signal. This in turn initiates a reset of the error bits in the motion blocks during the next cycle.

You can view detailed information via the configuration software.

Parameter assignment errors indicate faulty axis data; FC MC_Init detects these during the initialization and you cannot acknowledge them. Instead, you need to restart initialization (call of FC MC_Init) after the error is rectified.

Operator control

Open "VAT_Example_Multi", log on to the CPU and monitor the variables.

Enable the drives:
You must first enable the drives. To do so, set the "DB_Example_Multi.DriveEnable" bit and enable the control value.

Error acknowledgement:
After you have enabled the drives, you need to acknowledge "DB_Example_Multi.Achsdaten.Ax.Err.EnableDriveErr". To do so, set the "DB_Example_Multi.Achsdaten.Ax.ErrorAck" bit and enable the control value.

Jog mode:
Set and enable the desired direction bit "DB_Example_Multi.JogPos" or "DB_Example_Multi.JogNeg".

To stop the motion, reset the direction bit to "0".

Reference setting:
Prerequisite:
The axis is at a standstill.
To assign a coordinate value to the current axis position, enter this value at "DB_Example_Multi.PosOrDist". For setting the reference point, the velocity is set permanently to 0.0 in the program.
The function is executed by setting "DB_Example_Multi.ExeHome": the current process variable "DB_Example_Multi.ActPosition" accepts the reference point coordinates and the feedback "DB_Example_Multi.AxisSync" is set.

If you require a "reference point approach" instead of a "reference setting", you need to configure the corresponding velocity and direction at MC_Home. You can either do this in your program directly at the MC_Home by setting a velocity unequal to 0.0, or program it via the (expandable) input bar of the multiple instance FB and with a corresponding internal interconnection.

Absolute positioning:
In "DB_Example_Multi.PosOrDist", select the target position and then set "DB_Example_Multi.ExeAbsolute" to start the motion: the axis accelerates, moves at a constant speed and then decelerates to come to a standstill at the target position (monitor "DB_Example_Multi.ActPosition" while this motion is performed). When the axis reaches the target, you can see that "DB_Example_Multi.ExeAbsolute" is reset again to "0"; this has been implemented in the frame FB, so that the FB MC_MoveAbsolute is able to recognize the edge that triggers the next start.

Relative positioning:
In "DB_Example_Multi.PosOrDist", select the travel distance to be covered and then set "DB_Example_Multi.ExeRelativ" to start the motion: the axis accelerates, moves at a constant speed and then decelerates to come to a standstill at the target position (monitor "DB_Example_Multi.ActPosition" while this motion is performed).

To cancel positioning:
To abort a motion before the axis has reached the target, set the "DB_Example_Multi.ExeStop" bit. The axis decelerates according to the ramp set at the "DB_Example_Multi.Deceleration" parameter.

Error reaction:
The axis stops if an error occurs during the execution of a motion block.

A Appendix

A.1 Technical Data

A.1.1 Run Times

Travel FBs:

Minimal = no load call

Maximal = lead time of cycle at start of travel

Typical = lead time during travel

Other blocks:

Typical = lead time in normal mode

Block name		**CPU 416-2 6ES7416-2XK02-0AB0 [µs]**			**CPU 315-2 DP 6ES7315-2AF03-0AB0 [µs]**		
		Minimum	Maximum	Typical	Minimum	Maximum	Typical
MC_Init	FC 0			53			2203
MC_MoveAbsolute	FB 1	10	132	67	185	4525	2138
MC_MoveRelative	FB 2	10	93	67	183	3305	2143
MC_MoveJog	FB 3	10	88	48	184	2999	1387
MC_Home	FB 4	8	49	49	159	1796	1332
MC_StopMotion	FB 5	10	42	23	184	1464	696
MC_GearIn	FB 41	11	109	66	205	3832	2130
MC_Control	FB 11			27			819
MC_Simulation	FB 12			23			584
Input driver	FB 21...FB 29			50			1323
Output driver	FB 31...FB 39			20			413

Block name		CPU 314C-2 DP 6ES7314-6CF00-0AB0 [µs]			WinLC RTX V3.1 auf AMD with 1333 MHz [µs]		
		Minimum	Maximum	Typical	Minimum	Maximum	Typical
MC_Init	FC 0			967			21
MC_MoveAbsolute	FB 1	94	1969	908	3	42	18
MC_MoveRelative	FB 2	90	1430	911	3	31	18
MC_MoveJog	FB 3	93	1341	605	3	32	15
MC_Home	FB 4	77	797	592	3	20	15
MC_StopMotion	FB 5	94	649	309	3	14	7
MC_GearIn	FB 41	111	1671	931	4	39	21
MC_Control	FB 11			376			11
MC_Simulation	FB 12			259			6
Input driver	FB 21...FB 29			662			44
Output driver	FB 31...FB 39			223			31

A.1.2 Working Memory Assignment

		FC/FB					Instance data	
No.	**Block name**	**Version**	**Load memory required [byte]**	**Work memory required [Byte]**	**Local data [Byte]**	**System functions called**	**Load memory required [Byte]**	**Work memory required [Byte]**
FC 0	MC_Init	2.0	1482	1086	20		-	-
FB 1	MC_MoveAbsolute	2.0	4610	3924	94		590	112
FB 2	MC_MoveRelative	2.0	3586	2982	70		574	110
FB 3	MC_MoveJog	2.0	3706	3110	54		568	110
FB 4	MC_Home	2.0	3480	2886	54		558	104
FB 5	MC_StopMotion	2.0	1574	1114	8		470	70
FB 11	MC_Control	2.0	2234	1756	14		450	58
FB 12	MC_Simulation	2.0	824	410	12		460	64
FB 21	EncoderIM178-4	2.0	2774	2210	40		578	100
FB 22	EncoderFM450	2.0	3046	2334	44	SFC59 (RD_REC)	720	108
FB 23	EncoderET200S1SSI	2.0	2102	1566	78		556	76
FB 24	EncoderAbsSensorDP	2.0	1916	1420	38		532	88
FB 25	EncoderSM338	2.0	2718	2040	88	SFC59 (RD_REC)	696	98

		FC/FB					Instance data	
No.	**Block name**	**Version**	**Load memory required [byte]**	**Work memory required [Byte]**	**Local data [Byte]**	**System functions called**	**Load memory required [Byte]**	**Work memory required [Byte]**
FB 26	EncoderET200S 1COUNT	2.0	3114	2454	68		628	100
FB 27	EncoderFM350	2.0	3422	2654	76	SFC59 (RD_REC)	756	110
FB 28	EncoderCPU314C		2034	1476	62	SFB47 (COUNT)	638	128
FB 29	EncoderUniversal	2.0	1890	1416	36		494	78
FB 31	OutputIM178-4	2.0	856	436	12		454	54
FB 32	OutputSM432	2.0	758	352	10		442	52
FB 33	OutputET200S2AOU	2.0	950	498	52		486	54
FB 34	OutputCPU314C	2.0	764	356	12		444	52
FB 35	OutputSM332	2.0	950	498	52		486	54
FB 36	OutputUniversal	2.0	798	384	8		460	64
FB 37	OutputMM4_DP	2.0	1740	1242	12		478	68
FB 41	MC_GearIn	2.0	4442	3476	78		944	128
UDT1		2.0	-	-	-		830	318

A.2 Axis DB

Note

You may not alter any data not listed in this table.

Table B-1 Contents of Axis DB

Add.	Name	Type	Initial Value	Comment
0.0	Sample_T	REAL	0.01	Scan time of axis FBs [s]

4.0	AxisType	BOOL	FALSE	Axis type
				FALSE = Linear axis
				TRUE = Rotary axis

4.1	EncoderType	BOOL	FALSE	Encoder type
				FALSE = Incremental encoder
				TRUE = Absolute encoder

4.2	Sim	BOOL	FALSE	TRUE = Simulation mode is active
6.0	AxisLimitMax	REAL	$1.0e^{+6}$	Software limit switch end [unit of length]
10.0	AxisLimitMin	REAL	$-1.0e^{+6}$	Software limit switch start [unit of length]
14.0	MaxVelocity	REAL	0.1	Maximum axis velocity [unit of length/s]
18.0	MaxAcceleration	REAL	0.1	Max. axis acceleration [unit of length/ s^2]
22.0	MaxDeceleration	REAL	0.1	Max. axis deceleration [unit of length/ s^2]
26.0	MonTimeTargetAppr	REAL	1.0	Monitoring time for target approach [s]
30.0	TargetRange	REAL	1.0	Target range on positioning [unit of length]
34.0	StandstillRange	REAL	1.5	Standstill range without travel order [unit of length]
38.0	MaxFollowingDist	REAL	5.0	Maximum admissible following distance [unit of length]
42.0	MonitorTargetAppr	BOOL	TRUE	TRUE = engage monitoring of target range
42.1	SWLimitEnable	BOOL	TRUE	TRUE = engage monitoring of software limit switch
44.0	FactorP	REAL	1.0	Proportionality factor [1/s]
48.0	ManVelocity	REAL	0.0	Setpoint velocity in manual mode [unit of length/s]
52.0	ManEnable	BOOL	FALSE	TRUE = engage manual mode
54.0	EmergencyDec	REAL	1000.0	Axis deceleration for hard stop [unit of length/ s^2]
58.0	InputModuleInAddr	INT	0	Initial address of inputs of distance measurement module
60.0	InputModuleOutAddr	INT	0	Initial address of outputs of distance measurement module
62.0	InputChannelNo	INT	0	Channel number of distance measurement module

Add.	Name	Type	Initial Value	Comment
64.0	StepsPerRev	DINT	L#1	Steps per encoder revolution [steps/encoder revolution]
68.0	DisplacementPerRev	REAL	1.0	Distance per encoder revolution [unit of length/encoder revolution]
72.0	NumberRevs	INT	1	Number of encoder revolutions of an absolute encoder
74.0	PolarityEncoder	INT	1	Set encoder polarity
80.0	OutputModuleOutAddr	INT	0	Initial address of output module outputs
82.0	OutputModuleInAddr	INT	0	Initial address of output module inputs
84.0	OutputChannelNo	INT	0	Channel number of output module
86.0	PolarityDrive	INT	1	Set drive polarity
88.0	DriveInputAtMaxVel	REAL	1.0	Reference value for max. axis velocity [V]
92.0	OffsetCompensation	REAL	0.0	Offset compensation [V]
96.0	DriveInputAt100	REAL	10.0	Reference value for 100 % speed [V]
100.0	Override	REAL	100.0	Velocity override [%]
104.0	ActPosition	REAL	0.0	Actual position [unit of length]
108.0	FollowingDistance	REAL	0.0	Current following distance [unit of length]
112.0	RemainingDistance	REAL	0.0	Current residual distance to target [unit of length]
116.0	NomVelocity	REAL	0.0	Setpoint velocity of axis [unit of length/s]
120.0	ActVelocity	REAL	0.0	Actual speed of axis [unit of length/s]

124.0	Sync	BOOL	FALSE	FALSE = axis not synchronized
				TRUE = axis synchronized

124.1	Error	BOOL	FALSE	FALSE = no error
				TRUE = group error

124.2	ErrorAck	BOOL	FALSE	FALSE = no acknowledgement
				TRUE = acknowledge error

128.0	Err.SWLimitMinExceeded	BOOL	FALSE	TRUE = software limit switch start exceeded
128.1	Err.SWLimitMaxExceeded	BOOL	FALSE	TRUE = software limit switch end exceeded
128.2	Err.TargetErr	BOOL	FALSE	TRUE = target outside travel range
128.3	Err.NoSync	BOOL	FALSE	TRUE = axis not synchronized
128.4	Err.DirectionErr	BOOL	FALSE	TRUE = travel in specified direction inadmissible
128.5	Err.DataErr	BOOL	FALSE	TRUE = parameter of one travel FBs inadmissible
128.6	Err.StartErr	BOOL	FALSE	TRUE = start not possible in current axis status
128.7	Err.DistanceErr	BOOL	FALSE	TRUE = Overtravel
129.0	MasterErr	BOOL	FALSE	TRUE = Master axis in error state
130.0	Err.StoppedMotion	BOOL	FALSE	TRUE = axis in stop status requiring acknowledgement
130.1	Err.EnableDriveErr	BOOL	FALSE	TRUE = drive enable missing
130.2	Err.FollowingDistErr	BOOL	FALSE	TRUE = max. following distance exceeded
130.3	Err.StandstillErr	BOOL	FALSE	TRUE = standstill range exceeded

Add.	**Name**	**Type**	**Initial Value**	**Comment**
130.4	Err.TargetApproachErr	BOOL	FALSE	TRUE = error on target approach
130.5	Err.EncoderErr	BOOL	FALSE	TRUE = encoder error
130.6	Err.OutputErr	BOOL	FALSE	TRUE = error at output driver
130.7	Err.ConfigErr	BOOL	FALSE	TRUE = axis data incorrectly parameterized
131.0	ERR.DriveErr	BOOL	FALSE	TRUE = Drive error
132.0	Config.Err_AxisLimit	BOOL	FALSE	TRUE = axis limits incorrect
132.1	Config.Err_MaxVelocity	BOOL	FALSE	TRUE = max. velocity incorrect
132.2	Config.Err_Max Acceleration	BOOL	FALSE	TRUE = max. acceleration incorrect
132.3	Config.Err_Max Deceleration	BOOL	FALSE	TRUE = max. deceleration incorrect
132.4	Config.Err_MonTime TargetAppr	BOOL	FALSE	TRUE = monitoring time for target approach incorrect
132.5	Config.Err_TargetRange	BOOL	FALSE	TRUE = target range incorrect
132.6	Config.Err_Standstill Range	BOOL	FALSE	TRUE = standstill range incorrect
132.7	Config.Err_MaxFollowing Dist	BOOL	FALSE	TRUE = max. following distance incorrect
133.0	Config.Err_Emergency Dec	BOOL	FALSE	TRUE = deceleration for hard stop incorrect
133.1	Config.Err_StepsPerRev	BOOL	FALSE	TRUE = steps per encoder revolution incorrect
133.2	Config.Err_Displacement PerRev	BOOL	FALSE	TRUE = axis distance per encoder revolution incorrect
133.3	Config.Err_NumberRev	BOOL	FALSE	TRUE = number of encoder revolutions incorrect
133.4	Config.Err_Polarity Encoder	BOOL	FALSE	TRUE = set encoder polarity incorrect
133.5	Config.Err_PolarityDrive	BOOL	FALSE	TRUE = set drive polarity incorrect
133.6	Config.Err_VoltAtMax Velocity	BOOL	FALSE	TRUE =reference value for maximum velocity incorrect
133.7	Config.Err_AxisLength	BOOL	FALSE	TRUE = length of axis incorrect
134.0	Config.Err_Encoder Range	BOOL	FALSE	TRUE = encoder range is not suitable for axis length
134.1	Config.Err_MaxVelRotary Axis	BOOL	FALSE	TRUE = max. velocity and scan time do not match rotary axis length
134.2	Config.Err_DriveInputAt10 0	BOOL	FALSE	TRUE = Reference value for 100 % speed <= 0
136.0	EncoderValue	DINT	L#0	Current encoder value
278.0	Init	STRUCT	TRUE	Bit field for initializing 32 function blocks

A.3 Error Lists

A.3.1 Error Displays

Error with soft stop

Error with hard stop

Configuration error

A.3.2 JobErr Messages

JOB_ERR (Hex)	JOB_ERR (Dez)	JOB_ERR (Int)	Meaning
80A0	32928	-32608	Negative acknowledgement on reading from module. Module removed during reading process or module defective
80A1	32929	-32607	Negative acknowledgement on writing to module. Module removed during write process or module defective
80A2	32930	-32606	DP log error at layer 2
80A3	32931	-32605	DP log error at user interface/user
80A4	32932	-32604	Communication faulty at K-bus
80B0	32944	-32592	Data record/order unknown
80B1	32945	-32591	Length data incorrect. Parameter FM_TYPE in channel DB not properly set for the module used
80B2	32946	-32590	Configured slot is not assigned
80B3	32947	-32589	Actual module type not same as setpoint module type
80C0	32960	-32576	Module has not yet prepared the data to be read
80C1	32961	-32575	Data of a similar write order has not yet been processed on the module
80C2	32962	-32574	Module is currently processing the maximum possible number of orders
80C3	32963	-32573	Required resources (memory, etc.) are currently occupied
80C4	32964	-32572	Communication error
80C5	32965	-32571	Distributed I/O is not available
80C6	32966	-32570	Priority class abortion (restart or background)
8522	34082	-31454	Channel DB or parameter DB too short. Data cannot be read from the DB (write order)
8532	34098	-31438	DB number of parameter DB too high (write order)
853A	34106	-31430	Parameter DB not available (write order)
8544	34116	-31420	Error at n^{th} (n > 1) read access to a DB after an error has occurred (write order)
8723	34595	-30941	Channel DB or parameter DB too short. Data cannot be written into the DB (read order)

JOB_ERR (Hex)	**JOB_ERR (Dez)**	**JOB_ERR (Int)**	**Meaning**
80A0	32928	-32608	Negative acknowledgement on reading from module. Module removed during reading process or module defective
80A1	32929	-32607	Negative acknowledgement on writing to module. Module removed during write process or module defective
80A2	32930	-32606	DP log error at layer 2
80A3	32931	-32605	DP log error at user interface/user
80A4	32932	-32604	Communication faulty at K-bus
8730	34608	-30928	Parameter DB write-protected in the CPU Data cannot be written into the DB (read order)
8732	34610	-30926	DB number of parameter DB too high (read order)
873A	34618	-30918	Parameter DB not available (read order)
8745	34629	-30907	Error at nth (n > 1) write access to a DB after an error has occurred (read order)
Errors 80A2..80A4 and 80Cx are temporary, i.e. they can be rectified independently after a waiting time.			

Index

Siemens AG
Bereich Automation and Drives
Geschaeftsgebiet Industrial Automation Systems
Postfach 4848, D- 90327 Nuernberg

Siemens Aktiengesellschaft