

Lenze

Manual



Prepared Solution

FlyingSaw

9300 ServoPLC / ECSxA

Important note:

This software is supplied to the user as described in this document. All risks resulting from its quality or use remain the responsibility of the user. The user must provide all safety measures against possible incorrect operation.

We do not take any liability for direct or indirect damage, e.g. profit loss, order loss, or any loss regarding business.

© 2006 Lenze Drive Systems GmbH

No part of this documentation may be reproduced or made available to third parties without written consent from Lenze Drive Systems GmbH.

All indications given in this documentation have been carefully selected and comply with the hardware and software described. Nevertheless, deviations cannot be ruled out. We do not take any responsibility or liability for damage which might possibly occur. Required corrections will be made in the following editions.

All tradenames in this documentation are trademarks belonging to the related owners



1	Preface and general information.....	1-1
1.1	Version information.....	1-1
1.2	Scope of supply.....	1-2
1.3	About this manual.....	1-3
1.4	Range of functions and modes.....	1-4
1.4.1	Parameterisable variant.....	1-5
1.4.2	Programmable variant.....	1-7
1.4.3	"Flying Saw"Prepared Solution.....	1-9
1.5	Conventions used.....	1-9
2	Features of the Prepared Solution.....	2-1
2.1	Definition of a flying saw.....	2-2
2.2	Example layout and functional principle.....	2-3
2.2.1	Typical layout of a flying saw.....	2-3
2.3	Typical applications.....	2-4
2.3.1	Typical motion profiles.....	2-5
2.4	Synchronous synchronisation.....	2-8
2.5	Oversynchronous synchronisation.....	2-9
2.6	Length-controlled operation.....	2-10
2.7	Mark-controlled operation.....	2-11
2.8	"Start gap" function.....	2-12
2.9	Applicability criteria for the parameterisable Prepared Solution.....	2-13
2.10	Functions of the control inputs and outputs.....	2-18
2.11	Manual control.....	2-18
2.12	Start gap.....	2-18
2.13	Cut done.....	2-18
2.14	Immediate cut / top cut.....	2-18
2.15	Homing.....	2-19
2.16	Manual control.....	2-22
2.16.1	Limit switch integration.....	2-23
2.16.2	Software limit positions.....	2-23
2.17	Moving clear from a limit switch.....	2-25
2.18	Delay on the synchronised signal.....	2-26
2.19	Status signal "Double length".....	2-27
2.20	Status signal "Mark not detected".....	2-27
2.21	Following error monitoring.....	2-28
3	Functions.....	3-29
3.1	Cut counter.....	3-29
3.2	Scrap counter.....	3-29
3.3	Decrementing cut counter.....	3-29
3.4	Top cut counter.....	3-29
3.5	Resetting the counters.....	3-29
3.6	Automatic operation: length-controlled operation.....	3-30
3.7	Automatic operation: mark-controlled operation.....	3-30
3.7.1	Mark checking.....	3-30
3.7.2	Simulation of the master speed.....	3-30
3.8	Measuring material speed.....	3-31
3.8.1	Measuring wheel and encoder selection.....	3-32
	Control/status interface to the higher-level control.....	3-34
4	Commissioning the "(Flying Saw)".....	4-43
4.1	Requirements.....	4-43
4.1.1	Required components.....	4-43
4.2	Hardware layout.....	4-44
4.2.1	Wiring of the ServoPLC control terminals.....	4-45
4.2.2	Wiring of the ECSxA control terminals.....	4-46
4.3	Commissioning of the Prepared Solution (parameterisable variant).....	4-47
4.3.1	Download of the program FlyingSaw_SPLC_Vxxxxxx / FlyingSaw_ECS_Vxxxxxx.....	4-47
4.3.2	Sequence for online commissioning using GDC.....	4-53



4.3.3	Sequence for the online commissioning of the functions	4-60
5	Commissioning length-controlled operation	5-1
6	Commissioning mark-controlled operation	6-1
7	State machine of the Prepared Solution	7-1
7.1	Overview	7-1
7.1.1	Concise description of the states	7-2
7.2	Parallel functions	7-3
8	Program extensions/supplements	8-1
8.1	Configuration of the ServoPLC user interface	8-1
8.1.1	Default setting of the ServoPLC hardware inputs	8-1
8.1.2	Default setting of the ServoPLC hardware outputs	8-2
8.2	Configuration of the ECS user interface	8-3
8.2.1	Default setting of the ECS hardware inputs	8-3
8.2.2	Default setting of the ECS hardware outputs	8-4
8.3	Task management	8-5
9	Dimensioning aspects	9-1
9.1	Resolution of the system	9-1
9.2	Axis normalisation	9-2
9.3	Master frequency source measuring wheel	9-2
9.4	Master frequency source Servo / Servo PLC	9-4
10	Description of the function blocks	10-1
10.1	Function block <i>MotionControl</i>	10-1
10.2	Function block <i>Software_Limit</i>	10-4
10.3	Function block <i>RatioNormFlyingSaw</i>	10-5
10.4	Function block <i>Master Frequency</i>	10-6
10.5	Function block <i>LengthCalculation</i>	10-8
10.6	Function block <i>Offset Calculation</i>	10-10
10.7	Function block <i>Synchronize Control</i>	10-12
10.8	Function block <i>VersionHandling</i>	10-15
10.9	Function block <i>MultiplexerInput</i>	10-17
10.10	Function block <i>MultiplexerOutput</i>	10-21
11	Appendix	11-1
11.1	Possible error sources	11-1
11.1.1	Slip at the measuring wheel or at the material infeed	11-1
11.1.2	Interference on the master encoder signal	11-1
11.1.3	Incorrectly set synchronisation ratio / normalisation factor	11-1
11.2	Global variables	11-2
11.2.1	Global	11-2
11.2.2	VarCounter_FS	11-3
11.2.3	VarErrorHandling	11-3
11.2.4	VarInterfaceFlyingSaw	11-4
11.2.5	VarLimitsSwitches	11-5
11.2.6	VarNormFactor	11-5
11.2.7	VarOperationVisu	11-5
11.2.8	VarStatusMachine	11-5
11.2.9	VarVersion	11-6
11.3	Codes of the Prepared Solution	11-7
11.3.1	Table of application codes	11-7
11.3.2	Code initialisation values	11-11
11.4	Error messages	11-12
11.4.1	System error messages	11-13
11.4.2	Application error messages	11-19
11.4.3	User-defined error messages	11-20



1 Preface and general information

1.1 Version information

This document is valid for the Prepared Solution "FlyingSaw" in version V1.x

Version	ID number	Modifications
1.0 07/2007		New document
1.1 03/2008		Reworked document with target ECSxA

The versions for the project file and the application library for the Prepared Solution are displayed in the following format using codes C3999/001 and C3999/003:

Code	Possible settings:		Comment
	Default	Selection	
C3999	-	0 ... {1} ... 99.99	Display code: version for the Prepared Solution (subcode 1: project file, subcode 2: application library 1, subcode 3: application library 2): ▪ The numerals one and two define the main version ▪ The numerals three and four define the subversion ▪ The numerals five and six define the service-pack
1			
2			
3			



1.2 Scope of supply

The CD-ROM enclosed contains the following files:

File type	Use
*.BIN	Binary file:
FlyingSaw_SPLC_Vxxxxxx.bin FlyingSaw_ECS_Vxxxxxx.bin	The binary file contains the compiled project with all system and application codes (for information on application codes see chapter 11.3.1) and can be transferred to the target system using the Lenze software tool GDLoader. The binary file is required if the parameterisable variant (see chapter 1.4.1) is to be used.
*.LPC	Template model:
FlyingSaw_SPLC_Vxxxxxx.lpc FlyingSaw_ECS_Vxxxxxx.lpc	The template model contains the source code for the Prepared Solution and is required if the programmable variant is to be used (see chapter 1.4.2). It can be edited, translated and transferred to the target system using the Lenze software DDS.
*.LIB	Library file:
FlyingSaw_SPLC_Vxxxxxx.lib FlyingSaw_ECS_Vxxxxxx.lib	The library file contains the core functionality for the Prepared Solution and is the basis for the project file. Without the library file the project file cannot be completely translated in DDS.
*.PDB	Device description file:
FlyingSaw_SPLC_Vxxxxxx.pdb FlyingSaw_ECS_Vxxxxxx.pdb	The device description file is required for setting the parameters for the Prepared Solution (parameterisable version) using the Lenze software tool GDC. Before GDC is used, the device description file must be copied to the corresponding directory for the GDC software).
*.SDB	Symbol file for GDOscilloscope:
FlyingSaw_SPLC_Vxxxxxx.sdb FlyingSaw_ECS_Vxxxxxx.pdb	The symbol file for use in connection with the Lenze software Global Drive Oscilloscope contains a list of signals in the Prepared Solution that can be displayed using the oscilloscope. When the GDOscilloscope is started this file must be assigned to be able to access the required signals from the Prepared Solution.
*.PDF	PDF file (manual):
FlyingSaw_vx-x_DE.pdf FlyingSaw_vx-x_EN.pdf	This manual describes the Prepared Solution in detail.

The files are installed automatically by the setup provided on the product CD.



1.3 About this manual

Prepared Solutions from Lenze make it easier for the user to implement complex drive functions in specific practical situations. Prepared Solutions already contain the complete functionality for a machine (core functionality as well as peripheral functionality) and can be commissioned rapidly using only a few steps.

This manual on the one hand describes the basics of the application covered by the Prepared Solution. In this way the user is familiarised with the physical aspects of the system and obtains the necessary basic knowledge for commissioning and modifying the Prepared Solution to the specific application.

On the other hand technical details (e.g. a list of the necessary components, application parameters and application variables, step-by-step commissioning instructions, diagnostics and fault elimination) are addressed to ensure the Prepared Solution can be used quickly.



1.4 Range of functions and modes

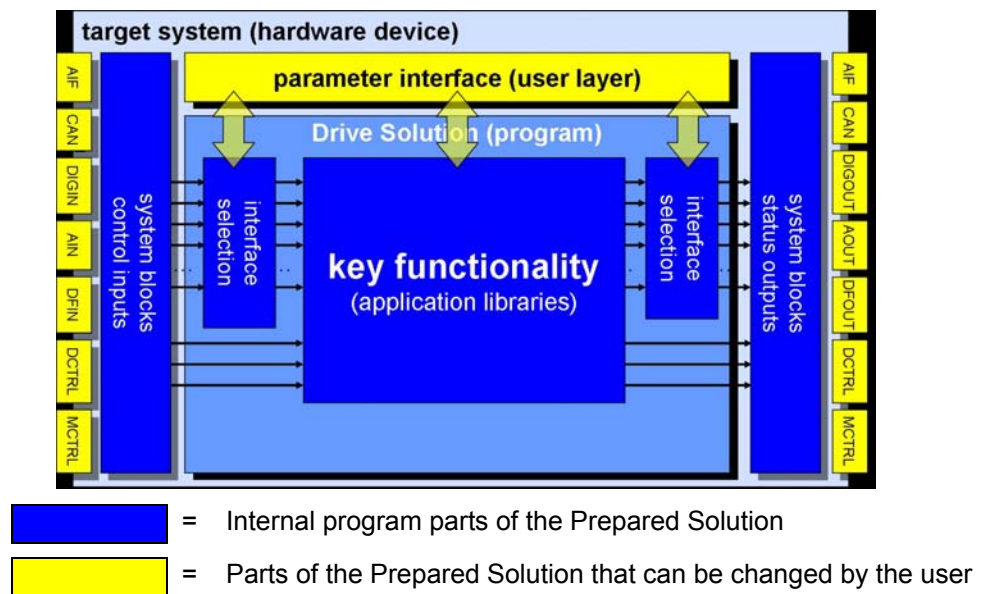
The Prepared Solutions essentially comprise a number of files that can be employed by the user as a basis for the specific application. For this purpose two different versions of the Prepared Solution are available: in the parameterisable version the Prepared Solution provides very quick commissioning with pre-defined interfaces and functionality. The programmable version is suitable particularly for flexible modification to special customer requirements. A corresponding control and status interface are available as a function, depending on whether the user chooses the parameterisable or the programmable variant.

	<i>Parameterisable variant</i>	<i>Programmable variant</i>
Predefined interfaces	✓	—
Free definition of the user interface	—	✓
Very quick commissioning by setting parameters	✓	—
Modifications/extensions to the functionality using custom sub-routines, programmed in the IEC programming languages	—	✓



1.4.1 Parameterisable variant

With the parameterisable variant all basic functions and the selection of the interface signals are defined solely by the use of parameters (codes). As a result a uniform, predefined interface is provided to the user. Typical signal wires, terminal assignments, interfaces to data bus systems, etc. can be selected via parameters (codes). Therefore the parameterisable variant of the Prepared Solution provides the quickest commissioning without restricting the functionality of the Prepared Solution in any way:



The system is configured entirely using codes (parameter interface). The function of the individual codes for setting parameters is listed in chapter 11.3.1.



When the parameterisable variant is used, the following Lenze software tools are required:



Global Drive Control (GDC):

For operation, parameter setting, and diagnostics



Global Drive Loader (GDLoader):

For transferring the binary file of the project and the application data (profile data)¹



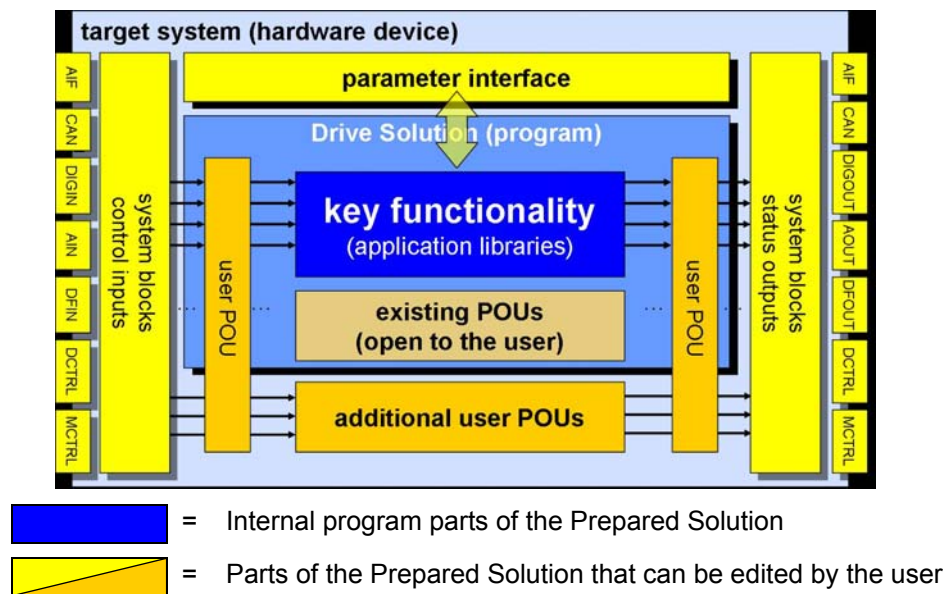
Global Drive Oscilloscope (GDO):

For diagnostics and recording of temporal characteristics



1.4.2 Programmable variant

The programmable variant includes the range of functions of the parameterisable variant, however, the user can modify the program for the Prepared Solution using IEC61131 modifications (e.g. for the definition of interfaces, modification of the external signal flow, ...) to add other, custom sub-routines. In this way the Prepared Solution can be adapted to the customer's requirements. It is also possible to optimally utilise the target platform's resources without affecting the core functionality.



The configuration of the signal flow beyond the core functionality in the case of the programmable variant is left to the user, as is the changing/modification/extension of existing sub-routines in/to the Prepared Solution (e.g. error handling). The interface between the sub-routines that contain the core functionality and the application sub-routines as well as the system blocks is similar to the templates for the software packages and uses global variables. The significance of the global interface variables is listed in chapter 11.2.



TIP!

The core functionality of the Prepared Solution cannot be changed by the user in either variant. In this way it is ensured that unintentional malfunctions cannot be programmed or monitoring is disabled.



When the programmable variant is used, the following Lenze software tools are required:



Drive PLC Developer Studio (DDS):

For the adaptation of the Prepared Solution program (editing the PRO file)



Global Drive Control (GDC):

For operation, parameter setting, and diagnostics



Global Drive Loader (GDLoader):

For transferring the binary file for the project and the application data (profile data)², if the customer project once produced is to be duplicated.



Global Drive Oscilloscope (GDO):

For diagnostics, optimisation and the recording of temporal characteristics



1.4.3 "Flying Saw" Prepared Solution

The "Flying saw" Prepared solution provides the software-based solution of a drive task. Flying saws are used in many production processes. Specifically whenever material is to be machined during a production process is running. This process does not necessarily has to be a sawing process. Filling or drilling processes, for instance, are also possible. A primary characteristic of a flying saw is the length or mark-controlled synchronisation to a production speed.

1.5 Conventions used

This manual uses the following conventions to distinguish between the different types of information:

Type of information	Representation	Example
Names of dialog boxes, input fields and selection lists	<i>Italic</i>	The <i>Options</i> dialog box
Buttons	Bold	Click OK to ...
Menu commands	Bold	Using the Messages command you can... If a function requires several commands to be carried out in succession, the individual commands are separated from each other by an arrow: select File → Open to...
Keyboard commands	<Bold>	You can open the input help using <F2>. If a command requires a key combination, a "+" is placed between the commands: using <Shift> + <ESC> you can...
Program listing	Courier	IF var1 < var2 THEN...
Keywords	Courier bold	...starts with FUNCTION and ends with END FUNCTION .
Important note		Attention! Do not use the Online → Controller inhibit command for an emergency stop via the PC, as this command only arrives at the controller with a delay.
Tip		TIP! If you keep the mouse pointer over a symbol on the toolbar for a short time, the corresponding command is indicated in a "tooltip".

Variable names

The conventions used by Lenze for the variable names of its system blocks, function blocks and functions are based on the so-called Hungarian Notation. This notation makes it possible to identify the most important properties (e.g. the data type) of the corresponding variable by means of its name, e.g. DIGIN_bIn1_b.

You will find information about the conventions in the appendix of the DDS Online documentation "Introduction to IEC 61131-3 programming".



2 Features of the Prepared Solution

In the following, the key features and functions of the flying saw are given as bullet points.

The Prepared Solution includes the following functions:

- Homing, determining the reference position of the axis
- Inching mode, manual positioning of the axis
- Absolute positioning
- Top cut at "zero" line speed
- Top cuts independent of the position (away from the initial position)
- Length counter for the total running metres
- Cutting length control (length calculator)
- Cutting mark control (touch probe, mark)
(with monitoring of the "forced cut" mark detection)
- Top cut counter for the number of top cuts carried out
- Cut counter for the total number of pieces
- Scrap counter (externally triggered)
- "Synchronous" synchronisation to a master speed
- "Oversynchronous" synchronisation to a master speed
- Simulation of the material line using adjustable, virtual line speed
- Correction value definition for the straightforward compensation of slip at the measuring wheel
- Error handling (system and application-specific error messages)
- Limit switch monitoring
- Software limit switch monitoring



2.1 Definition of a flying saw

A flying saw is a slave axis which is synchronised to a moving master axis, e.g. a wood conveyor. In the simplest case, the slave axis travels in parallel with the master axis (parallel slave), but it can also travel diagonally to the master axis (diagonal slave axis).

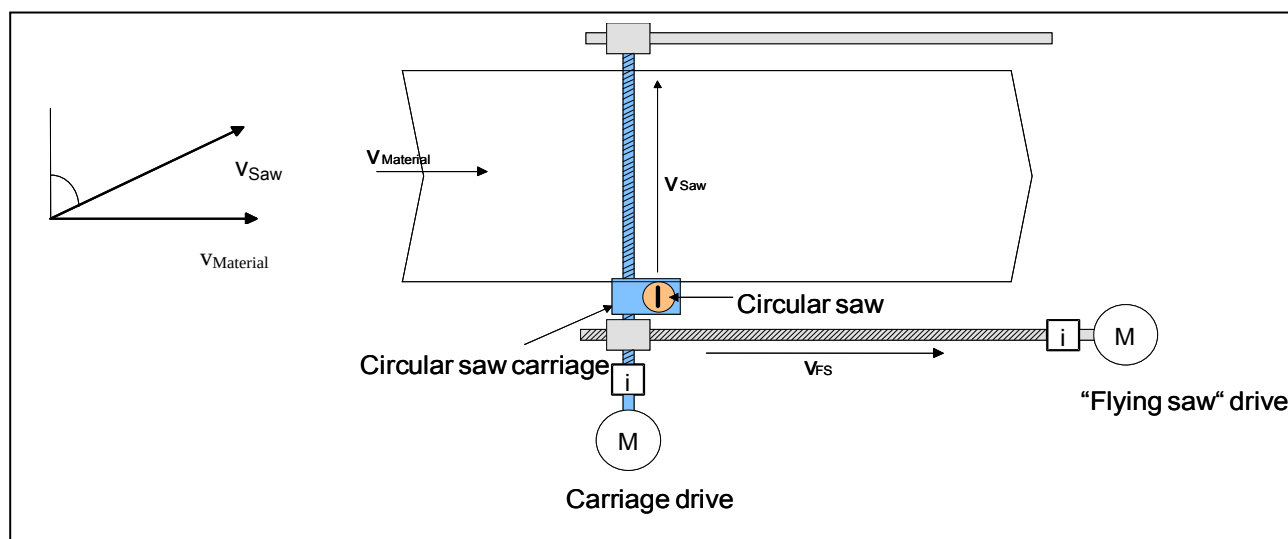
Parallel slave

A parallel slave is an axis which is synchronised from standstill to a master axis travelling with the constant speed v_m . The synchronous speed v_{FS} is reached at a previously determined position (master slave position, cutting position).

Diagonal slave axis

Just like a parallel slave axis, a diagonal slave axis is synchronised to a moving master axis. However, the slave axis moves diagonally to the master axis with an angle of $0^\circ < \alpha \leq 90^\circ$. This means that the diagonal slave is faster than a parallel slave once it has reached the synchronous position. The speed can be calculated depending on the angle: $v_{FS} = v_m / \sin(\alpha)$. In the case of a parallel slave, the angle is $\alpha = 90^\circ$.

In the illustration below, the difference between a "parallel flying saw" and a "diagonal flying saw" is shown. The relationship between the speeds can be seen in the vector diagram. If the cutting angle is $\alpha=90^\circ$, the vectors are on top of each other. In this case, the speeds are equal, i.e. the synchronous speed of the flying saw corresponds to the master speed of the material path. This Prepared Solution only includes the "parallel flying saw".



Top view of a possible process

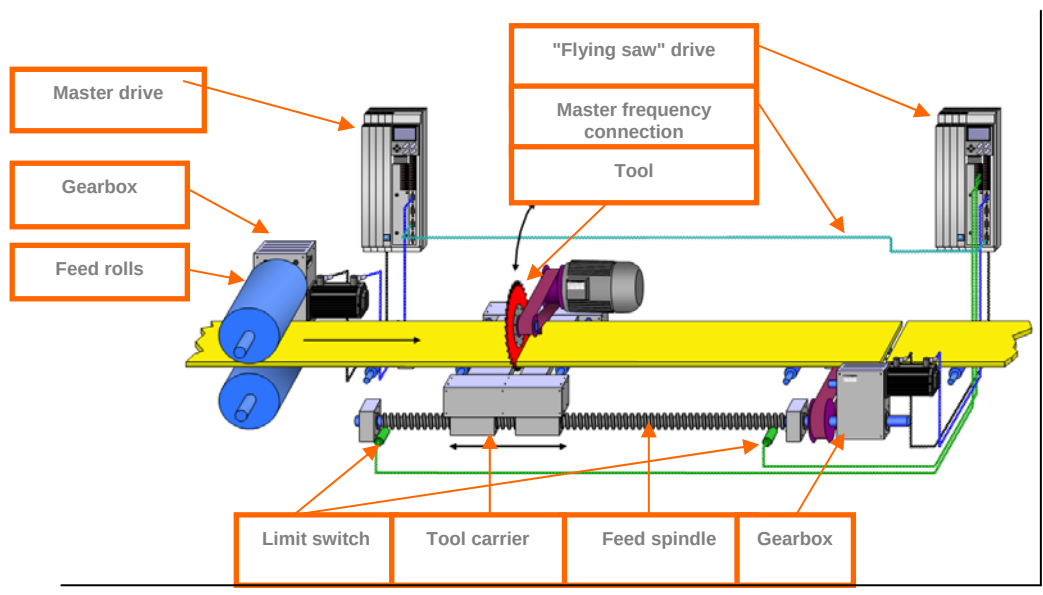


2.2 Example layout and functional principle

Flying saws are used in many production processes. Specifically whenever material is to be machined during a production process is running. This process does not necessarily have to be a sawing process. Filling or drilling processes, for instance, are also possible. A primary characteristic of a flying saw is the length or mark-controlled synchronisation to a production speed.

Using the length calculator you can machine the continuous material at the lengths set during feed, e.g. sawing, filling, punching, drilling, etc. If your material has defined marks used as a reference for material machining, you can use mark control.

2.2.1 Typical layout of a flying saw



The overall installation always includes a feeding process, in this case a feed roll, a system for detecting material speed and, if necessary, a transport process for the processed material after the flying saw.



2.3 Typical applications

The flying saw is always used if continuous material feed does not permit processing at standstill.

Task	
Filling moving containers	
Processing, gripping and checking of moving workpieces	
Spraying paint / painting moving workpieces	
Cutting to length / separating a moving web	
Printing / embossing / marking moving workpieces	

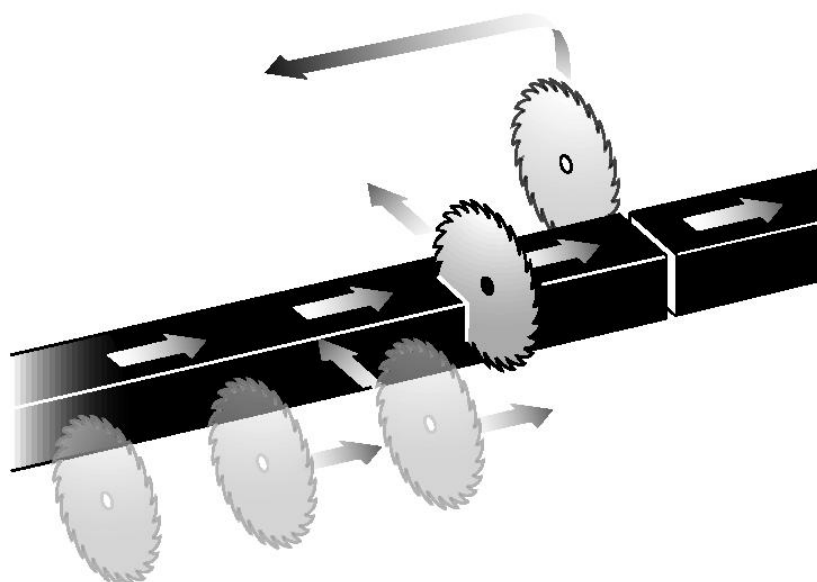


2.3.1 Typical motion profiles

Flying saws are used to cut continuous material to length if the material cannot be stopped during the cutting process.

The mechanical design includes a saw carriage that moves in the direction of the material. The carriage moves in synchronism with the material during the cutting process and returns to its initial position once the cut has been made.

The advantage is an improved number of cycles and a better use of resources in a production process. Therefore it is necessary to synchronise tools and workpieces with reference to their position and speed such that the tool can be used in the same way as if the workpiece were stationary. An example for such an application is a cross-saw which cuts across the material (at an angle of 90° to the workpiece) while it is transported.



The picture above shows a "Flying saw" application. Continuous material such as extruded material has to be cut to length. As soon as the cutting process starts, the saw is synchronised to the speed of the material to be cut to length. During the cutting process, the saw runs in synchronism. At the end of the process, the saw moves back to its initial position.



The start signal for synchronising can be generated in two ways:

- **Cutting mark control:**

A sensor, e.g. a photoelectric barrier, registers the cutting marks present on the material. This sensor signal is processed as an interrupt in the inverter and starts the sawing process. This method is used if there are cutting marks on the material which have to be referred to, e.g. when using printed materials.

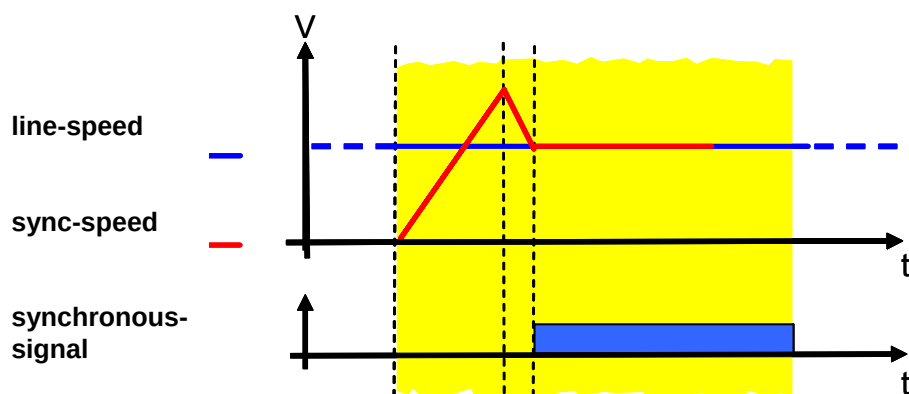
- **Cutting length control:**

An encoder on the material registers the material speed of the production process. This information is processed by the controller. Cutting marks on the material are not required. The cutting length control provides equidistant cutting lengths. The advantage of cutting length control is that no cutting marks are required on the material. A length calculator calculates equidistant lengths in the controller and generates a start signal for the synchronising process.

The synchronising process either started by cutting mark or cutting length control can be synchronous or oversynchronous:

- **Oversynchronous:**

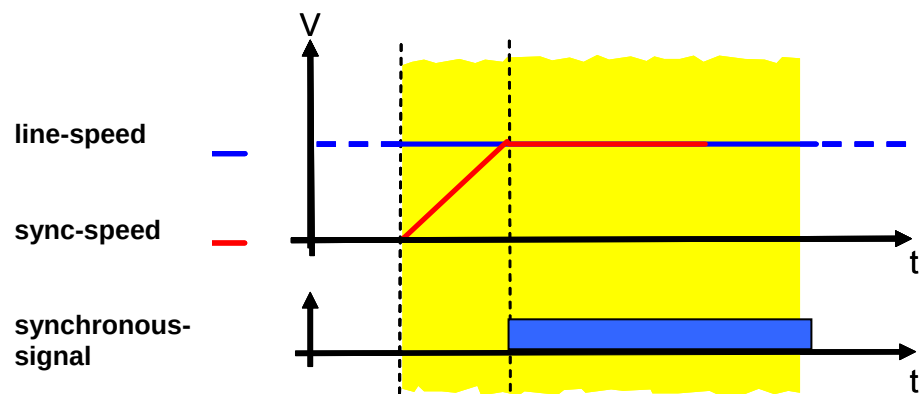
The "Flying saw" moves faster than the material/belt during the synchronising process. It starts exactly at the cutting position and catches up with all increments made during the synchronising process. Then the saw axis has achieved angular and rotational speed synchronism and switches the *synchronised signal*. Then the machining process can be started.





- **Synchronous:**

The "Flying saw" never moves faster than the material. For this purpose an offset is calculated online and the "Flying saw" therefore starts before the cutting position reaches the saw. The advantage this mode offers for synchronisation is primarily that maximum process speeds for the synchronous axis cannot be exceeded. This situation could be the case when the master speed for the material to be machined is already very high and the synchronous axis would hit the maximum speed limit trying to catch up with the cutting point.



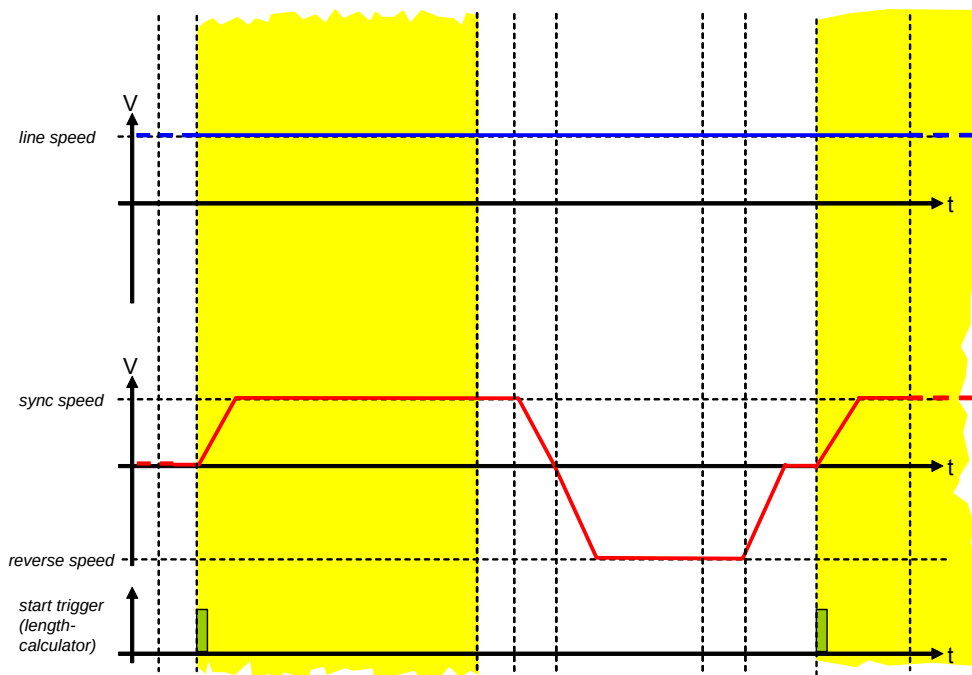
Comment: The time delay between reaching the synchronous speed and the output of the synchronised signal can be set using a parameter in the Prepared Solution.



2.4 Synchronous synchronisation

The saw axis is started using the start signal from the length calculator or a mark on the material and synchronised as per entries for the line speed with speed and angular synchronism.

The following diagram shows a synchronising process in detail.



The trigger for the saw axis is the start signal (positive edge), in the above example triggered by the internal length calculator. The saw axis then starts and is in synchronism on reaching the line speed. An "oversynchronous" movement is not necessary, as the saw axis is started earlier using a calculated offset.

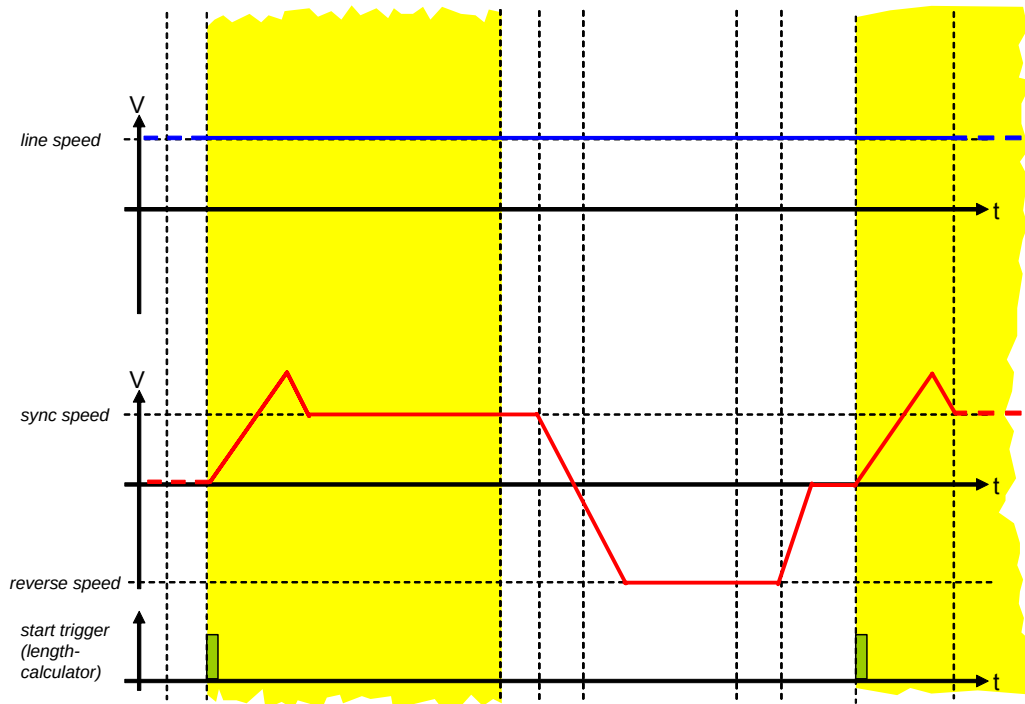
The same applies to starting using a mark on the material.

Switching between synchronous and oversynchronous operation is performed via the application control word using the *Synchronisation mode* bit (default bit 05). Here setting the bit means that the axis is synchronised synchronously. If the bit is not set the axis moves oversynchronously during the synchronisation movement.



2.5 Oversynchronous synchronisation

The saw axis synchronises on a start signal from the length calculator or a mark on the material.



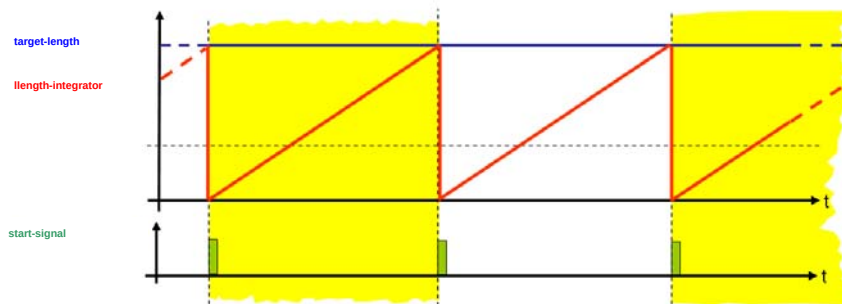
The trigger for the saw axis is the start signal (start trigger), in the diagram above triggered by an internal length calculator. The saw axis then starts and catches up all increments that have passed. After the "oversynchronous" movement the saw is synchronous. Unlike the "synchronous" synchronisation the axis starts immediately at the cutting mark and catches up all increments to achieve synchronicity.

Switching between synchronous and oversynchronous operation is performed via the application control word using the *Synchronisation mode* bit (default bit 05). Here setting the bit means that the axis is synchronised synchronously. If the bit is not set the axis moves oversynchronously during the synchronisation movement.



2.6 Length-controlled operation

In the case of length-controlled operation, a master encoder on the material registers the material speed of the production process. This information is processed by the controller. Cutting marks on the material are not required. The cutting length control provides equidistant cutting lengths. The advantage of cutting length control is that no cutting marks are required on the material. A length calculator calculates equidistant lengths in the controller and generates a start signal for the synchronising process.

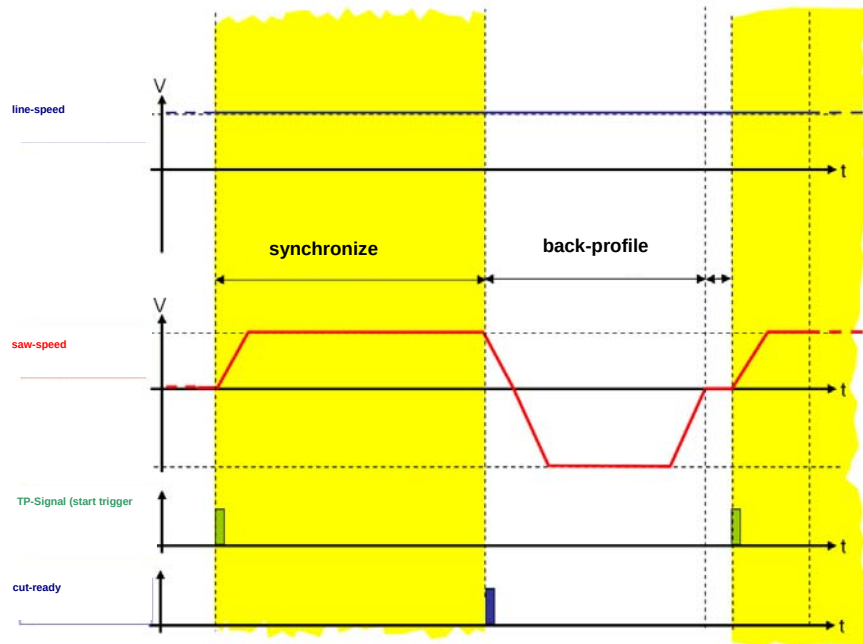


The length calculator integrates the material speed registered by the measuring wheel or by a digital frequency coupling. The integral, designated with length integrator in the above illustration, starts at zero and runs against the setpoint length entered. If the setpoint length is reached, a Boolean signal (start signal) is generated and the flying saw starts the synchronising process. Meanwhile the length integrator is reset to the value zero, so that the next length can be calculated.



2.7 Mark-controlled operation

A sensor, e.g. a photoelectric barrier, registers the cutting marks present on the material. This sensor signal is processed as an interrupt in the inverter and starts the sawing process. This method is used if there are cutting marks on the material which have to be referred to, e.g. when using printed materials.

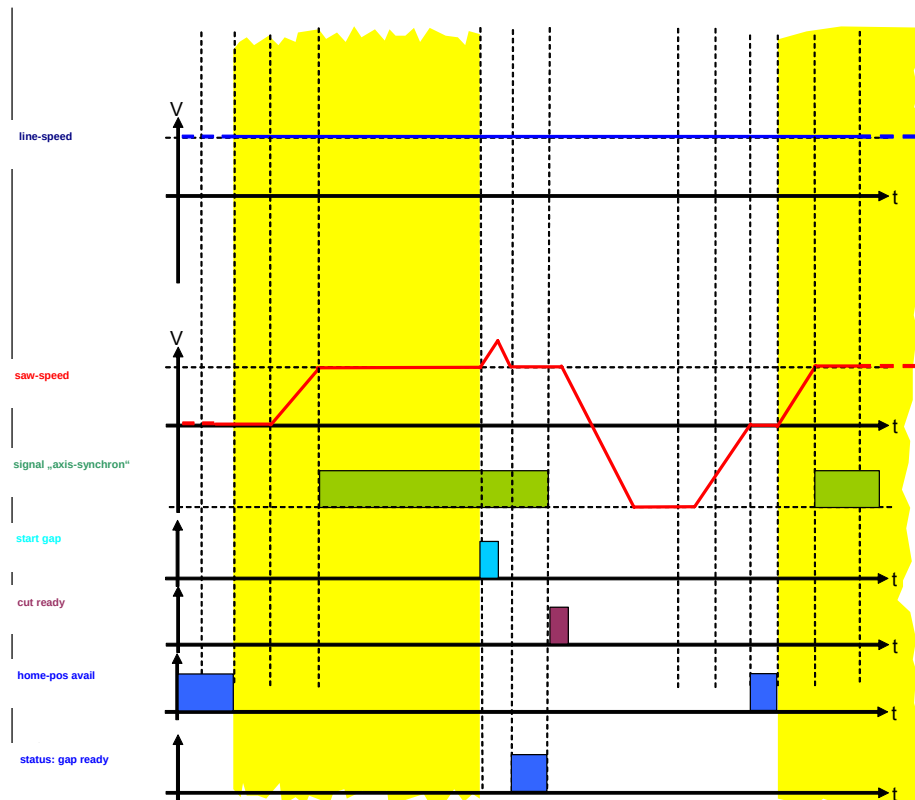


The above diagram shows a mark-controlled flying saw. The start signal for a synchronising process then is not generated by the length calculator as in the case of length-controlled operation, but it is directly cut on the material on marks. The marks are read in the drive via touch probe and the motion process is started, as illustrated above.



2.8 "Start gap" function

Using the "start gap" function the items cut can be separated after cutting.



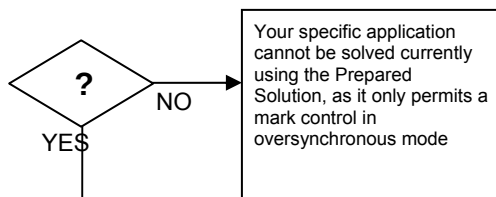
Using the "make gap" function the saw carriage is briefly moved oversynchronously in relation to the material before the saw blade is pulled out. In this way a gap forms between the cut edge and the saw blade and marks from saw blade on the cut edge are prevented. This function is suitable for cut edge protection for delicate material. This function can also be used for separating the cut material.



2.9 Applicability criteria for the parameterisable Prepared Solution

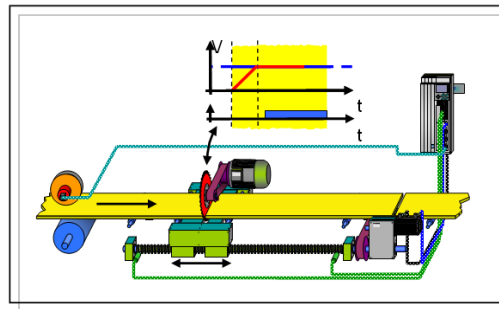
In order to be able to check and estimate the suitability of the Prepared Solution with regard to your application case, the following decision criteria and boundary conditions within which the Prepared Solution can be used are provided

Are the possible combinations adequate for your specific application?

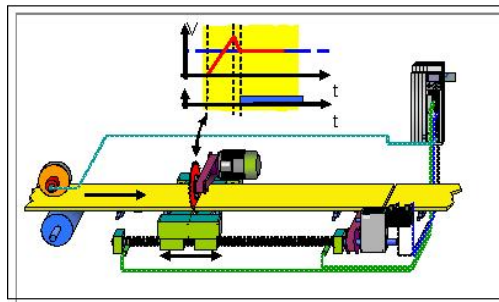


Depending on the specific application, different combinations for the use of the flying saw can be considered

1. Synchronous synchronisation, during this process the master value is measured by a measuring wheel. The lengths are calculated by the internal length calculator.

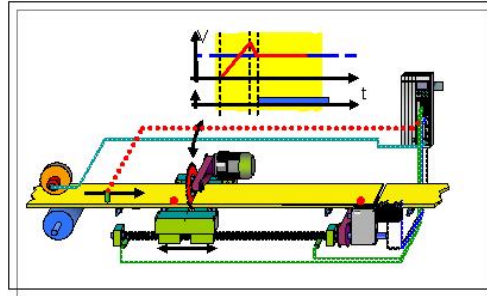


2. Oversynchronous synchronisation, during this process the master value is measured by a measuring wheel. The lengths are calculated by the internal length calculator.

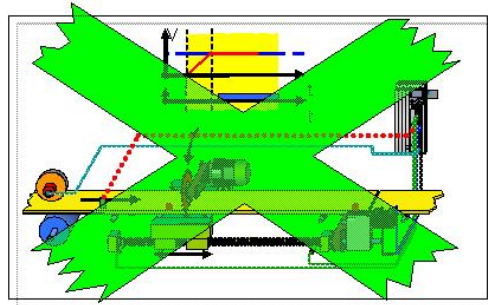




3. Oversynchronous synchronisation, during this process the master value is measured by a measuring wheel. The lengths are detected from marks on the material and are processed.



4. Synchronous synchronisation, during this process the master value is measured by a measuring wheel. The lengths are detected from marks on the material and are processed.



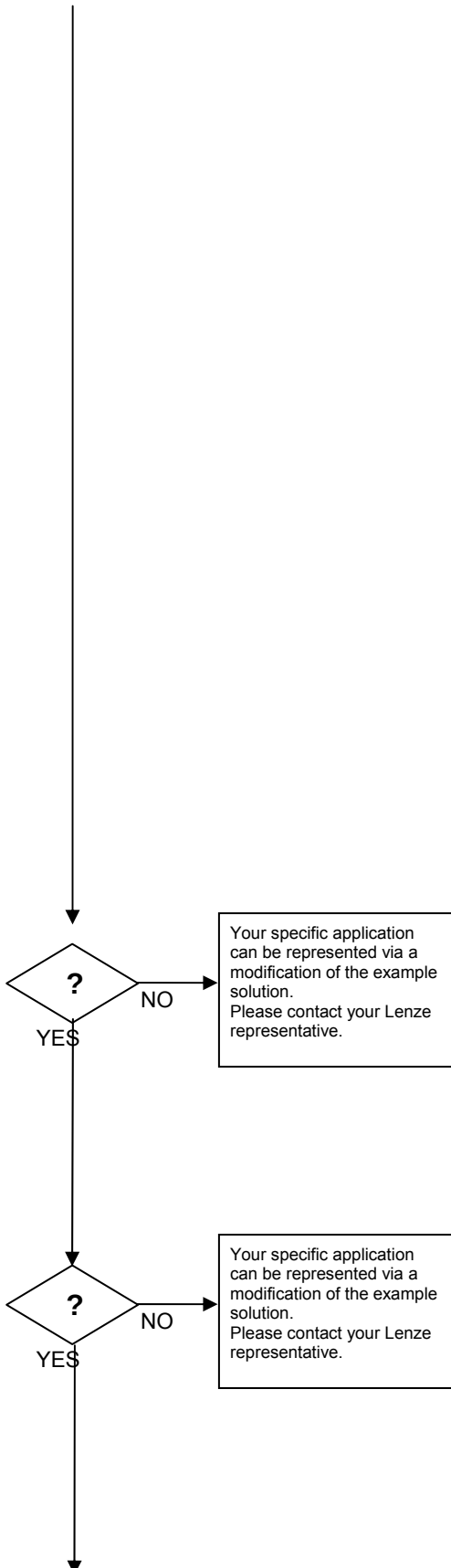
Can the following physical requirements be taken into account in your application?

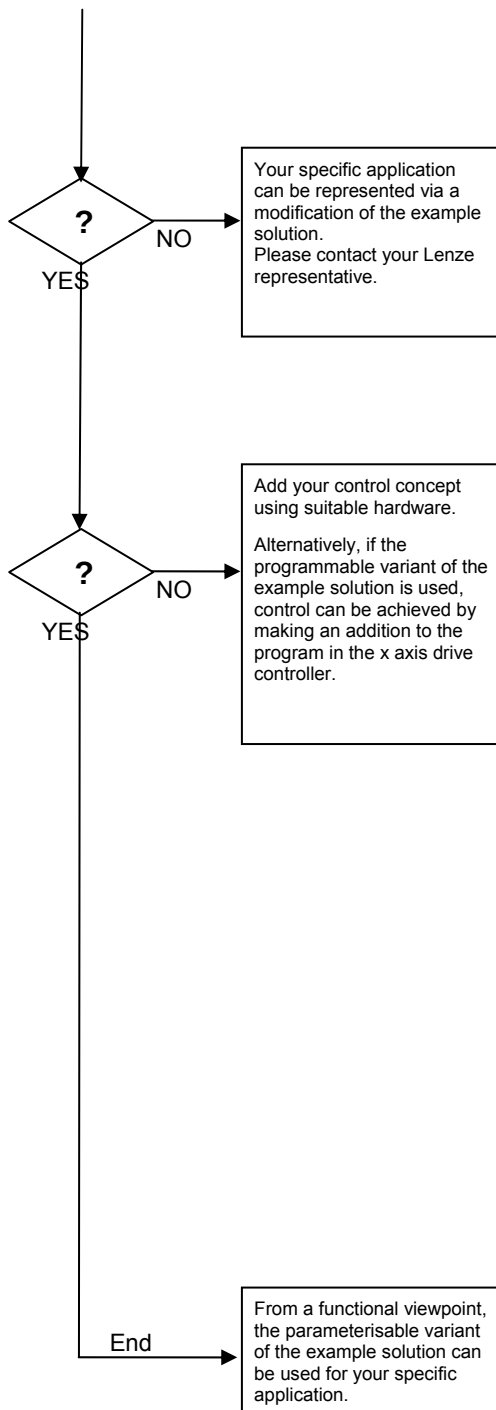
You should take into account the following basic design items in your application:

- Free travel (distance from initial position – end position) must be sufficient so that a process sequence can be performed successfully
- Maximum cut duration
- Maximum velocity of the carriage
- Maximum synchronisation velocity

Do you want to process in "parallel" and your process does not require "diagonal" processing?

During the acceleration to the line speed there is no stopping of the line? (start of flying saw to hit the synchronous)





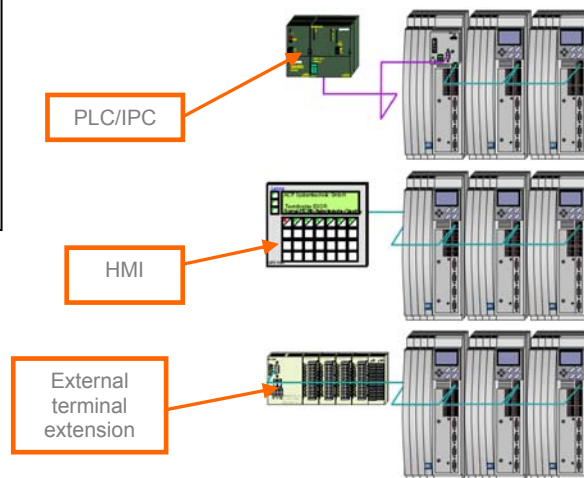
Is this process sequence acceptable for your application?

Process sequence

The process can only be started if the flying saw is in its initial position:

- There is no print mark in the sensor's field of view (mark control), currently the length calculator is not generating a start signal

To generate the control signals for the drive system and to evaluate the states, further components are required in addition to the drive controllers:



Is a higher-level PLC/IPC connected via a bus system, a terminal extension (e.g. Lenze series EPM terminal extensions) or an HMI connected using the system bus or the automation interface (AIF)?



Machine parameters

The machine parameters define the relationship between the application measuring system and the incremental measuring system (internal measuring system in the controller). The following variables define the relationship between one motor revolution and a feed motion in the application measuring system:

- Feed constant: this defines by how many application units the load moves when the drive makes one complete revolution on the output side of the gearbox. This constant is generally given in [units/rev].
Example: on a spindle drive the feed constant is the same as the pitch on the spindle, scaled in [mm/revolution].
- Gearbox ratio, divided into numerator and denominator term: both the numerator and denominator term are generally given as integers (corresponding to the number of teeth on the gearbox stages).
- Resolution of one motor revolution: the ServoPLC internally resolves one motor revolution into 65536 ($=2^{16}$) steps.

The conversion between the two measuring systems can be carried out with the knowledge of these mechanical machine constants using the following formulae:

a) Conversion from application units [units] to incremental units [incr.]:

$$[incr.] = [units] \cdot \frac{Z_i}{N_i} \cdot \frac{65536[incr./rev]}{m_{FeedConstant} [units / rev]}$$

b) Conversion from incremental units [incr.] to application units [units]:

$$[units] = [incr.] \cdot \frac{N_i}{Z_i} \cdot \frac{m_{FeedConstant} [units / rev]}{65536[incr./rev]}$$

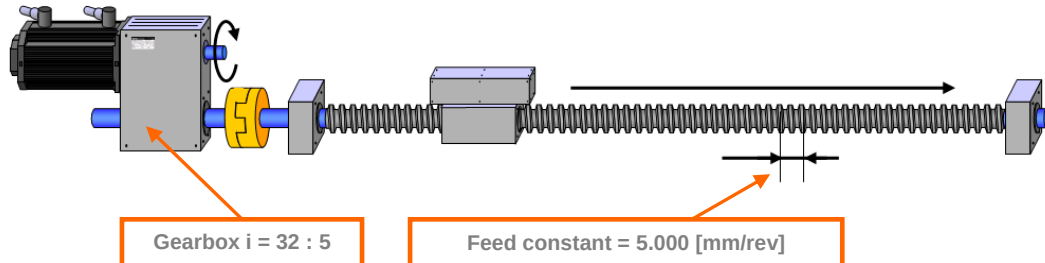
With	Z_i	= Gearbox ratio numerator
	N_i	= Gearbox ratio denominator
	$m_{FeedConstant}$	= Feed constant

As part of the "FlyingSaw" Prepared Solution, the mechanical machine constants are entered using codes.



Example:

A position is to be converted from [units] = 123.4567mm into an incremental value. The mechanical system for this example has the following characteristics:



$$[incr.] = [units] \cdot \frac{Z_i}{N_i} \cdot \frac{65536[incr./rev]}{m_{FeedConstant}[units/rev]} = 123.4567[mm] \cdot \frac{32}{5} \cdot \frac{65536[incr./rev]}{5.0000[mm/rev]} = 10356298[incr.]$$



TIP!

When the parameterisable variant of the "Flying saw" Prepared Solution is used, the conversion of variables in application units into incremental variables is effected automatically.

The machine parameters are set in the user codes using a parameter setting tool (e.g. GDC or DDS).

Code	Possible settings:			Comments
	Default	Selection		
C1202	-	0 ...	{1} ... 65535	Gearbox factor numerator
C1203	-	0 ...	{1} ... 65535	Gearbox factor denominator
C1204	-	0.0000...	{0.0001 [units]/rev} ... 214748.0000	Feed constant



2.10 Functions of the control inputs and outputs

2.11 Manual control

With a "high" signal on these inputs, the saw carriage can be moved manually in both directions at the jog speed defined. The traverse path is limited at the end positions in each case by the two software limit switches. On completion of a jog movement, the saw carriage is held in its new position electrically. After a jog movement, the saw carriage must be moved back to the initial position by a positioning movement. A start signal for performing a cut is only accepted if the saw carriage is in the initial position.

Manual control in the positive direction is started using the application control word with the *Manual control in positive direction* bit (default bit 02).

Manual control in the negative direction is started using the application control word with the *Manual control in negative direction* bit (default bit 03).

2.12 Start gap

On completion of the cut, a gap can be generated between the piece cut and the length of material using this input. For this purpose the saw carriage is briefly accelerated and in this way generates the gap required by the operator. The size of the gap can be adjusted. The "Gap done" output signals when the separation is complete.

The gap is started using the application control word with the *Start make gap* bit (default bit 08).

2.13 Cut done

This input **must** receive a signal as soon as a cut or processing operation has been completed mechanically. This feedback triggers the saw carriage return cycle to its initial position. The direction of the signal (rising or falling edge) can be adjusted. If there is no signal, the saw blade moves to the rear software limit switch and triggers an alarm. Each "Cut done" signal increments the built-in cut counter

The cut done signal is triggered using the application control word with the *Cut done signal* bit (default bit 07).

2.14 Immediate cut / top cut

A rising edge on this input starts a cutting process immediately, independent of the cutting length set. The next cut again is the same as the pre-selected length, unless an immediate cut is triggered again. This function for instance makes it possible to cut out poor sections of material during production is running. The built-in scrap counter is incremented with each immediate cut.

A top cut is started using the application control word with the *Trigger top cut* bit (default bit 06).



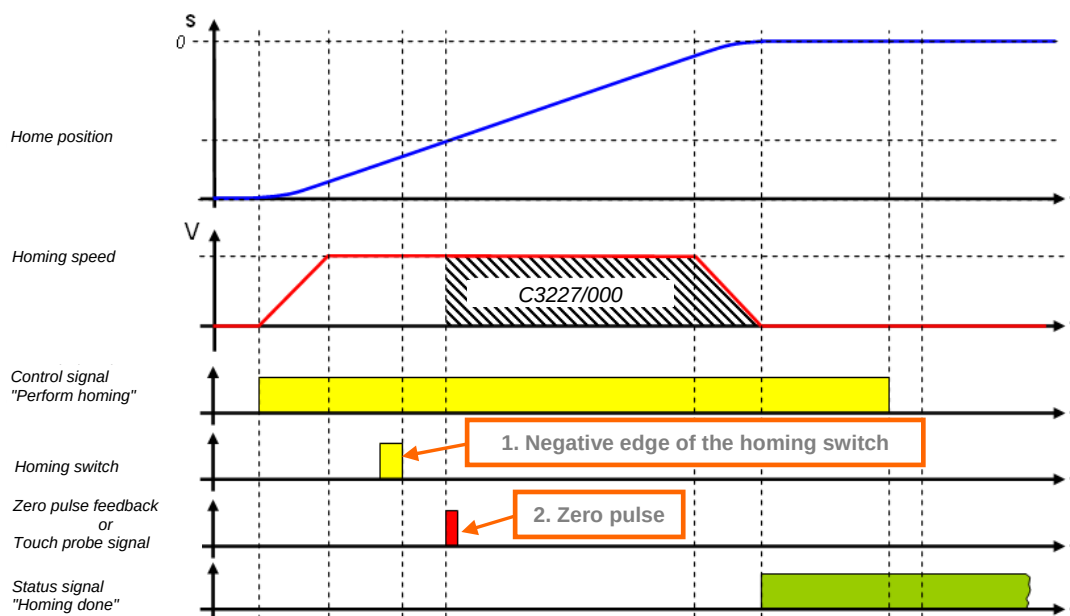
2.15 Homing

After the machine is switched on, the zero point for this measuring system must be made known to the drive (home position). As the home position is mostly defined by specific sensors (e.g. home proximity switches, limit switches or the zero pulse from the motor feedback system), it is necessary to move past these sensors as part of homing. During this process the homing mode defines the traversing direction for the homing and the type of signals evaluated. The following homing modes can be selected:

- Homing mode 0:

The drive rotates in positive direction (clockwise rotation) until the negative edge from the homing switch is detected. The next zero pulse from the motor feedback system or the next touch probe signal on the digital input allocated defines the home position (machine zero point).

If the home position does not represent the zero point of the absolute measuring system used, the home position can be defined in relation to the zero point for the measuring system using an offset (machine zero distance, C3227/000). After the homing has been completed, the drive stays on the position 0.000[units]:



In the example shown above, the home position represents a negative value in relation to the measuring system zero point, because after the acquisition of the zero pulse/touch probe signal the drive continues to move in the positive direction to a position 0.0000[units].



- Homing mode 1:
(like homing mode 0, but negative traversing direction)



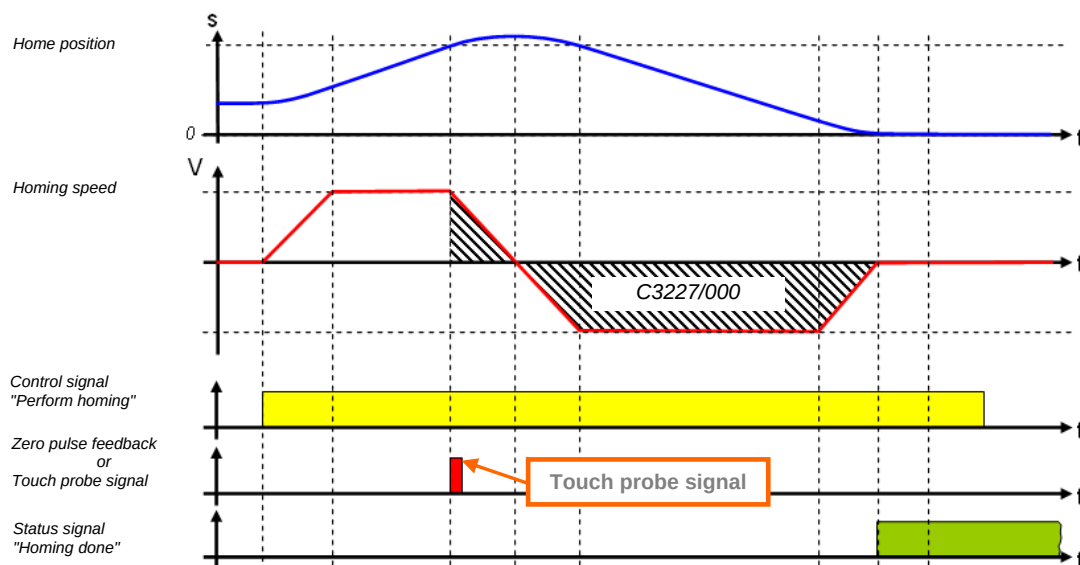
TIP!

Instead of the zero pulse for the feedback system, it is also possible to define a touch probe signal as the signal for the homing, derived from digital input E4 (however, this is not common in mode 0 and 1).

Homing mode 8:

The drive rotates in the positive direction (clockwise rotation) until a zero pulse from the motor feedback system is detected or a touch probe signal is detected on the related digital input. This signal defines the home position (machine zero point).

If the home position does not represent the zero point of the absolute measuring system used, the home position can be defined in relation to the zero point for the measuring system using an offset (machine zero distance, C3227/000). After the homing has been completed, the drive stays on the position 0.0000[units]:



In the example shown above, the home position represents a positive value in relation to the measuring system zero point, because after the acquisition of the zero pulse/touch probe signal the drive reverses and moves back to the measuring system zero position (0.0000[units]).

- Homing mode 9:
(like homing mode 8, but negative traversing direction)



TIP!

Instead of the touch probe signal from digital input E4, it is also possible to define the zero pulse for the feedback system as the signal for defining the homing (however this is not common in modes 8 and 9).



If limit switches are used for the homing, it must be ensured that the drive detects a limit switch at the latest after a traverse path of $\pm 2^{31}-1$ increments. Otherwise the homing must be started again using a positive signal edge on the corresponding control signal.

Apart from the homing mode (C3213/000), the speed (C3242/000) for the homing and the ramp times (C3252/000) for the homing can be set using further parameters.

Altogether the following parameters have an effect on the sequence of the homing:

Code	Possible settings:		Comments
	Default	Selection	
C0911	0	0: Zero pulse of the position feedback system (MP) 1: Touch probe input (TP, terminal E4)	Possible setting for the homing signal
C3213	0	0: >_Rn_MP/TP 1: <_Rn_MP/TP 8: >_MP/TP 9: <_MP/TP	Definition of the homing mode: <u>Symbology:</u> > Movement in positive direction < Movement in negative direction Lp Positive limit switch Ln Negative limit switch Rp Positive edge on the homing switch Rn Negative edge on the homing switch MP/TP Zero pulse from the motor feedback system or touch-probe edge on a digital input
C3225	0	-214748.0000... {0.0001 [units]} ... 214748.0000	Home offset
C3242	100	1 {1[rpm]} ... 16000	Speed for the homing
C3252	5.00	0.01 {0.01[s]} ... 650.00	Ramp times for the homing: These ramp times refer to C0011/000.



TIP!

In the parameterisable variant of the flying saw the homing switch is permanently linked to digital input 3 on the servo PLC!



2.16 Manual control

The Prepared Solution makes it possible to move the axis manually by hand. During this process a differentiation is made between positive and negative inching mode. The parameters that can be set for manual control apply to both traversing directions.

Activated software limit positions are monitored in the manual control mode and trigger an error in the case of overtravel.

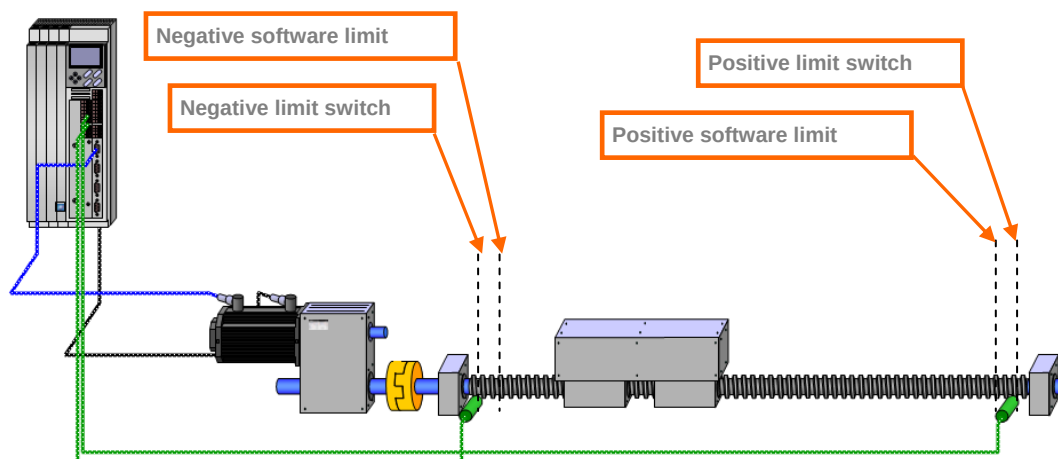
Code	Possible settings:			Comments
	Default	Selection		
C3100	500.0000	-214000.0000...	{0.0001 [units/s]} ... 214000.0000	Velocity for manual operation (inching) The selection relates to the entry for the machine parameter
C3101	750.0000	-214000.0000...	{0.0001 [units/s^2]} ... 214000.0000	Entry for the acceleration The selection relates to the entry for the machine parameter
C3102	750.0000	-214000.0000	{0.0001 [units/s^2]} ... 214000.0000	Entry for the deceleration The selection relates to the entry for the machine parameter



2.16.1 Limit switch integration

2.16.2 Software limit positions

By the use of software limit positions the user can limit the absolute traversing range of the drive in individual operation in a defined manner. In this way the drive can be used reliably for each individual axis. The software limit positions are programmable positions at the outer ends of an absolute traversing range and are normally placed at a short distance in front of the hardware limit switches.



Activated software limit positions ensure that the drive does not move past the software limit positions under any circumstances, even if the drive is set to positions beyond the software limit positions by the positioning control (e.g. continuous signal on manual inching, positioning to an invalid target position, ...). In these cases early braking and shutdown of the drive is initiated by the respective software limit position.

The software limit positions are active if

- the home position for the measuring system is known (accordingly the software limit positions are inactive during homing) **and**
- the positive software limit position (C3223/000) is set greater than the negative software limit position (C3224/000).



The distance between the software limit positions must not be more than $2^{31}-1$ increments.

Code	Possible settings:			Comments
	Default	Selection		
C3223	3600.0000	0.0000... {0.0001 [units]}	... 214748.0000	Definition of the positive software limit <u>Comment:</u> software limits are only active if the home position for the axis is known and is not set to 0.
C3224	-3600.0000	-214748.0000... {0.0001 [units]}	... 0.0000	Definition of the negative software limit <u>Comment:</u> software limits are only active if the home position for the axis is known and is not set to 0.



TIP!

If a 0 is entered for the software limit positions, the software limit positions are deactivated.

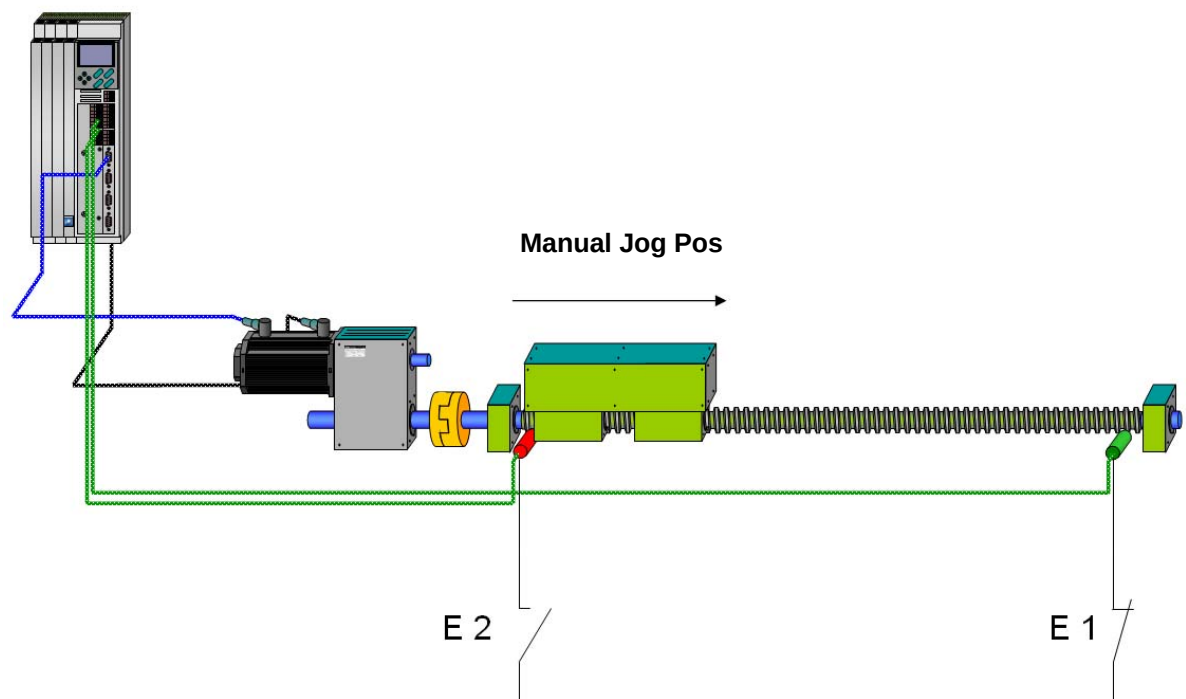


2.17 Moving clear from a limit switch

If the drive moves to a limit switch in the negative or positive direction, then it is possible to retract the actuated limit switch in the opposite direction by using the inching mode.

Example:

The negative limit switch (E2) has been reached and triggers the error with the number 401. The limit switch can be retracted using the "inching positive" function and the error is reset automatically. The axis automatically remains stationary after retracting.



The same applies to the software limit positions. The software limit position to which the drive has moved, in the positive or negative direction, can also be retracted in the opposite direction using the inching mode.



TIP!

After a limit switch that has been approached is retracted, the axis stops automatically and changes to the "Standby" state



Attention!

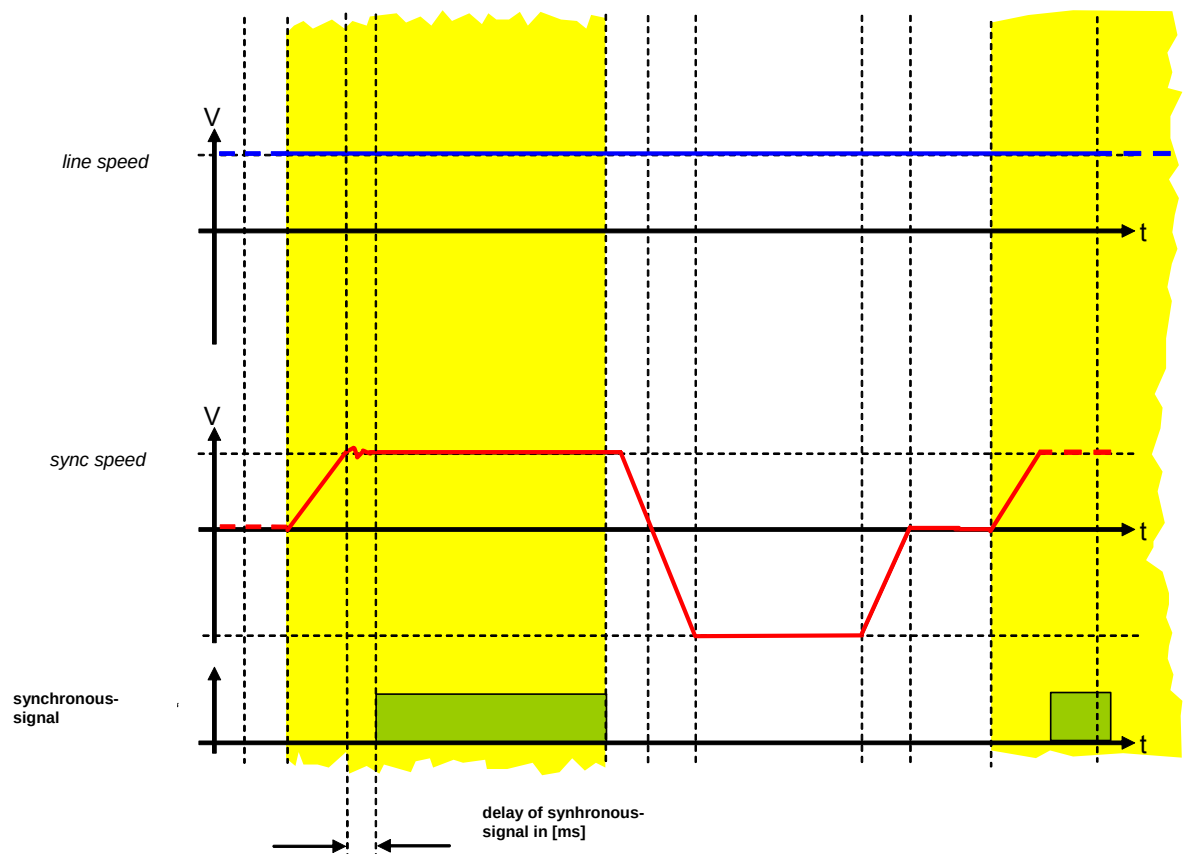
For the retracting function after a damping process, hardware limit switches may not be overtravelled! If limit switches are overtravelled, the retracting function is deactivated.



2.18 Delay on the synchronised signal

After synchronisation the flying saw is synchronised by speed and angle with the master value. However, on reaching the master speed a settling process takes place that must be taken into account for the synchronised signal. For this reason it is possible to set a delay using the code C3009/000. The optimal setting is achieved if a synchronising process is recorded using GDO.

In the example below the effects of C3009/000 can clearly be seen:





2.19 Status signal "Double length"

If it is detected in the system that the saw axis will not reach the initial position before a new cut has been calculated, the *Double length detected* signal is set in the application status word (default bit 17).

The double length means that the calculated cut is ignored and a double length of material is processed. If the cutting length is very short and the synchronisation travel very long, it may occur that several cuts cannot be performed. The signal is then also set. However you will not receive any information on how many pieces or cuts have been ignored.

2.20 Status signal "Mark not detected"

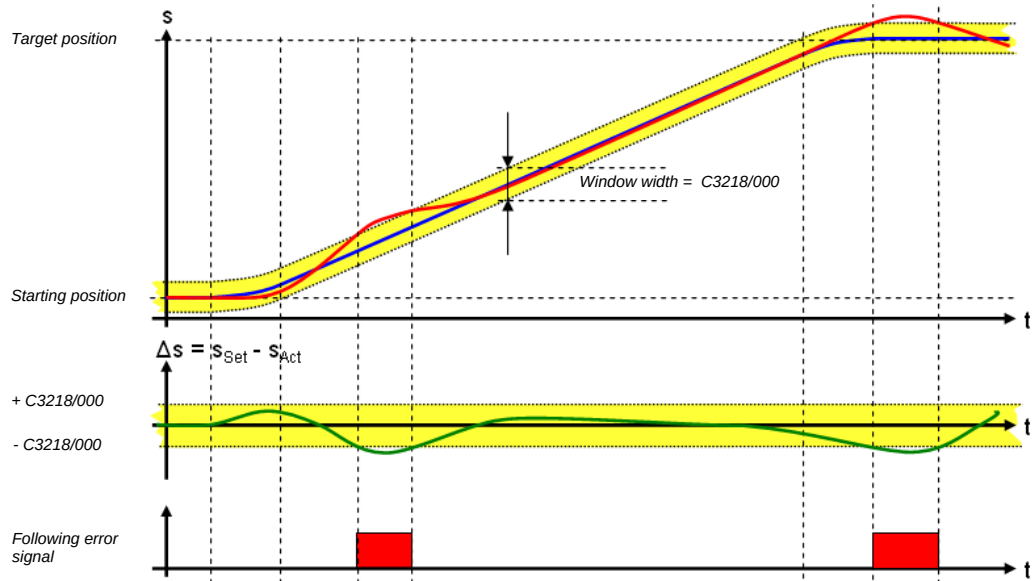
If a mark is not detected in mark-controlled operation, the "Mark not detected" signal is set in the application status word (default bit 19)

The status signal requires, however, active mark monitoring. The monitoring can be activated in the code C3017/000. If a mark is missed, a cut is made using the length calculator.



2.21 Following error monitoring

The drive system continuously monitors the following error with the servo controller enabled. The following error Δs (green curve) is defined by the difference between the set position s_{Set} (blue curve) and the actual position s_{Act} (red curve). If this deviation exceeds a defined limit, a following error signal is generated:



- Set position of the drive s_{Set}
- Actual position of the drive s_{Act}
- Following error $\Delta s = s_{\text{Set}} - s_{\text{Act}}$

The magnitude of the following error shutdown value can be set using the code C3218/000.

Code	Possible settings:			Comments
	Default	Selection		
C3218	10.0000	0.0001...	{0.0001 ... 214748.0000 [units]}	Entry of the following error shutdown limit



3 Functions

3.1 Cut counter

The cut counter is incremented after each cut started by the length calculator. The counter is incremented by exactly one cut as a function of the "Cut done" signal. In the "Mark control" mode the "Cut done" signal is also used for incrementing.

The cut counter is reset using the application control word with the *Reset cut counter* bit (default bit 15)

3.2 Scrap counter

The operator can trigger the scrap counter externally for sorting out pieces detected visually that do not meet the quality requirements. The scrap counter records each external trigger signal.

The scrap counter is increased using the application control word with the *Increment scrap counter* bit (default bit 13)

The scrap counter is reset using the application control word with the *Reset scrap counter* bit (default bit 16)

3.3 Decrementing cut counter

The cut counter is incremented with each step. If a piece cut cannot be used for some reason (e.g. material defect), the cut counter can be reduced by one. The cut counter is automatically decremented if scrap has been detected.

3.4 Top cut counter

The top cut counter is incremented with each top cut. The immediate cuts triggered by the user are also counted. On the triggering of an immediate cut the top cut counter is incremented, the scrap counter incremented and the cut counter decremented.

The top cut counter is reset using the application control word with the *Reset top cut counter* bit (default bit 14)

3.5 Resetting the counters

Each counter can be reset independently. It is also possible to reset all counters with one signal.

The counters (cut counter, scrap counter and top cut counter) are reset using the application control word with the *Reset counters* bit (default bit 17)



3.6 Automatic operation: length-controlled operation

When the flying saw is operated with length control, equidistant pieces of material are processed. This means that the length calculator calculates the lengths as a function of the master speed and generates the start signal for synchronising. The main feature of length-controlled operation is that there do not need to be any marks on the material to achieve exactly the same lengths.

The length-controlled cut is started with a top cut triggered by the operator. The length calculator then calculates the start signals for synchronising based on the required length entered and the master speed.

Top cuts / immediate cuts result in the immediate synchronisation of the flying saw. Subsequent cuts keep exactly to the required cut length set, as this interrupt is taken into account in the application.

3.7 Automatic operation: mark-controlled operation

When the flying saw is operated with mark control, cuts are made at marks applied to the material. In this case the distance between mark detection and the initial position of the flying saw must be measured and entered in the application so that the cut or the processing can take place exactly at the position of the mark.

If the marks are distributed equidistantly on the material, the length calculator can monitor whether a mark has been detected or not. If this a mark is not detected, the length calculator will "force" a cut. The forcing of the cut and the length of the forced cut must be set in the application.

3.7.1 Mark checking

When the flying saw is operated with mark control it is possible to check the distances between the marks. For this purpose the specified length between the marks is entered and, if a mark is not detected, the length calculator starts the synchronising process. The failure to detect a mark can result in continuous material being too long for subsequent processes and machine damage cannot be excluded. Mark monitoring is activated using the code C3017/000.

3.7.2 Simulation of the master speed

The simulation of the master speed makes it possible to operate the flying saw axis without a real master value on interface X9. For this purpose an incremental value in the unit [inc/ms] is defined in the code C3010/000 simulation of the master speed. The simulation is activated in the application control word using the Select master frequency bit [default bit 20]. If the bit is set there, the master value from code C3010/000 is processed in the project. If the bit is not set, the interface X9 is processed in the project.



3.8 Measuring material speed

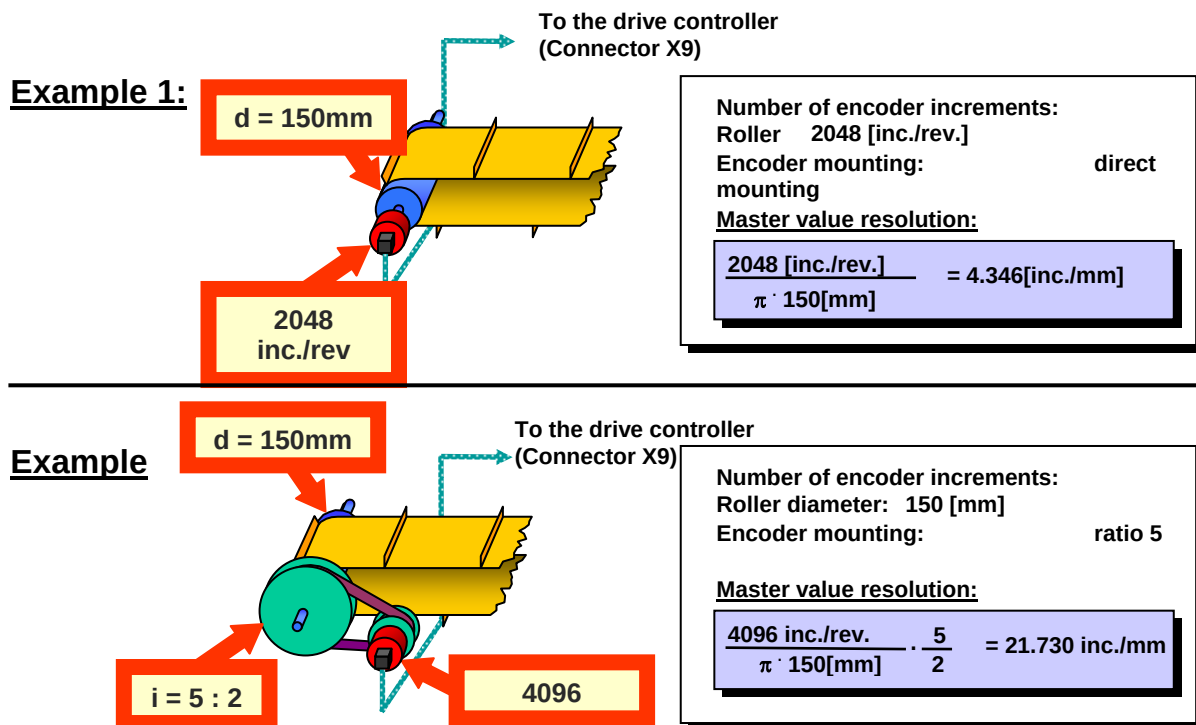
To be able to set the cutting length for the sawing process, the material speed must be known. The material speed can be measured in the following manner:

An encoder is fitted as close as possible to the "flying saw" without any slip. This encoder is connected as an external encoder to the master frequency input X9 on the saw carriage drive. The velocity and the material position are determined using the incremental position information from the external encoder. At least 10 increments should be available each time accuracy is required to determine the material speed sufficiently accurately.

$$\text{master - resolution}[\text{inc} / \text{mm}] = \frac{\text{encoder - pulse}[\text{inc} / \text{umdr}]}{\text{diameter - measurewheel}[\text{mm}] * \pi}$$

On the use of the master frequency input (connector X9) as the master value, the incoming pulses from the incremental encoder connected are counted. The speed of the master axis is then determined with the aid of the number of encoder increments and the master frequency constant (code C0425/000).

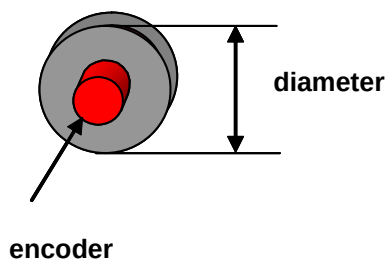
Example master value resolution





3.8.1 Measuring wheel and encoder selection

In order to achieve an exact cutting result, the flying saw requires the speed of the material path. For this purpose, in practice often a measuring wheel in combination with an incremental encoder is used. The measuring wheel is pressed on the material path by means of a spring, so that no slip can develop between the material path and the measuring wheel. Like this, the incremental encoder connected to the measuring wheel measures the speed of the material. Usually incremental encoders with two tracks with zero pulse, shifted by 90 degrees, are used to be able to use the so-called quadruple evaluation that provides for a resolution improvement. The resolution can further be increased by using measuring wheels with a small diameter or by mounting a gearbox between the measuring wheel and the incremental encoder, increasing the speed of the incremental encoder.



In practice it has been shown that the resolution of the registration of the material position has to be 10 times higher than the cutting accuracy required. This means that if a cutting accuracy of 1 mm is to be achieved, the encoder at least has to provide 10 position encoder increments for a material feed of 1 mm. Here one can benefit from the quadruple evaluation brought about by the use of incremental encoders. During the position registration each edge of the position tracks is evaluated, by which the resolution of the position is quadrupled. (A 1024-pulse encoder provides 4096 edges per revolution.) Therefore one position encoder increment corresponds to the $1/(4 \text{ pulse number})$ -th part of a revolution.

Example:

Cutting accuracy required: 0.5 mm
Diameter of impeller: 200 mm
Circumference of impeller: 628.32 mm

The measuring wheel is directly coupled to the incremental encoder.

Calculation of the required edges per revolution to achieve the required accuracy of 0.5 mm:

$$\text{number_of_edges}_{\min} = \frac{10 \cdot U_{\text{Measuring_wheel}}}{\text{required_accuracy}} \cdot i_{\text{Measuring_wheel}}$$

where:

Number of edges_{min} = minimum required number of edges of incremental encoder

$U_{\text{Measuring_wheel}}$ = circumference of measuring wheel

$i_{\text{Measuring_wheel}}$ = gearbox ratio between the measuring wheel and incremental encoder

$$\text{number_of_edges}_{\min} = \frac{10 \cdot 628,32 \text{ mm}}{0,5 \text{ mm}} \cdot 1 = \underline{\underline{12566,4}}$$



Because of the quadruple evaluation of the incremental encoder, furthermore it possible to divide this value by the factor four. Then a value of 3141.6 increments per revolutions results.

As there is no such incremental encoder, the incremental encoder with the next higher number of increments is used. In this example one would use an incremental encoder with 4096 pulses/revolution to maintain the required accuracy of 0.5 mm.

The accuracies specified and to be expected only can be considered with regard to the electrical system and do not include the mechanical conditions, as for instance gearbox backlashes.



Control/status interface to the higher-level control

In the parameterisable variant the control and status interface is already pre-configured. An overview of the signal assignments and possibilities for changing the configuration (multiplexer) of the control/status signals is given below.

The flying saw Prepared Solution can be controlled using a higher-level control over various bus systems (AIF interface):

	Autonomous CAN bus: An additional bus module EMF2171IB or EMF2172IB is required in this case for the AIF slot.
	CANopen: An additional bus module EMF2175IB is required in this case for the AIF slot.
	DeviceNet: An additional bus module EMF2179IB is required in this case for the AIF slot.
	INTERBUS: An additional bus module EMF2113IB is required in this case for the AIF slot.
	Serial data transmission RS232/RS485: The Prepared Solution can be controlled using the AIF automation module EMF2102IB by accessing the control/status codes
	PROFIBUS DP: An additional bus module EMF2133IB is required in this case for the AIF slot.



Control interface

		Meaning
Word 0	System control word	Bit 00: Dependent on the setting in code C4000/000 Bit 01: Dependent on the setting in code C4000/000 Bit 02: Dependent on the setting in code C4000/000 Bit 03: Dependent on the setting in code C4000/000 Bit 04: Dependent on the setting in code C4000/000 Bit 05: Dependent on the setting in code C4000/000 Bit 06: Dependent on the setting in code C4000/000 Bit 07: Dependent on the setting in code C4000/000 Bit 08: Dependent on the setting in code C4000/000 Bit 09: Dependent on the setting in code C4000/000 Bit 10: Dependent on the setting in code C4000/000 Bit 11: Dependent on the setting in code C4000/000 Bit 12: Dependent on the setting in code C4000/000 Bit 13: Dependent on the setting in code C4000/000 Bit 14: Dependent on the setting in code C4000/000 Bit 15: Dependent on the setting in code C4000/000 C4000/000 = 0: System variables <i>DCTRL_wAIF1Ctrl</i> and <i>DCTRL_wCAN1Ctrl</i> are always set to zero. C4000/000 = 1, 2, 3: The drive control word is copied to <i>DCTRL_wCAN1Ctrl</i>; <i>DCTRL_wAIF1Ctrl</i> is set to zero. C4000/000 = 4, 5, 6: The drive control word is copied to <i>DCTRL_wAIF1Ctrl</i>; <i>DCTRL_wCAN1Ctrl</i> is set to zero.
Word 1	Data word 1	(not used)
Word 2	Application control word	Bit 00: Start of homing Bit 01: Homing switch Bit 02: Manual control in positive direction Bit 03: Manual control in negative direction Bit 04: Start automatic Bit 05: Synchronisation mode (TRUE = synchronous, FALSE = oversynchronous) Bit 06: Trigger top cut Bit 07: Cut done signal Bit 08: Start make gap Bit 09: Move to initial position Bit 10: Activate mark cut Bit 11: Reset error Bit 12: Set user error Bit 13: Increment scrap counter Bit 14: Reset top cut counter Bit 15: Reset cut counter Bit 16: Reset scrap counter Bit 17: Reset counters (global all) Bit 18: (not used) Bit 19: CINH Bit 20: Selection of the master frequency source, TRUE = simulation (code), FALSE = DFIN (X9) Bit 21: (not used) Bit 22: (not used) Bit 23: (not used) Bit 24: (not used) Bit 25: (not used) Bit 26: (not used) Bit 27: (not used) Bit 28: (not used) Bit 29: (not used) Bit 30: (not used) Bit 31: (not used)
Word 3		



The signal sources for the data words (shown in the illustration as *Word 0*, *Word 1*, *Word 2* and *Word 3*) are dependent on the operating mode C4000/000:

Signal source	Word 0	Word 1	Word 2	Word 3
C4000/000 = 0	- ³	C3261/000	C4135/000	
C4000/000 = 1	CAN1_wDctrlCtrl	CAN1_nlnW1_a	CAN1_nlnW2_a	CAN1_nlnW3_a
C4000/000 = 2	CAN2_nlnW1_a	CAN2_nlnW2_a	CAN2_nlnW3_a	CAN2_nlnW4_a
C4000/000 = 3	CAN3_nlnW1_a	CAN3_nlnW2_a	CAN3_nlnW3_a	CAN3_nlnW4_a
C4000/000 = 4	AIF1_wDctrlCtrl	AIF1_nlnW1_a	AIF1_nlnW2_a	AIF1_nlnW3_a
C4000/000 = 5	AIF2_nlnW1_a	AIF2_nlnW2_a	AIF2_nlnW3_a	AIF2_nlnW4_a
C4000/000 = 6	AIF3_nlnW1_a	AIF3_nlnW2_a	AIF3_nlnW3_a	AIF3_nlnW4_a

The figure on the previous page of the 32-bit application control word (*word 3* and *word 4*) shows the factory-set default configuration. The user can also adapt the configuration of the application control word using the codes C4010/001 to C4010/032 to suit specific requirements.

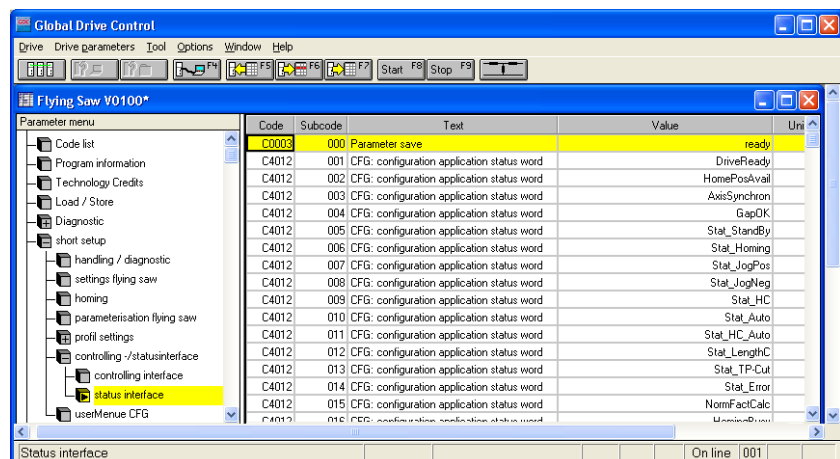
As an example it is shown in the following how the application control word can be re-configured bit by bit with the aid of the parameter setting software **GDC**:

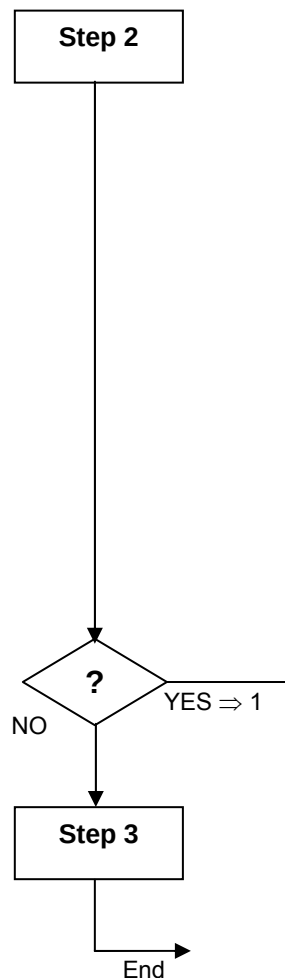
Start

Step 1

Select the code C4010/00x allocated to the function to be configured.

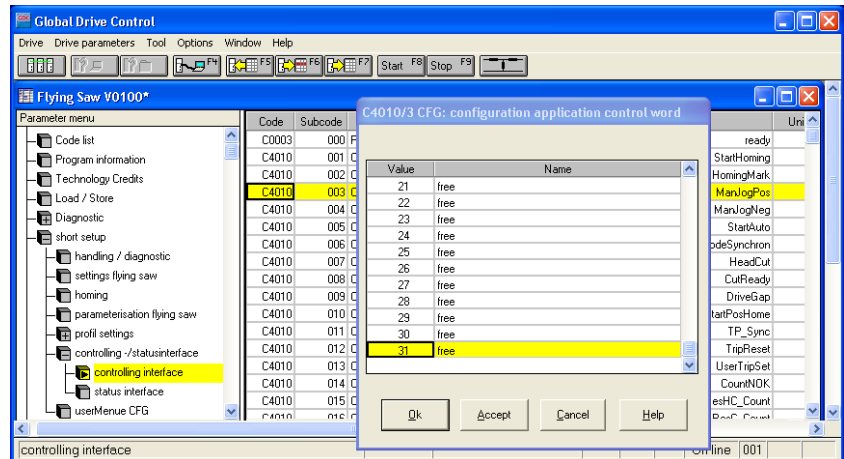
Example: To define the control signal for positive manual control on the x axis the code C4010/003 is available:





Click the code and select the required control bit.

Example: To control the positive manual control on the X axis bit 31 must be used. Set C4010/008 = 31 and accept using OK.



If the configuration of further control bits is to be changed, continue with step 1.

Save the modified settings safe against mains failure using C0003/000 = 1.



Code	Possible settings:		Comments
	Default	Selection	
C4000	0	0: Internal control words 1: CAN 1 2: CAN 2 3: CAN 3 4: AIF1 5: AIF2 6: AIF3	Preselection of the control interface: The axis can be controlled using the control interface.
C4010			Configuration codes for the control interface:
1	0	0: Control bit 0	1: Start of homing
2	1	1: Control bit 1	2: Homing switch
3	2	2: Control bit 2	3: Positive manual control
4	3	3: Control bit 3	4: Negative manual control
5	4	4: Control bit 4	5: Start automatic
6	5	5: Control bit 5	6: Synchronising mode
7	6	6: Control bit 6	7: Trigger top cut
8	7	7: Control bit 7	8: Cut done signal
9	8	8: Control bit 8	9: Start make gap
10	9	9: Control bit 9	10: Move to initial position
11	10	10: Control bit 10	11: Activate mark cuts
12	11	11: Control bit 11	12: Reset error
13	12	12: Control bit 12	13: Set user error
14	13	13: Control bit 13	14: Increment scrap counter
15	14	14: Control bit 14	15: Reset top cut counter
16	15	15: Control bit 15	16: Reset cut counter
17	16	16: Control bit 16	17: Reset scrap counter
18	17	17: Control bit 17	18: Reset all counters
19	18	18: Control bit 18	19: CINH
20	19	19: Control bit 19	20: (not used)
21	20	20: Control bit 20	21: Selection of the master frequency source
22	21	21: Control bit 21	22: (not used)
23	22	22: Control bit 22	23: (not used)
24	23	23: Control bit 23	24: (not used)
25	24	24: Control bit 24	25: (not used)
26	25	25: Control bit 25	26: (not used)
27	26	26: Control bit 26	27: (not used)
28	27	27: Control bit 27	28: (not used)
29	28	28: Control bit 28	29: (not used)
30	29	29: Control bit 29	30: (not used)
31	30	30: Control bit 30	31: (not used)
32	31	31: Control bit 31	32: (not used)
C4135	0	0 ... {1} ... 4294967295	Control code: Using the control code the flying saw can be controlled (with C4000/000 = 0). The individual bits of the control code can be defined as required using the configuration codes C4010/001 ... 0032. The factory-set bit assignment is given in code C4010/xxx.
C4136	-	0 ... {1} ... 4294967295	Indication of the actual application control word: Dependent on C4000/000 the actual application control word is displayed here. The meaning of the individual bits of the application control word can be defined as required using the configuration codes C4010/001 ... 0032. The factory-set bit assignment is given in code C4010/xxx.



Status interface

Meaning		
Bit 00: Dependent on the setting in code C4000/000	System status word	Word 0
Bit 01: Dependent on the setting in code C4000/000		
Bit 02: Dependent on the setting in code C4000/000		
Bit 03: Dependent on the setting in code C4000/000		
Bit 04: Dependent on the setting in code C4000/000		
Bit 05: Dependent on the setting in code C4000/000		
Bit 06: Dependent on the setting in code C4000/000		
Bit 07: Dependent on the setting in code C4000/000		
Bit 08: Dependent on the setting in code C4000/000		
Bit 09: Dependent on the setting in code C4000/000		
Bit 10: Dependent on the setting in code C4000/000		
Bit 11: Dependent on the setting in code C4000/000		
Bit 12: Dependent on the setting in code C4000/000		
Bit 13: Dependent on the setting in code C4000/000		
Bit 14: Dependent on the setting in code C4000/000		
Bit 15: Dependent on the setting in code C4000/000		
(not used)	Data word 1	Word 1
Bit 00: Program initialisation	Application status word	Word 2
Bit 01: Homing done		
Bit 02: Synchronised signal		
Bit 03: Gap made		
Bit 04: STATUS standby		
Bit 05: STATUS homing active		
Bit 06: STATUS inching in positive direction active		
Bit 07: STATUS inching in negative direction active		
Bit 08: STATUS top cut active		
Bit 09: STATUS automatic selected		
Bit 10: STATUS top cut in the ongoing process		
Bit 11: STATUS length cut active		
Bit 12: STATUS mark cut active		
Bit 13: STATUS error active, global error message		
Bit 14: Normalisation factor calculated		
Bit 15: Homing active		
Bit 16: Positioning active		
Bit 17: Double length (only length-controlled operation)		
Bit 18: Initial position reached		
Bit 19: Mark not detected (only mark-controlled operation)		
Bit 20: Error detected		
Bit 21: Warning		
Bit 22: Message		
Bit 23: Fail-QSP		
Bit 24: (not used)		Word 3
Bit 25: DCTRL ready		
Bit 26: Axis has controller inhibit		
Bit 27: Axis is in QSP		
Bit 28: (not used)		
Bit 29: (not used)		
Bit 30: (not used)		
Bit 31: (not used)		



The output of the individual status words to system variables (shown in the illustration as *Word 0*, *Word 1*, *Word 2* and *Word 4*) is dependent on the operating mode C4000/000:

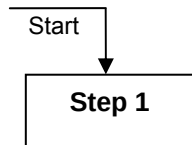
Signal source	Word 0	Word 1	Word 2	Word 3
C4000/000 = 0	- ⁴	-	C4150/000	
C4000/000 = 1	CAN1_wDctrlStat	CAN1_nOutW1_a	CAN1_nOutW2_a	CAN1_nOutW3_a
C4000/000 = 2	CAN2_nOutW1_a	CAN2_nOutW2_a	CAN2_nOutW3_a	CAN2_nOutW4_a
C4000/000 = 3	CAN3_nOutW1_a	CAN3_nOutW2_a	CAN3_nOutW3_a	CAN3_nOutW4_a
C4000/000 = 4	AIF1_wDctrlStat	AIF1_nOutW1_a	AIF1_nOutW2_a	AIF1_nOutW3_a
C4000/000 = 5	AIF2_nOutW1_a	AIF2_nOutW2_a	AIF2_nOutW3_a	AIF2_nOutW4_a
C4000/000 = 6	AIF3_nOutW1_a	AIF3_nOutW2_a	AIF3_nOutW3_a	AIF3_nOutW4_a



The figure on page 3-31 (*word 2 and word 3*) shows the factory-set default configuration of the 32-bit application status word. The user can also adapt the configuration of the application status word using the codes C4012/001 to C4012/032 to suit specific requirements.

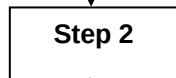
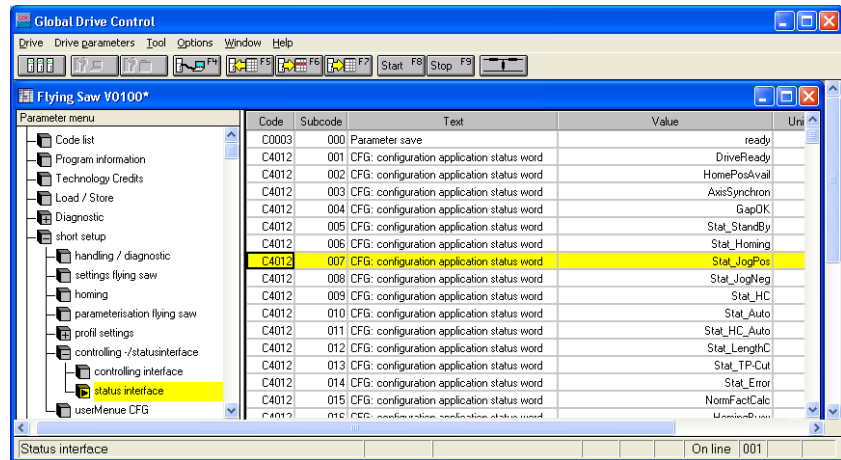
It is shown in the following how the application status word can be re-configured bit by bit with the aid of the parameter setting software **GDC**:

You'll find the code in the complete code list or in the Short setup => Control/status interface menu



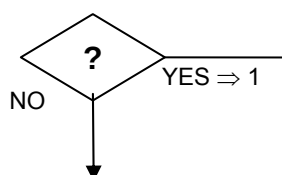
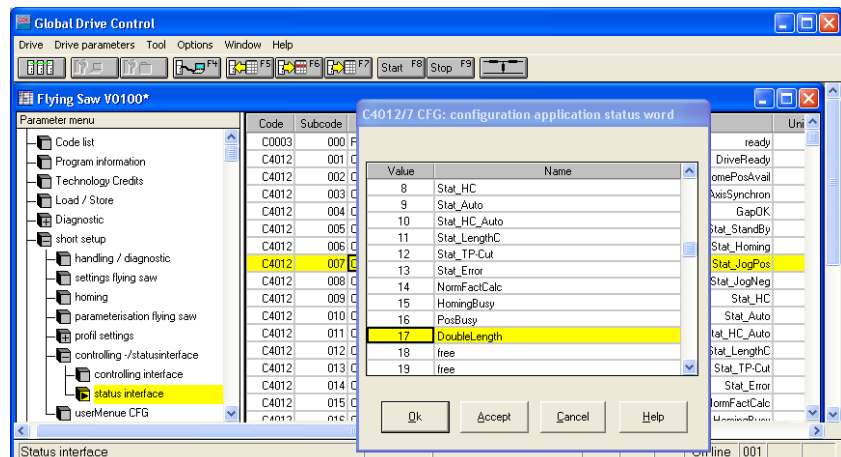
Select the application status word you want to re-configure

Example: To assign bit 6 of the application status word, select the code C4012/007:



Click the code and select the status information that you want to represent using this status bit.

Example: If the signal for the "double length" is to be displayed, select C4012/007 = 17 (Double Length) and accept using Ok.



If the configuration of further status control bits is to be changed, continue with step 1.



Step 3

Save the modified settings safe against mains failure using C0003/000 = 1.

End



TIP!

The application status word is displayed using code C4150/000 independent of the operating mode you select using C4000/000.

Code	Possible settings:		Comments
	Default	Selection	
C4000	0	0: Internal control words 1: CAN 1 2: CAN 2 3: CAN 3 4: AIF1 5: AIF2 6: AIF3	Preselection of the control interface: All three axes can be controlled using the control interface.
C4012			Configuration codes for the status interface:
1	0	0: Program initialisation complete	1: Status bit 0
2	1	1: Homing done	2: Status bit 1
3	2	2: Axis synchronised signal	3: Status bit 2
4	3	3: Gap made	4: Status bit 3
5	4	4: STATUS standby	5: Status bit 4
6	5	5: STATUS homing active	6: Status bit 5
7	6	6: STATUS inching in positive direction	7: Status bit 6
8	7	7: STATUS inching in negative direction	8: Status bit 7
9	8	8: STATUS top cut is being performed	9: Status bit 8
10	9	9: STATUS automatic selected	10: Status bit 9
11	10	10: STATUS top cut in the ongoing process	11: Status bit 10
12	11	11: STATUS length cut active	12: Status bit 11
13	12	12: STATUS mark cut active	13: Status bit 12
14	13	13: STATUS error detected	14: Status bit 13
15	14	14: Normalisation factor calculated	15: Status bit 14
16	15	15: Homing active	16: Status bit 15
17	16	16: Positioning active	17: Status bit 16
18	17	17: Double length detected	18: Status bit 17
19	18	18: Initial position reached	19: Status bit 18
20	19	19: Mark not detected	20: Status bit 19
21	20	20: Error / TRIP	21: Status bit 20
22	21	21: Warning	22: Status bit 21
23	22	22: Message	23: Status bit 22
24	23	23: QSP active	24: Status bit 23
25	24	24: (not used)	25: Status bit 24
26	25	25: DCTRL ready	26: Status bit 25
27	26	26: Axis in controller inhibit	27: Status bit 26
28	27	27: Axis is in QSP	28: Status bit 27
29	28	28: (not used)	29: Status bit 28
30	29	29: (not used)	30: Status bit 29
31	30	30: (not used)	31: Status bit 30
32	31	31: (not used)	32: Status bit 31
C4150	-	0 ... {1} ... 4294967295	Status code (display code): Using the status code the flying saw can be monitored (with C4000/000 = 0). The individual bits of the status code can be defined as required using the configuration codes C4012/001 ... 0032. The factory-set bit assignment is given in code C4012/xxx.
C4152	-	-32767... {1} ... 32768	Indication of the status word 1 (16-bit integer value): This code is not predefined in the factory.



4 Commissioning the "(Flying Saw)"

4.1 Requirements

4.1.1 Required components

Lenze hardware

Product	Type designation	Hardware version	Firmware version
9300ET ServoPLC	EVS93xxET	7A or later	6.5 or later
ECSxA	ECSxA	4B or later	7.4 or later
PC system bus adapter	EMF2177IB	1.3 or later	1.7 or later
Hand-held control unit (keypad XT)	EMZ9371BC		
Master frequency connection			

Lenze software

Product	Type designation	Version
Global Drive Loader (GDLoader)	(Freeware)	3.0 or later
Drive PLC Developer Studio (DDS)	ESP-DDS1-P	2.2 or later
Global Drive Control (GDC)	ESP-GDC2	4.9 or later
Global Drive Oscilloscope		1.2 or later
Library LenzeMotionControl	ESP-SPAC-POS1 Software Package Positioner	4.0

Third-party components

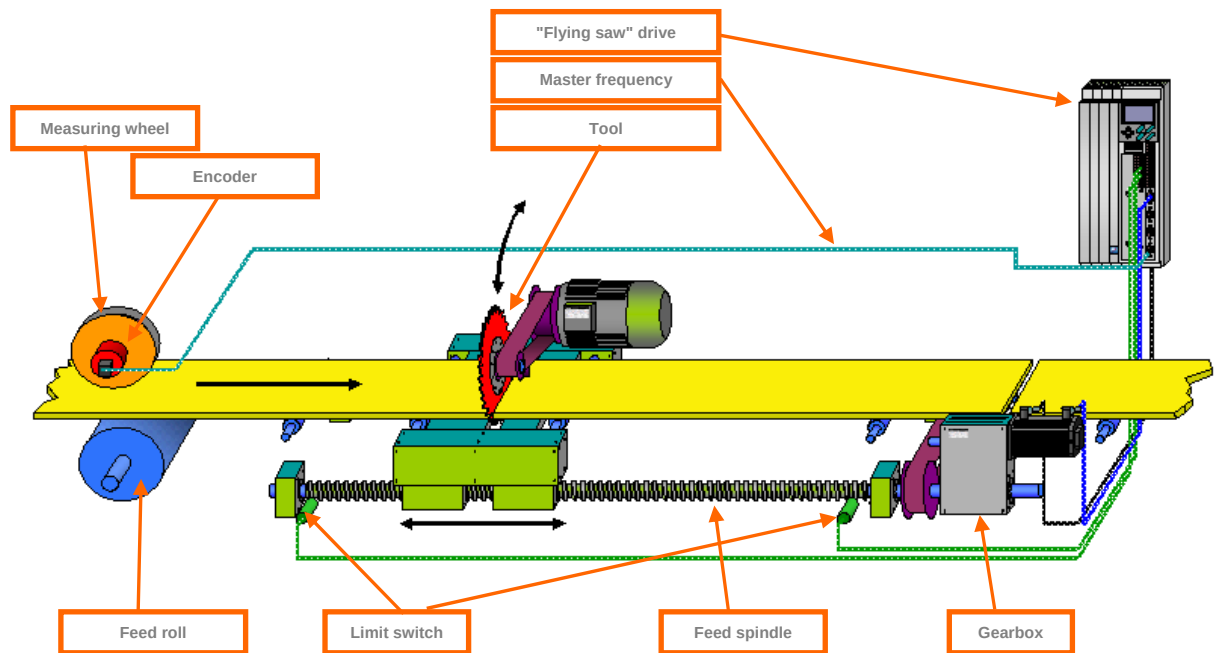
Product	Comment/specification
Limit switch (normally closed contact)	Safety-related sensors, operation without limit switches not allowed
Measuring wheel	Design as per documentation
Encoder (measuring wheel)	Design as per documentation

Only required for programmable variant of the Prepared Solution.



4.2 Hardware layout

The Prepared Solution for the flying saw requires the following hardware configuration:

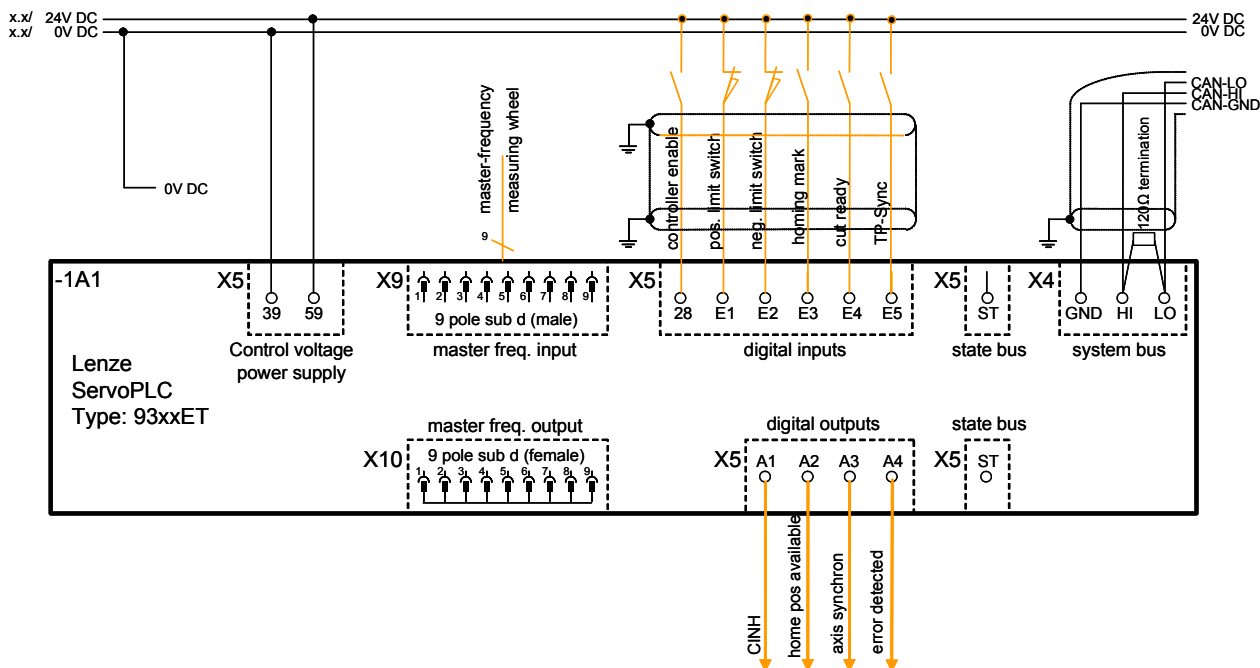


Attention!

Limit switches represent safety-related devices for a linear axis and must always be wired up (see also next chapter)!



4.2.1 Wiring of the ServoPLC control terminals



Attention!

The connection plan shown gives the elementary wiring necessary for the correct function of the Prepared Solution.

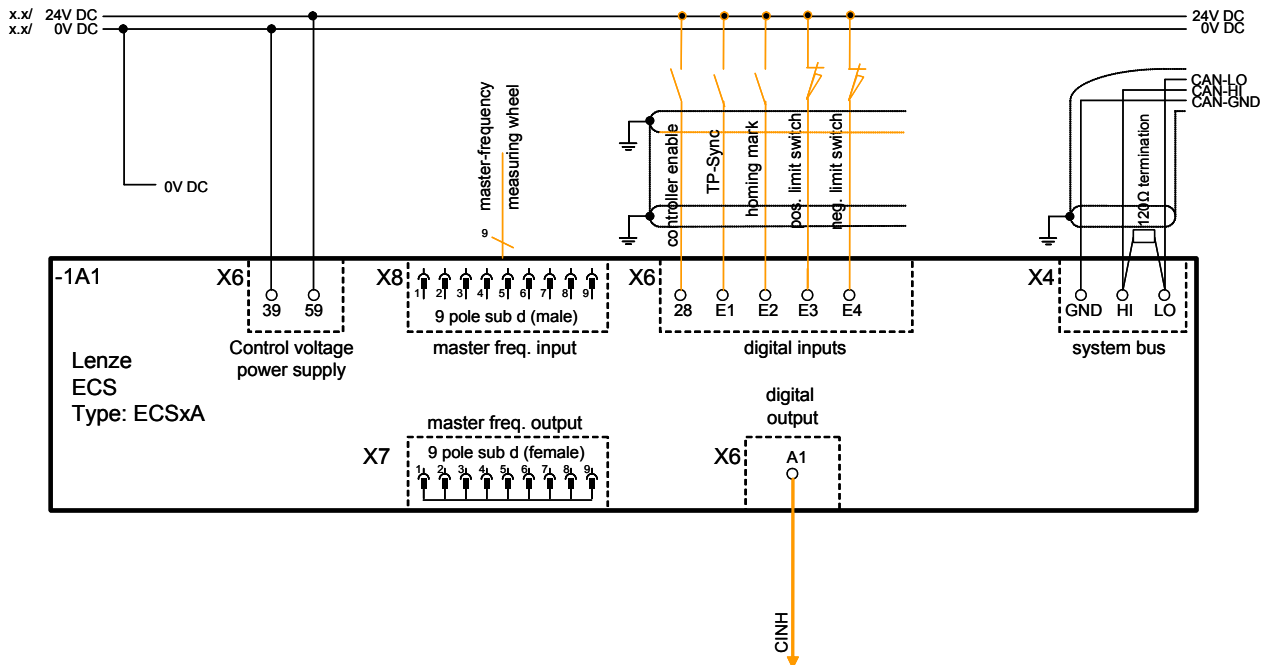


Attention!

The basic wiring (e.g. connection to mains and motor, connection of the feedback system) is not given in this chapter. For further information on the basic wiring of the device, please refer to chapter 4 in the operating instructions for the device (can also be downloaded from the Lenze homepage www.lenze.de).



4.2.2 Wiring of the ECSxA control terminals



Attention!

The connection plan shown gives the elementary wiring necessary for the correct function of the Prepared Solution.



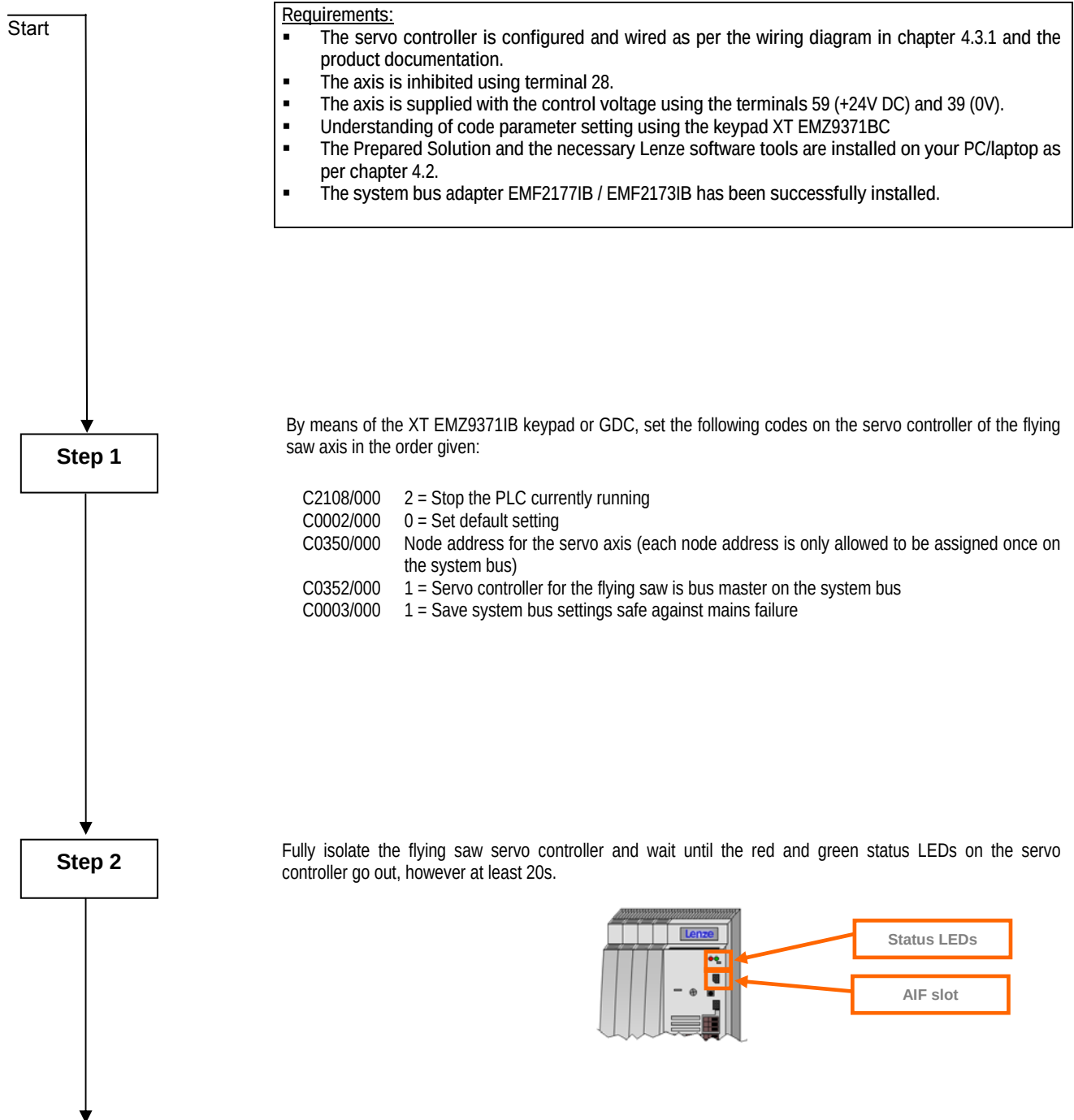
Attention!

The basic wiring (e.g. connection to mains and motor, connection of the feedback system) is not given in this chapter. For further information on the basic wiring of the device, please refer to chapter 4 in the operating instructions for the device (can also be downloaded from the Lenze homepage www.lenze.de).



4.3 Commissioning of the Prepared Solution (parameterisable variant)

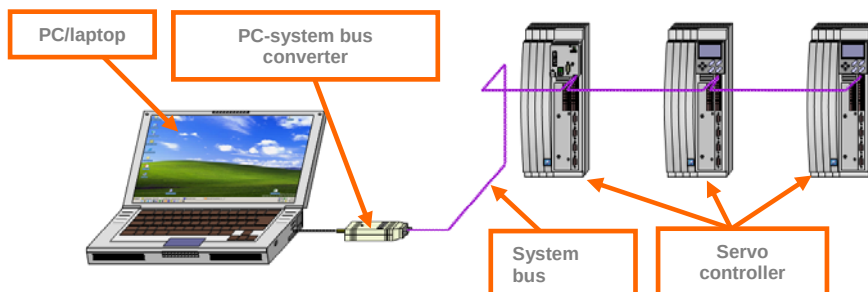
4.3.1 Download of the program FlyingSaw_SPLC_Vxxxxxx / FlyingSaw_ECS_Vxxxxxx





Step 3

Connect your PC/laptop to the system bus for the servo axis using the PC-system bus adapter EMF2177IB / EMF2173IB (it is also possible that you have several axes in the drive system):



TIP!

To make it easier to make the connection a three-pin system bus connector is included with the PC-system bus adapter EMF2177IB / EMF2173IB; using this connector the PC-system bus adapter can be straightforwardly connected to the system bus.

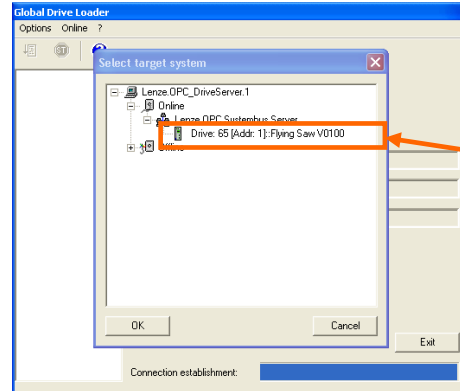
Step 4

Switch back on the control voltage on the servo controller.



Step 5

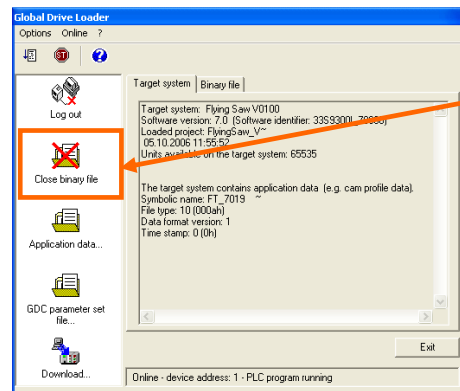
Start the GDLoader and run a search for all drives connected via the system bus. GDLoader now scans the system bus and finds the servo drive:



Click the required servo controller (in the example the controller with system bus node address 1) and accept using Ok.

Step 6

Now allocate the files to the required servo axis by clicking the corresponding icons in the column on the left:



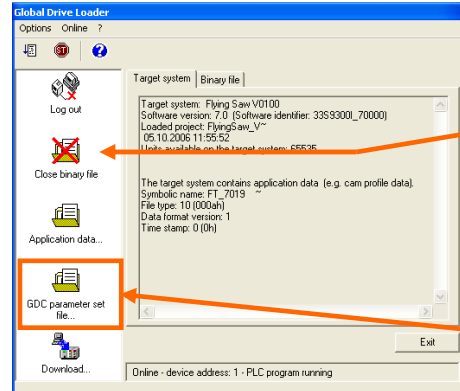
For the DDS binary file select the following file:

Project: FlyingSaw_SPLC_Vxxxxxx.bin (ServoPLC)
FlyingSaw_ECS_Vxxxxxx.bin (ECSxA)



Step 7

Allocate the parameter file (GDC file) by clicking the GDC parameter set file icon in the column on the left: This step is called for operating the GDLoader. The scope of supply of the Prepared Solution does not include a GDC file

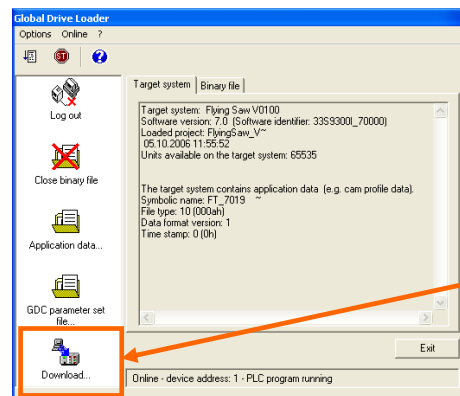


File types for which you have already made an allocation are shown red and crossed out. You can undo the allocation by clicking the crossed-out icon again.

Allocate the prepared parameter set (GDC files):

Step 8

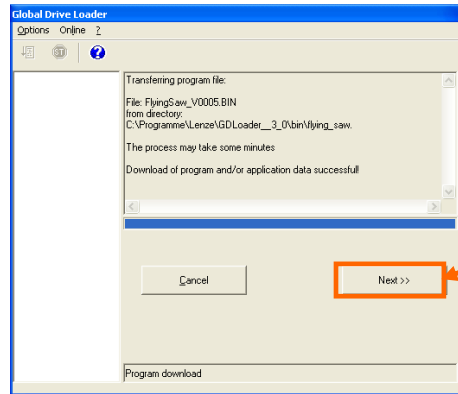
Start the download to the servo axis by clicking the Download icon:





Step 9

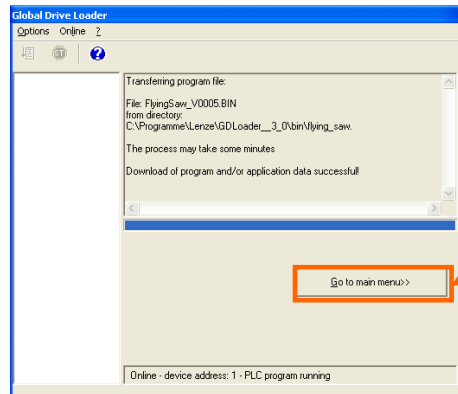
Wait for the download to finish.



Once the download is complete, click Next>>

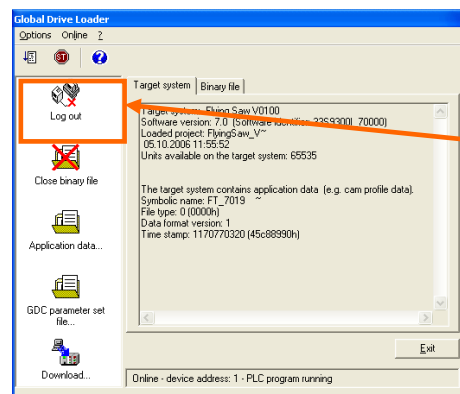
Step 10

Click Go to main menu>> (Continue to main menu) to open the main GDLoader window:



Step 11

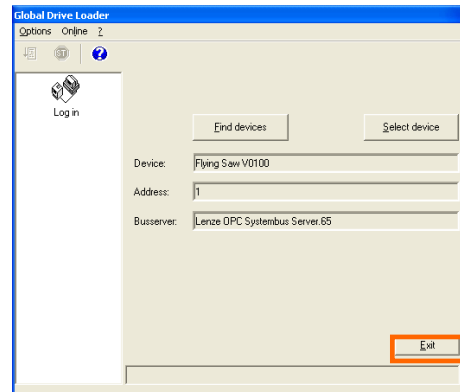
Break the connection to the servo controller by clicking the Ausloggen (Log out) button:





Step 12

Close GDLoader:



End

The program has been fully loaded into the controller.



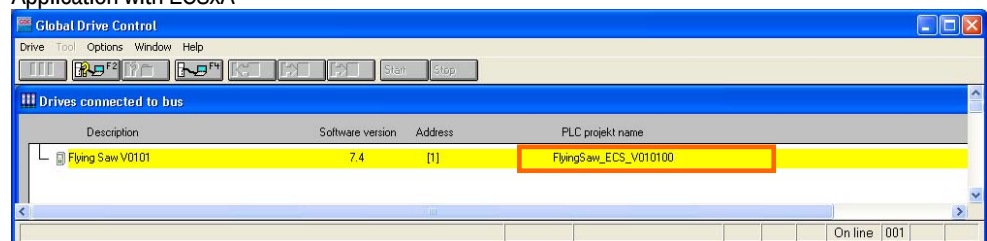
4.3.2 Sequence for online commissioning using GDC

Start

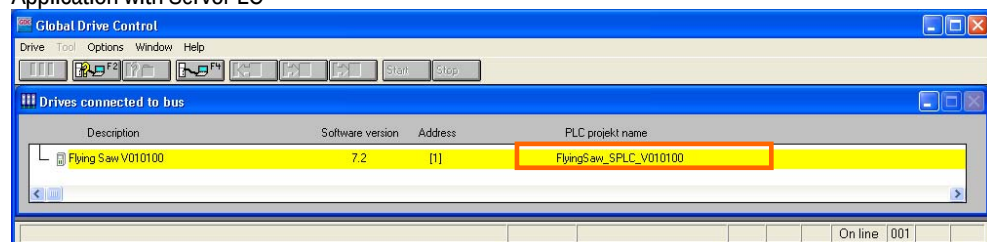
Step 1

Start GDC and run a scan on the system bus for drive controllers connected. GDC finds the drive controllers connected. These include the drive controller that is loaded with the *FlyingSaw_SPLC_Vxxxxxx* project or with the *FlyingSaw_ECS_Vxxxxxx* project.

Application with ECSxA



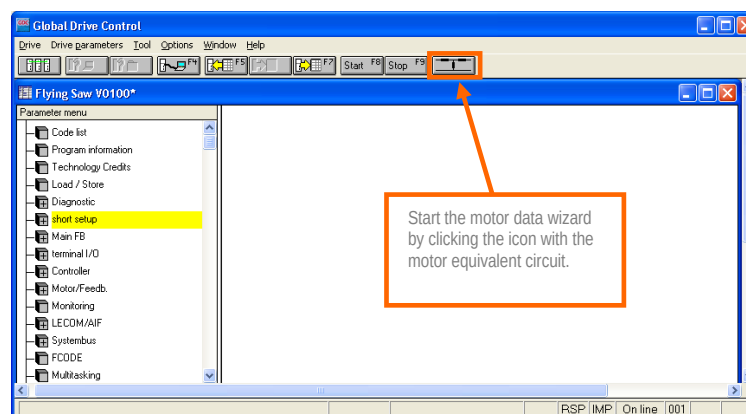
Application with ServoPLC



Step 2

Double-click the drive for the flying saw axis and read the parameters back from the drive.

Start the motor data wizard:

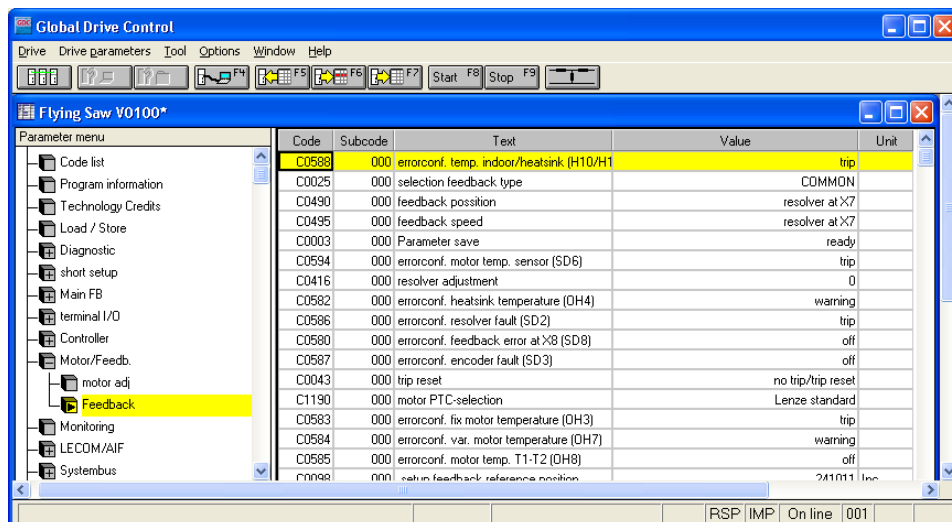


Follow the instructions in the motor data wizard to correctly set the motor connected. Once you have finished setting the parameters for the motor, close the motor data wizard.



Step 3

Open the parameters for the configuration of the speed and position feedback system
((Motor/Feedback systems) → (Feedback systems)):



Set the following codes:

- C0025/000 = Feedback system default setting
- C0420/000 = Number of encoder increments on the use of TTL incremental encoders on X8 or number of periods from Sin/Cos encoders on X8
- C0421/000 = Setting for the supply voltage for the feedback system on X8: as a rule the default value of 5.0V can be retained.
Exception: Absolute value encoders with HIPERFACE interface on X8 require a higher supply voltage of 8.0V.



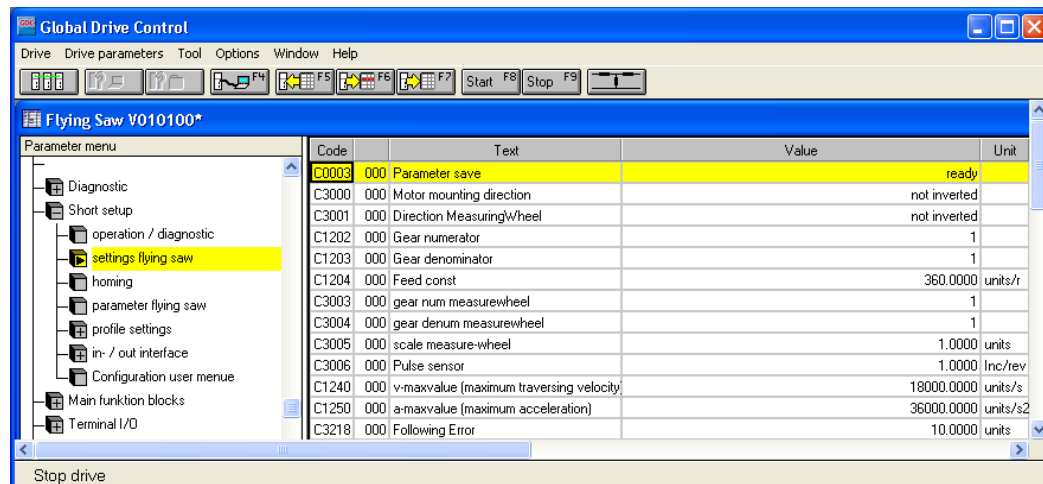
Attention!

Applications with position feedback on the load side are not supported by the parameterisable variant of the Prepared Solution. If it is necessary to use a position feedback system on the load side, use the programmable variant of the Prepared Solution.



Step 4

Now set the machine data for the flying saw
(Short setup → Settings/flying saw):



Set the following codes:

- C3000/000 Setting for the motor mounting position (position polarity)
- C1202/000 Gearbox numerator factor
- C1203/000 Gearbox denominator factor
- C1204/000 Feed constant
- C1240/000 Maximum traversing speed
- C1250/000 Maximum acceleration
- C3218/000 Following error limit (scaled in application units [units])



TIP!

Calculation of the maximum traversing speed (C1240/000).

Caution: The limit must be set such that it can still be reached with the max. motor speed:

$$v_{\max} \leq n_{\max} \cdot \frac{\text{Feed constant}}{60} \cdot \frac{\text{Gearbox (denominator)}}{\text{Gearbox (numerator)}}$$

$$C1240 \leq C0011 \cdot \frac{C1204}{60} \cdot \frac{C1203}{C1202}$$

The maximum acceleration figure is determined from the maximum motor torque and the moment of inertia to be accelerated:

$$\text{Maximum acceleration} = \frac{\text{Maximum motor torque}}{\text{Total moment of inertia}}$$

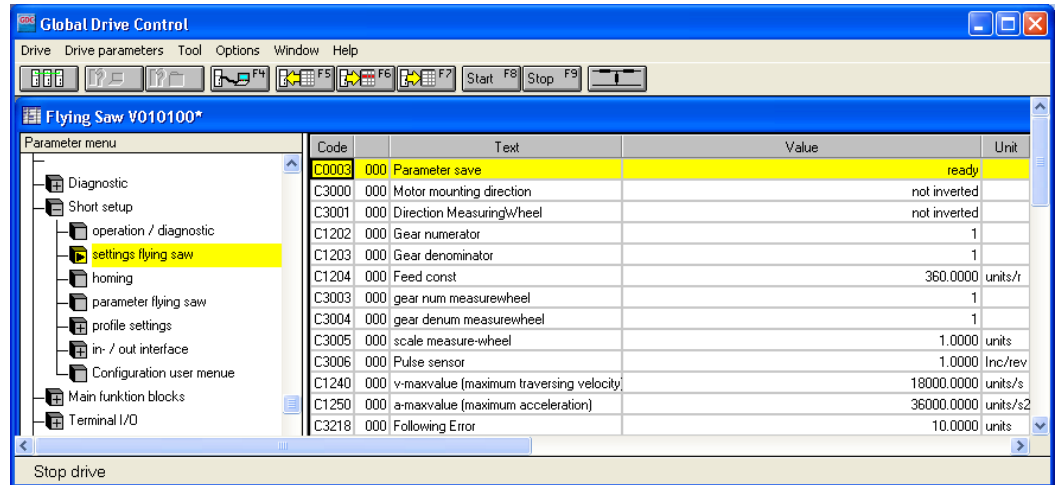
An initial reference value can be defined using the maximum speed set in C1240. If it is ensured the machine can accelerate to maximum speed in a time defined by you, for C1250 the following is found:

$$C1250 = \frac{C1240}{\text{Time from 0 [unit]/s to C1240}}$$



Step 5

Open the parameters for the configuration of the measuring wheel (FlyingSaw_SPLS_V0100 → settings flying saw):



Set the following codes:

- C3003/000 = gear numerator of the measuring wheel
- C3004/000 = gear denominator of the measuring wheel
- C3005/000 = Circumference of the measuring wheel in [units]
- C3006/000 = Encoder pulses from the measuring wheel

The normalisation factor is determined automatically and displayed in the codes C3003 (numerator normalisation factor) and code C3004 (denominator normalisation factor).



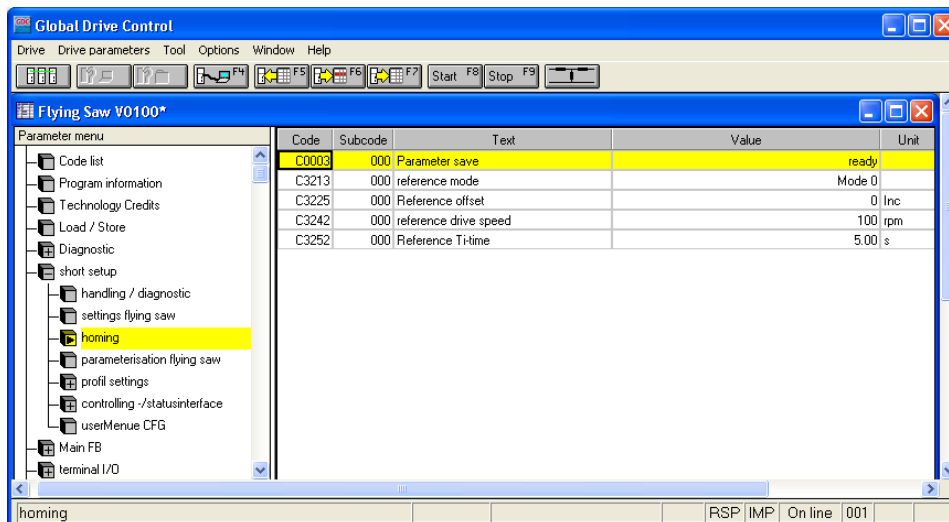
TIP!

The calculation of the normalisation factor is explained in chapter 9.



Step 6

Parameterise the homing next:



Set the following codes for setting the parameters:

- C3213/000 = Homing mode
- C3225/000 = Homing offset (reference: 65536 = one motor revolution)
- C3242/000 = Homing speed in [rpm]
- C3252/000 = Acceleration and deceleration ramps for the homing in [s]

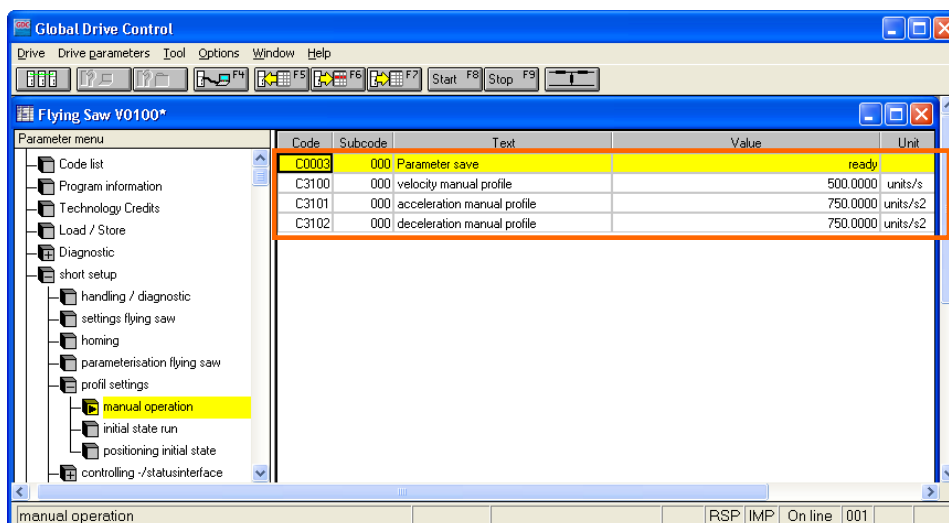


TIP!

You will find further information on the homing in chapter 2.12.1

Step 7

Parameterise the settings for manual jog (short setup → profile settings):



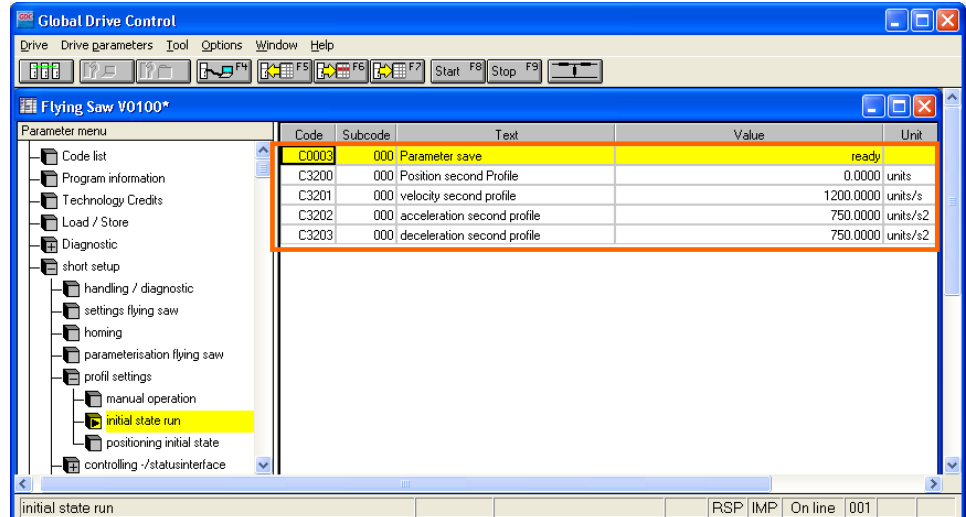
Set the following codes for setting the parameters for manual control:

- C3100/000 = Velocity for manual control in [units/s]
- C3101/000 = Acceleration for manual control in [units/s²]
- C3102/000 = Deceleration for manual control in [units/s²]



Step 8

Parameterise the profiles for the return travel of the flying saw and for the approach of the basic position next :



Set the following codes for setting the parameters:

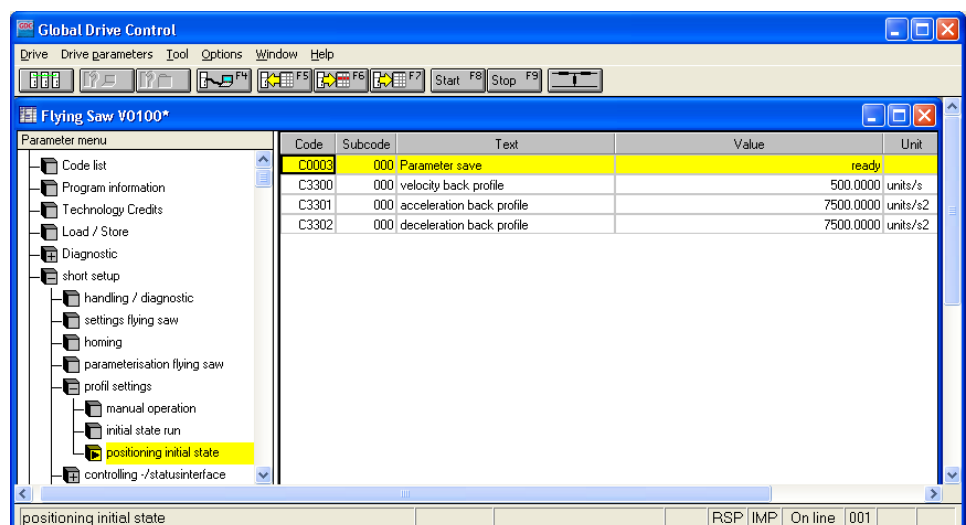
Parameters for the movement to the initial position:

- C3200/000 = Position entry for the movement to the initial position in [units]
- C3201/000 = Velocity for the movement to the initial position in [units/s]
- C3202/000 = Acceleration for the movement to the initial position in [units/s²]
- C3203/000 = Deceleration for the movement to the initial position in [units/s²]

Parameters for the return movement profile after sawing

- C3300/000 = Velocity in [units/s]
- C3301/000 = Acceleration in [units/s²]
- C3302/000 = Deceleration in [units/s²]
- C3303/000 = Jerk time in [s]

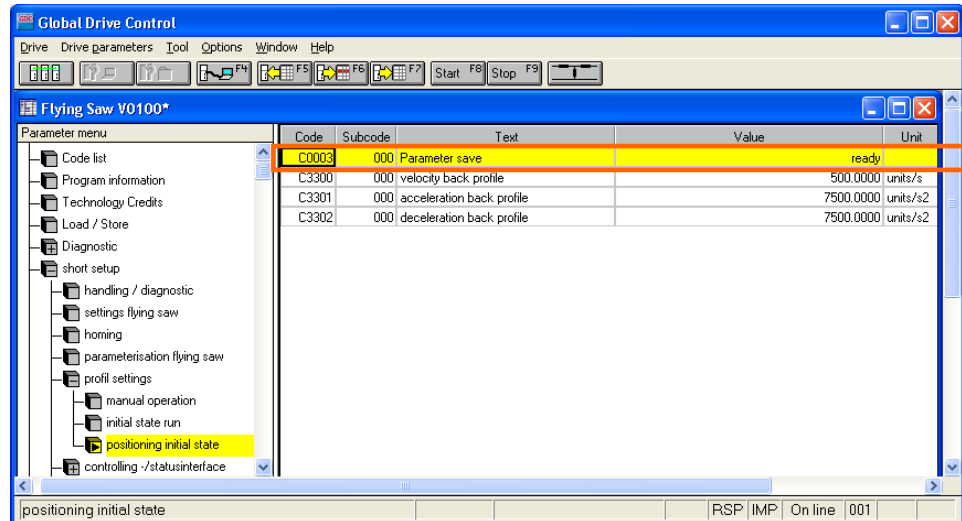
Step 9





Step 10

Now save the parameter set as follows:

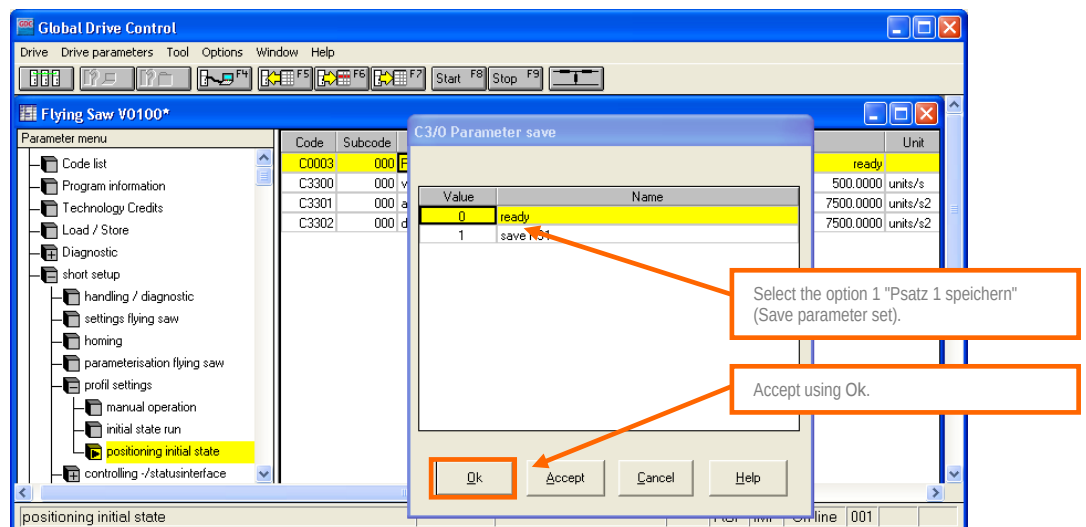


Click code C0003 to save the parameter set:

C0003/000 = Save parameter set

Step 11

A dialog box for code C0003 opens.



The parameter settings are now saved safe against mains failure.

End

The basic settings for the peripheral drive functions are now complete.



4.3.3 Sequence for the online commissioning of the functions



Attention!

The commissioning of the Prepared Solution described in the following relates to the parameterisable variant.

Start

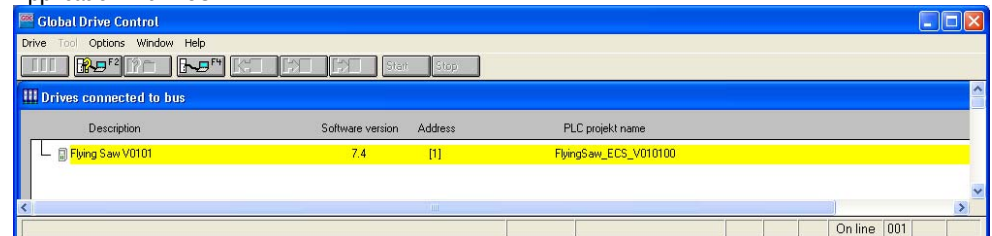
Step 1

Requirements:

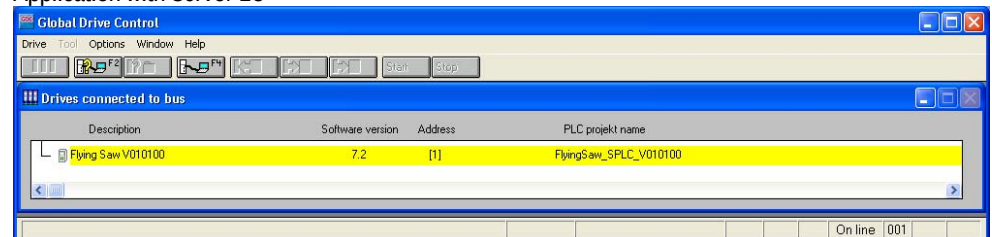
- The parameters for the servo controller for the flying saw axis have been set to suit the specific case, as described in chapter 4.4.3.
- The axis is inhibited using terminal 28.
- The axis is supplied with control voltage using terminals 59 (+24V DC) and 39 (0V) and with power using terminals L1, L2 and L3.

Start GDC and perform a search for the drive controller connected to the system bus. GDC finds the servo axis as follows:

Application with ECSxA



Application with ServoPLC

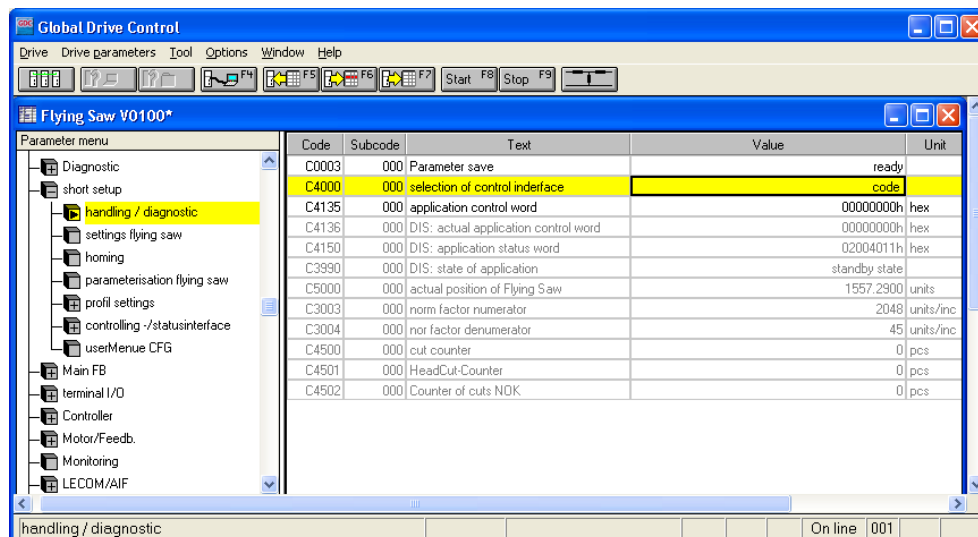


Double-click the servo controller for the flying saw axis and upload the parameter data for the selected drive from GDC to your PC.



Step 2

Select the interface via which you want to control the Prepared Solution :



For this purpose set code C4000/000:

C4000/000 = Control/status interface selection

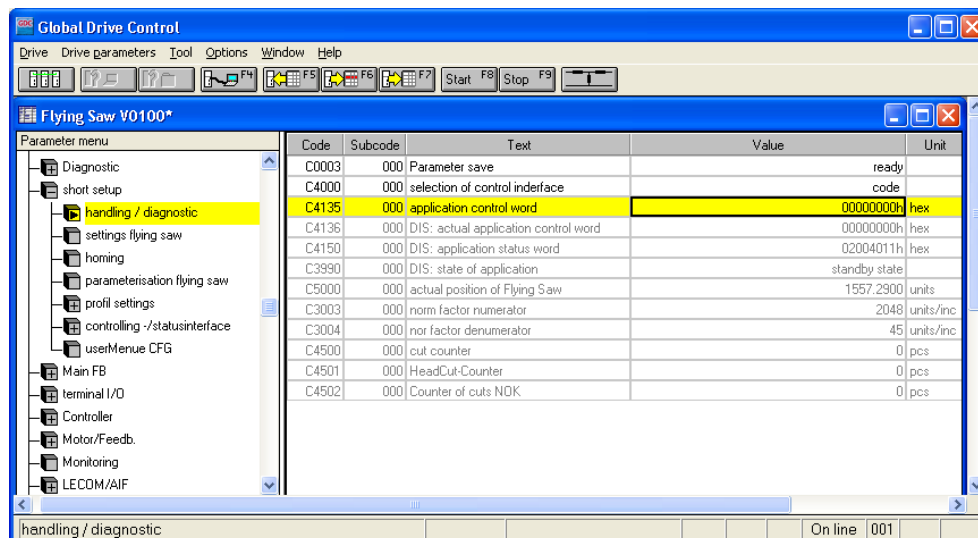


TIP!

The bit assignment for all 32 control bits of the application control word (word 3 and word 4) can be changed as required.

Step 3

Test the correct operation of the application control word (word 3 and word 4) by setting/resetting the control bits for the functions used in the higher-level PLC and checking the corresponding representation of the control commands in code C4136/000:



Only run the test when the servo controller is inhibited.



Step 4

Check the correct feedback of the application status word (word 3 and word 4) from the servo controller for the flying saw axis to the higher-level PLC. For this purpose you can display the actual application status word from code C4150/000 and compare it with the value received in the PLC:

Code	Subcode	Text	Value	Unit
C0003	000	Parameter save		ready
C4000	000	selection of control interface		code
C4135	000	application control word	00000000h	hex
C4150	000	DIS: actual application control word	00000000h	hex
C4150	000	DIS: application status word	02004011h	hex
C3990	000	DIS: state of application	standby state	
C5000	000	actual position of Flying Saw	1557.2900	units
C3003	000	norm factor numerator	2048	units/inc
C3004	000	norm factor denominator	45	units/inc
C4500	000	cut counter	0	pcs
C4501	000	HeadCut-Counter	0	pcs
C4502	000	Counter of cuts NOK	0	pcs

Step 5

Enable the servo controller by connecting terminal 28 to 24V. Test the positive manual control functions by setting the corresponding control bit in the application control word (positive manual control: bit 2).

Code	Subcode	Text	Value	Unit
C0003	000	Parameter save		ready
C4000	000	selection of control interface		code
C4135	000	application control word	00000000h	hex
C4136	000	DIS: actual application	00000000h	hex
C4150	000	DIS: application status word	02004011h	hex
C3990	000	DIS: state of application	standby state	
C5000	000	actual position of Flying	0.0000	units
C4003	000	norm fact num	2048	units/inc
C4004	000	norm fact denum	45	units/inc
C4500	000	cut counter	0	pcs
C4501	000	HeadCut-Counter	0	pcs
C4502	000	Counter of cuts NOK	0	pcs

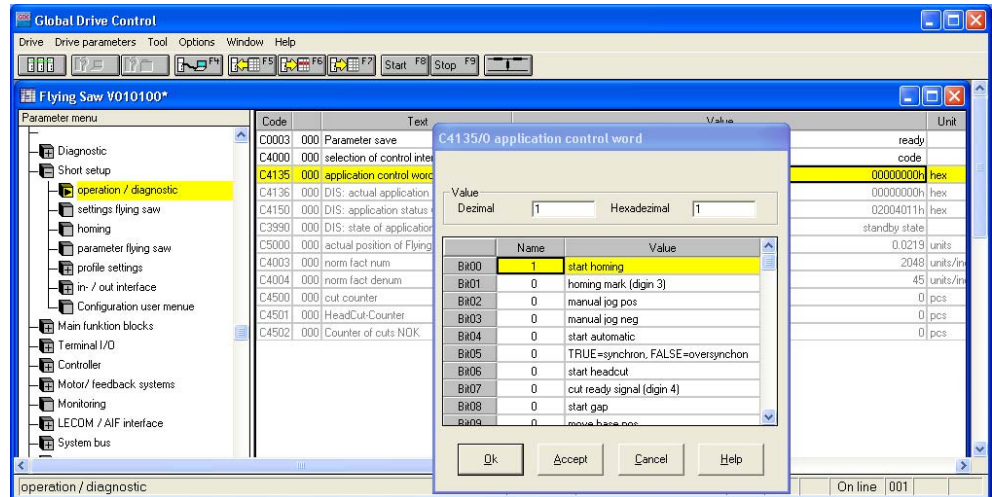
Name	Value
Bit00	0 start homing
Bit01	0 homing mark (digin 3)
Bit02	1 manual jog pos
Bit03	0 manual jog neg
Bit04	0 start automatic
Bit05	0 TRUE=synchron, FALSE=oversynchron
Bit06	0 start headcut
Bit07	0 cut ready signal (digin 4)
Bit08	0 start gap
Bit09	0 move base pos

If the drive does not move from the initial position in the direction of the cutting position during positive manual control, change code C3000/000 (motor mounting direction) so that the required position polarity is achieved.



Step 6

Test the homing function on the axis by setting the corresponding control bit in the application control word (start homing: bit 0).

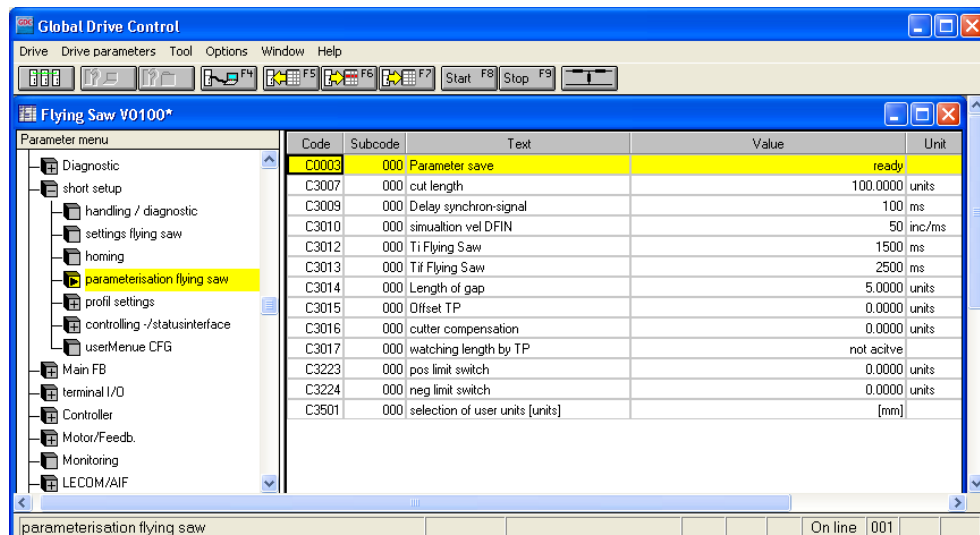


Possibly correct the settings for homing if the reference run does not proceed as desired.



Step 7

Now enter the parameters of the flying saw, which are required for length- and mark-controlled operation.



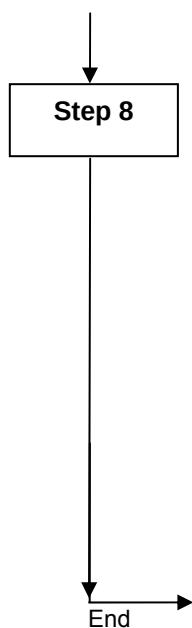
Set the following codes for setting the parameters for the flying saw:

- C3009/000 = Set here the delay time for the synchronised signal. The entry is made in [ms] and has the background that on axes with a large moment of inertia it may occur that a control process is still taking place to compensate for a possible following error as the synchronised signal is output.
- C3010/000 = Simulation velocity in the unit [inc/ms]
(This value is only processed if bit 20 in the application control word is set, otherwise input X9 is always processed as the velocity!)
- C3012/000 = The acceleration time describes the ramp during the period when the flying saw axis reaches master speed. The entry is made in the unit [ms].
- C3013/000 = The deceleration time describes the ramp during the period the axis needs to decelerate from the master speed to zero and then to reverse and move to the initial position.
- C3014/000 = If you want to create a gap in the material, you can define the length of the gap using this parameter. The entry is made in the unit [units].
- C3016/000 = With very "large" tools the width of the cut can be compensated. E.g. this setting is necessary for very wide and coarse saw blades, as a large amount of material is removed in the cut and this material is then missing from the length. The entry is made in the unit [units].
- C3223/000 = position of the positive software limit switch, entry in [units]
- C3224/000 = position of the negative software limit switch, entry in [units]



TIP!

To set the software limit switches you can move the drive using manual control to the required position and then apply the position from code C5000/000 in the codes for the software limit switches.



The commissioning of the length-controlled and mark-controlled operation is described in chapter 5 and 6.



TIP!

You will find information on the bit assignment for the application control word/status word as a function of C4000/000 in chapter 3.8.2.



5 Commissioning length-controlled operation



Attention!

The commissioning of the length-controlled operation of the Prepared Solution described in the following relates to the parameterisable variant.

Start

Step 1

Requirements:

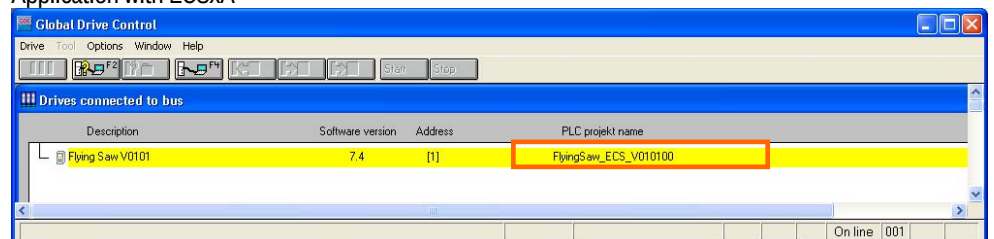
- The parameters for the servo controller for the flying saw axis have been specifically set to suit the specific case, as described in chapter 4.4.3.
- The axis is supplied with control voltage using terminals 59 (+24V DC) and 39 (0V) and with power using terminals L1, L2 and L3.

Initial situation:

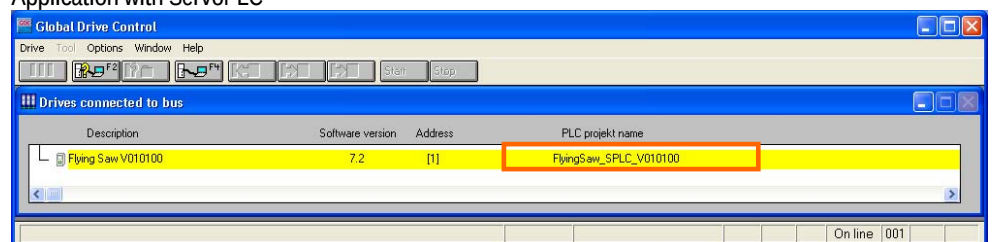
- The servo axis is not indicating any error.
- The servo axis is enabled.
- Application control word and application status word have not been specifically set using C4010/0xx and C4012/0xx.
- The axis has been successfully homed.

Start GDC and perform a search for drive controllers connected to the system bus. GDC finds the drive controllers connected. These include the drive controller loaded with the *FlyingSaw_SPLSC_V0100* project.

Application with ECSxA



Application with ServoPLC



Double-click the controller to be loaded with the *FlyingSaw_SPLC_V0100* application.

Step 2

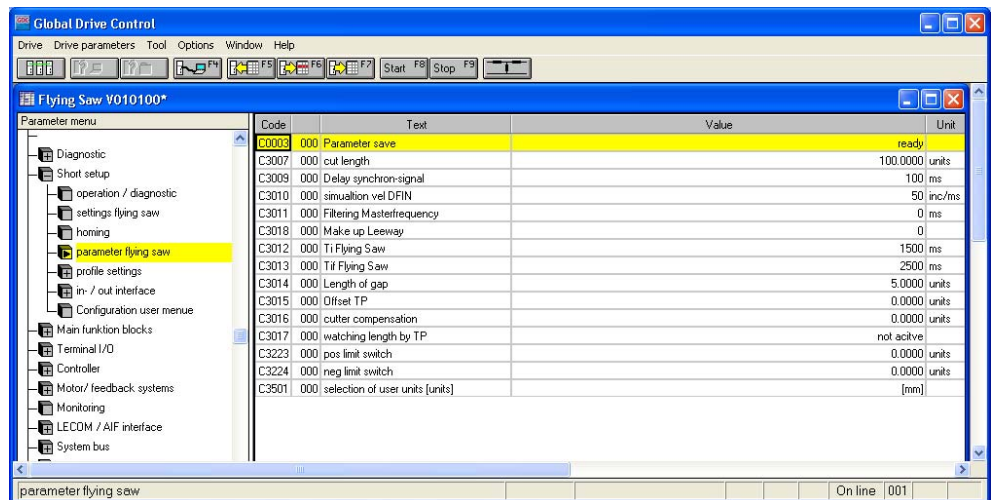
Now set the specific parameters of your flying saw application

FlyingSaw

Commissioning length-controlled operation



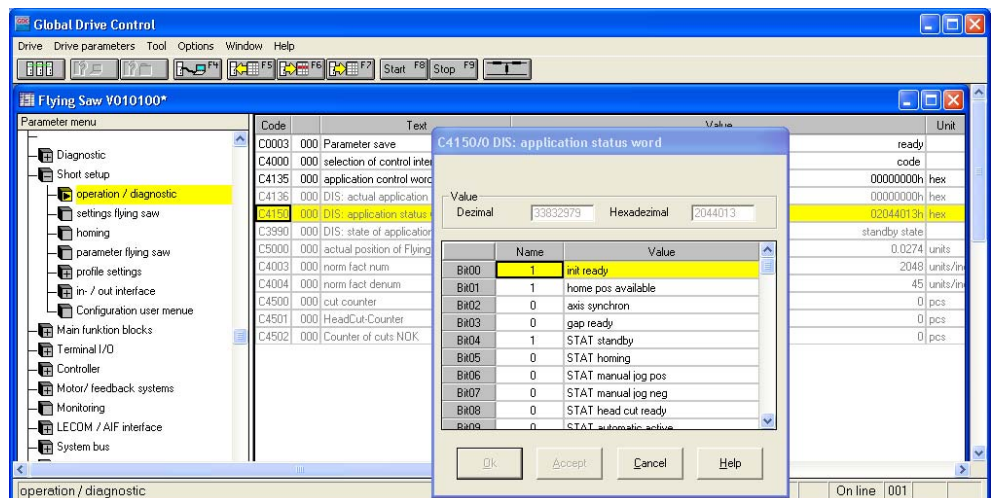
Step 3



You can describe your application for length-controlled operation using the following parameters:

C3007/000 Enter is this parameter the set length of the continuous material that you want to process. This parameter is entered in the unit [unit]

Now check in the application status word whether the bits for the starting conditions are set correctly.



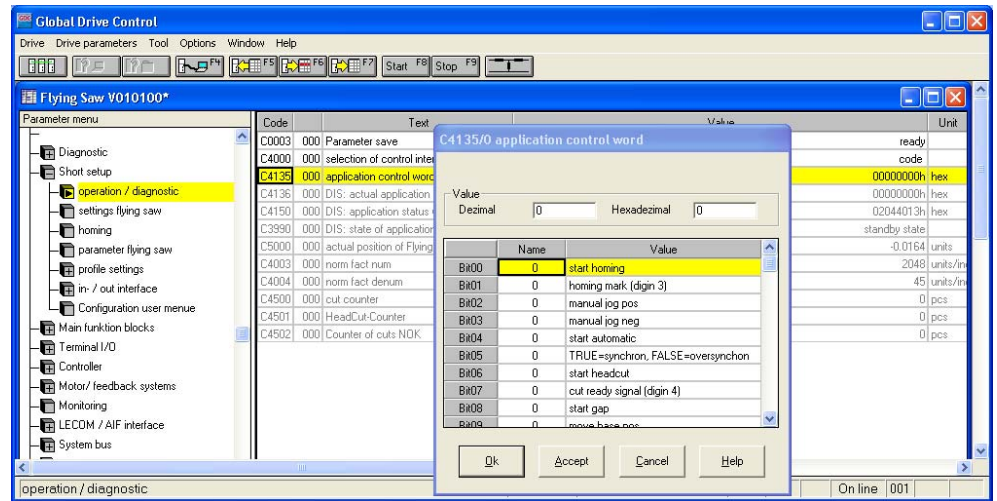
The following bits must be set for starting the length-controlled operation:

- Bit 00: Program initialisation complete.
- Bit 01: The axis home position is known.
- Bit 04: The axis is in the "Standby" state.
- Bit 14: The normalisation factor has been calculated.
- Bit 18: The axis is in the initial position.



Step 4

Now change over to the application control word.



To start the length-controlled operation, set the following bits in the order given:

Bit 04: Change to the "Flying saw mode" state

Bit 06: Perform a top cut

After bit 06 is set in the application control word, the axis synchronises to the line speed. The material is processed once synchronicity has been achieved.

Reset bit 06.

When the processing is complete and the flying saw axis is to move back to the initial position, activate bit 07. This "cut done signal" causes the axis to decelerate and subsequent positioning at the initial position.

By performing the top cut you have started length-controlled operation. All other start signals for synchronising are generated by the length calculator which calculates equidistant lengths.

If you want to stop operation, bit 04 must be reset. The axis is then back in the "Standby" state. A change to length-controlled operation can be made at any time as described in step 4.

End



6 Commissioning mark-controlled operation



Attention!

The commissioning of the Prepared Solution described in the following relates to the parameterisable variant.

Start

Step 1

Requirements:

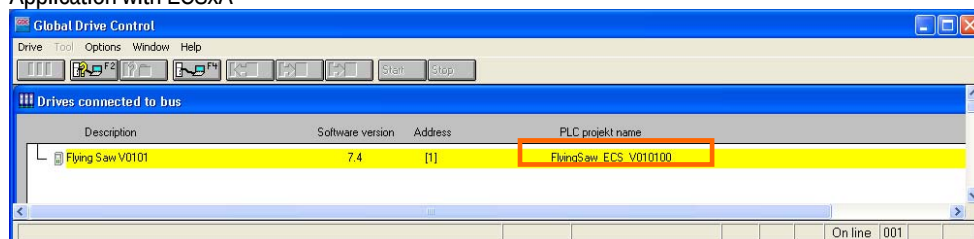
- The parameters for the servo controller for the flying saw axis have been specifically set to suit the specific case, as described in chapter 4.4.3.
- The axis is supplied with control voltage using terminals 59 (+24V DC) and 39 (0V) and with power using terminals L1, L2 and L3.

Initial situation:

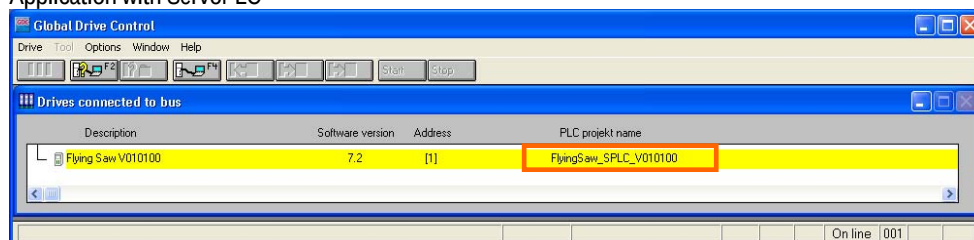
- The servo axis is not indicating any error.
- The servo axis is enabled.
- Application control word and application status word have not been specifically set using C4010/0xx and C4012/0xx.
- The axis has been successfully homed

Start GDC and perform a search for drive controllers connected to the system bus. GDC finds the drive controllers connected. These include the drive controller loaded with the *FlyingSaw_SPLC_V0100* project.

Application with ECSxA



Application with ServoPLC

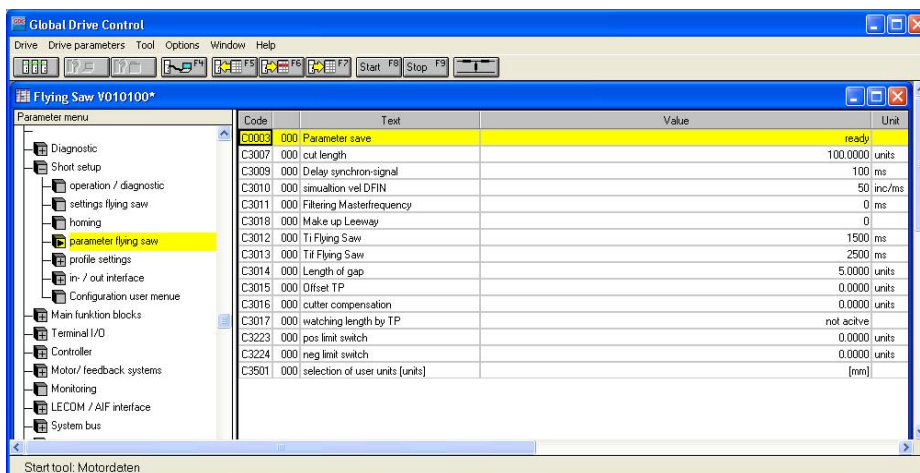


Double-click the controller to be loaded with the *FlyingSaw_V0100* application.



Step 2

Now set the specific parameters of your flying saw application.

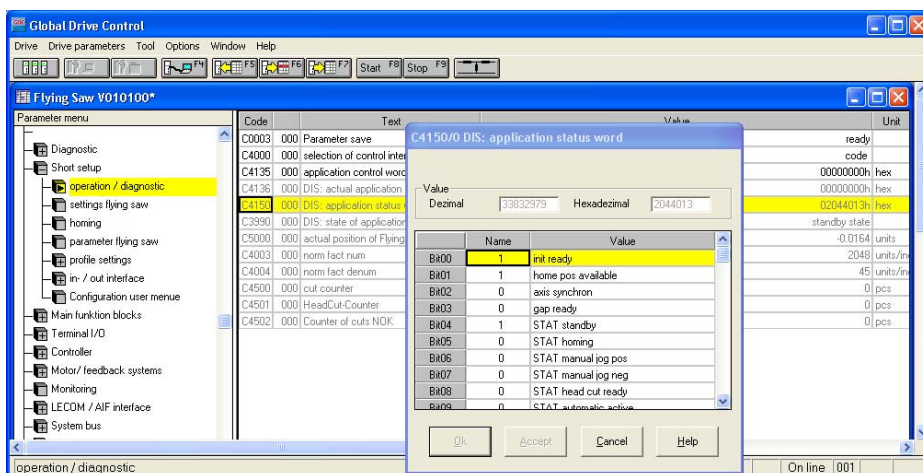


You can describe your application for length-controlled operation using the following parameters:

- C3007/000** In this parameter enter the set length. If a mark is not detected within the set length, the material is processed as per the set length set here. This parameter is entered in the unit [unit].
- C3015/000** Distance between a touch-probe detection and the initial position of the tool. The entry is made in the unit [units].
- C3017/000** Activation of the length monitoring during mark control. The monitoring can be activated or deactivated using this parameter.

Now check in the application status word whether the bits for the starting conditions are set correctly.

Step 3



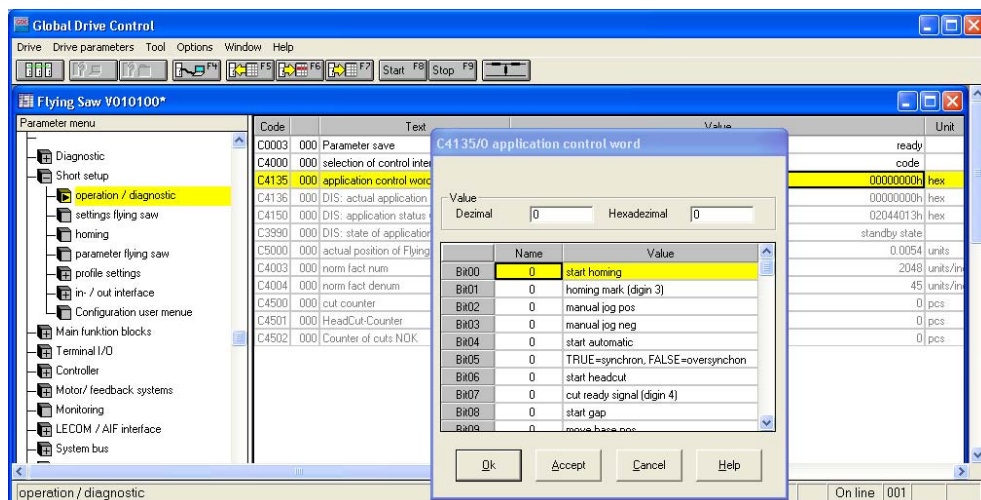
The following bits must be set for starting the length-controlled operation:

- Bit 00 Program initialisation complete.
- Bit 01: The axis home position is known.
- Bit 04: The axis is in the "Standby" state.
- Bit 14: The normalisation factor has been calculated.
- Bit 18: The axis is in the initial position.



Step 4

Now change over to the application control word.



To start the mark-controlled operation, set the following bits in the order given:

Bit 04: Change to the "Flying saw mode" state

Bit 10: Activate mark cuts

A top cut is not necessary in mark-controlled operation. The processing is started at the first mark on the material.

Once bit 10 is activated, a cut is made at the next mark detected.

If you want to leave mark-controlled operation again, then deactivate bit 10. The flying saw is then still in the "Flying saw mode".

The "Flying saw mode" state is left by resetting the bit 04.

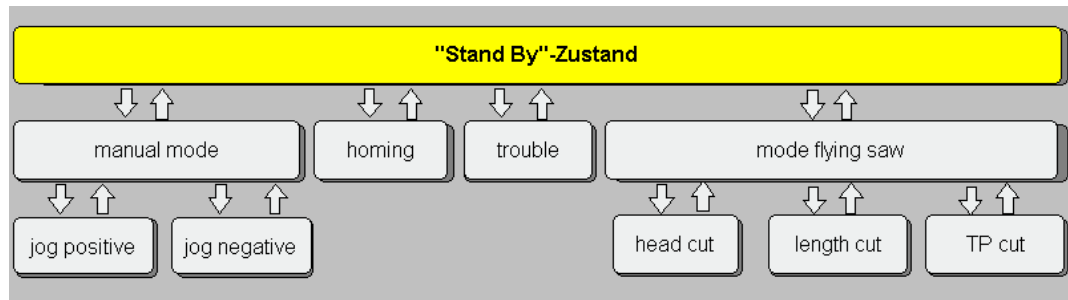
End



7 State machine of the Prepared Solution

7.1 Overview

The states for the Prepared Solution are represented by a so-called state machine. The state machine indicates a total of 11 states between which it is possible to change as follows:



The individual states are represented in the Prepared Solution using a global variable `g_wState` (ENUM) and also displayed to the operator using the code C3990/000.

Code	Possible settings:		Comments
	Default	Selection	
C3990	-	1: trouble 10: standby 20: manual mode 21: manual jog positive 22: manual jog negative 30: homing operation 40: flying saw operation 41: head cut 42: length cut 43: TP cut 50: positioning operation 100: initialisation	Error state Standby state Manual operation Positive manual control Negative manual control Homing active Automatic "flying saw" Top cut is being performed Length cut active Mark cut active Move to initial position Initialisation



TIP!

The code C3990/000 is permanently linked to the parameterisable variant in the GDC Monitor window such that the states of the Prepared Solution can be easily followed.



7.1.1 Concise description of the states

"Initialisation" state ($g_wState = 100$: initialisation)

During the initialisation phase, the normalisation factor necessary for the flying saw is calculated online. The calculation is only allowed in this state and in some circumstances requires a few minutes until the factor (rational fraction) is determined. The application then changes automatically to the *Standby* state.

On completion of initialisation, the *Program initialisation complete* bit is set in the application status word (default bit 00)

"Standby" state ($g_wState = 10$: standby)

The servo controller can be changed to all other states from the "Standby" state. The axis itself does not move when the "Standby" state is active.

A change is always made to the *initialisation* state if a new normalisation factor needs to be calculated. Re-calculation is triggered by changing the following codes:

- C1202 gearbox factor numerator
- C1203 gearbox factor denominator
- C1204 feed constant
- C3003 gearbox factor numerator of measuring wheel
- C3004 gearbox factor denominator of measuring wheel
- C3005 circumference of measuring wheel
- C3006 encoder pulses of measuring wheel

The *STATUS Standby* bit is set in the application status word (default bit 04).

"Manual operation" state ($g_wState = 20$: manual mode)

The "Manual operation" state is a transient state and is only adopted in the Prepared Solution for one cycle. As a rule after this state the state changes to the "Positive inching" or "Negative inching" state. The axis does not move when this state is active, but checks for various requirements for manual operation (e.g., whether a home position is known and whether software limit switches are to be used).

"Positive inching" state ($g_wState = 21$: manual jog positive)

The drive moves in the positive direction at the velocity set in the parameters in C3100/000 and the acceleration and deceleration set in the parameters in C3101/000 and C3102/000 respectively. During this process the positive software end position (C3223/000) is taken into account if the home position is known. The state is left on the clearing of the selection of "Positive inching", in case of controller inhibit, in case of quick stop or in case of an error in the drive system.

The *STATUS Inching in negative direction* bit is set in the application status word (default bit 06).

"Negative inching" state ($g_wState = 22$: manual jog negative)

The drive moves in the negative direction at the velocity set in the parameters in C3100/000 and the acceleration and deceleration set in the parameters in C3101/000 and C3102/000 respectively. During this process the negative software end position (C3224/000) is taken into account if the home position is known. The state is left on the clearing of the selection of "Negative inching", in case of controller inhibit, in case of quick stop or in case of an error in the drive system.

The *STATUS Inching in positive direction* bit is set in the application status word (default bit 07).



"Homing" state (*g_wState* = 30: homing operation)

If homing is activated using the global variable *g_bStartHomning*, this state is adopted. Ongoing homing is indicated by the global variable *g_bHomingBusy* = TRUE. If the home position is known (homing completed successfully), the drive sets the global variable *g_bHomePositionAvailable* = TRUE. The "Homing" state is left on reaching or defining the home position and the state changes to "Standby".

The *STATUS Homing active* bit is set in the application status word (default bit 05).

"Automatic flying saw" state (*g_wState* = 40: flyingsaw operation)

The "Automatic flying saw" state requires the home position to be known. Three other states branch out from this state; these three states describe the sequence of a flying saw.

The *STATUS Automatic selected* bit is set in the application status word (default bit 09).

"Top cut" state (*g_wState* = 41: headcut)

In the "Top cut" state the axis is performing a top cut.

The *STATUS Top cut is being made* bit is set in the application status word (default bit 08).

"Length cut" state (*g_wState* = 42: lengthcut)

This state indicates that a cut is being performed using the length calculator. The requirement for a length cut is a prior top cut; this cut starts the length calculator so that equidistant lengths can be calculated.

The *STATUS Length cut active* bit is set in the application status word (default bit 11).

"Mark cut" state (*g_wState* = 43: TPcut)

In the "Mark cut" state the axis waits at the initial position until a mark on the material has been detected and processed. The axis synchronises on the mark and once synchronicity is reached, processing is performed.

The *STATUS Mark cut active* bit is set in the application status word (default bit 12).

"Move to initial position" state (*g_wState* = 50: positioning_operation)

This state indicates that the axis is in the positioning mode and is moving to the initial position (home position).

"Trouble" state (*g_wState* = 1: trouble)

The "Trouble" state can be reached from any other state. The only transition criterion is the global variable *g_bGlobalError*. In the 'Trouble' state the target system is signalling a trip. As a result the system variables *DCTRL_bCinh_b* or *MCTRL_bQspln_b* = TRUE are set. It is only possible to change to the "Standby" state; this change is made when the global variable *g_bGlobalError* has been reset. This situation is possible when the cause of the error has been rectified and the error has been acknowledged using a positive edge on the global variable *g_bTripReset*.

The *STATUS Error detected* bit is set in the application status word (default bit 13).

7.2 Parallel functions

The software for the Prepared Solution provides the user with comprehensive error handling that can be expanded by programming (POU *ErrorHandling*, function block *ErrorHandling*). The



POU is called independent of the state machine and is therefore a process that runs in parallel to the state machine. If the error handling detects an error state, the interface to the state machine ensures the state machine changes to the 'Trouble' state. A reset to the 'Standby' state is only possible after the cause of the error has been rectified and the error acknowledged.

The POU *ErrorHandling* is not part of a library in the project, instead it is programmed externally in structured text as IEC code. In this way the existing error handling can be expanded with error messages defined by the user.



TIP!

You will find further information on handling errors in the section on the programmable variant of the Prepared Solution, chapter 8.3



8 Program extensions/supplements

Program extensions and supplements are possible if the programmable variant is used as the basis for the Prepared Solution. The program is edited using the Lenze software Drive PLC Developer Studio (DDS), version V2.2 or later. The chapters that follow require basic knowledge of the IEC programming languages and the configuration of programmable logic controllers.

8.1 Configuration of the ServoPLC user interface

8.1.1 Default setting of the ServoPLC hardware inputs

Digital inputs (X6)		
Terminal	Signal	Value/meaning
28		Controller enable
		0V Inhibit servo controller
		24V Enable servo controller
E1	g_bLimitSwitchPos	Signal from the positive limit switch
		0V Positive limit switch actuated
		24V Positive limit switch not actuated
E2	g_bLimitSwitchNeg	Signal from the negative limit switch
		0V Negative limit switch actuated
		24V Negative limit switch not actuated
E3	g_bHomingMark	Signal from the homing switch
		The signal is only relevant for the homing modes 0, 1, 2, 3, 4 and 5 – you will find further information on the individual homing modes in chapter 2.12.1
		24V => 0V Homing switch
E4	g_bCutReady	Cut done signal
		The signal is sent from the processing tool to the application. The tool is then returned to the initial position
		0V => 24V Cut done signal
E5	DFIN_bTPReceived_b	Touch probe input for mark-controlled synchronisation. Note: The code C0428/000 must always be set to the value 1 so that the mark is detected using input E5
Analog inputs (X6)		
Terminal	Signal	Value/meaning
1, 2		(not used)
3, 4		(not used)
State bus		
Terminal	Signal	Value/meaning
ST		(not used)
Master frequency input (X9)		
Terminal	Signal	Value/meaning
X9	DFIN_nIn_v	Input of the master speed as phase difference signal: The number of increments in the signal present on X9 can be set in the parameters using code C0425/000, however a value for C0425/000 = 6 (16384[incr.rev]) should always be used to ensure the greatest possible resolution for the master position.



8.1.2 Default setting of the ServoPLC hardware outputs

Digital outputs (X6)			
Terminal	Signal	Value/meaning	
A1	DCTRL_bCInh_b	Controller inhibit	
		0V	Controller is inhibited
		24V	Controller is enabled
A2	g_bHomePositionAvailable	Home position known	
		0V	No home position known
		24V	Home position known
A3	g_bAxisSynchron	Control of the processing tool, e.g. a saw that starts processing when this signal is switched	
		0V	Axis not synchronised to the master speed
		24V	Axis is synchronised to the master speed
A4	g_bGlobalError	Output of the error state	
		0V	Drive in correct working order
		24V	Drive has an error
Analog outputs (X6)			
Terminal	Signal	Value/meaning	
62		(not used)	
63		(not used)	
State bus			
Terminal	Signal	Value/meaning	
ST		(not used)	
Master frequency output (X10)			
Terminal	Signal	Value/meaning	
X10		(not used)	



8.2 Configuration of the ECS user interface

8.2.1 Default setting of the ECS hardware inputs

Digital inputs (X6)			
terminal	signal	value/meaning	
28		Controller enable	
		0V	Inhibit servo controller
		24V	Enable servo controller
E1	DFIN_bTPReceived_b	Touch probe input for mark-controlled synchronisation. Note: The code C0428/000 must always be set to the value 1 so that the mark is detected using input E1	
E2	g_bHomingMark	Signal from the homing switch The signal is only relevant for the homing modes 0, 1, 2, 3, 4 and 5 – you will find further information on the individual homing modes in chapter 2.12.1	
		24V => 0V	Homing switch
E3	g_bLimitSwitchPos	Signal from the positive limit switch	
		0V	Positive limit switch actuated
		24V	Positive limit switch not actuated
E4	g_bLimitSwitchNeg	Signal from the negative limit switch	
		0V	Negative limit switch not actuated
		24V	Negative limit switch not actuated
Analog inputs (X6)			
terminal	signal	value/meaning	
1, 2		(not used)	
3, 4		(not used)	
Master frequency input (X8)			
terminal	signal	value/meaning	
X8	DFIN_nIn_v	Input of the master speed as phase difference signal: The number of increments in the signal present on X8 can be set in the parameters using code C0425/000, however a value for C0425/000 = 6 (16384[incr./rev]) should always be used to ensure the greatest possible resolution for the master position.	



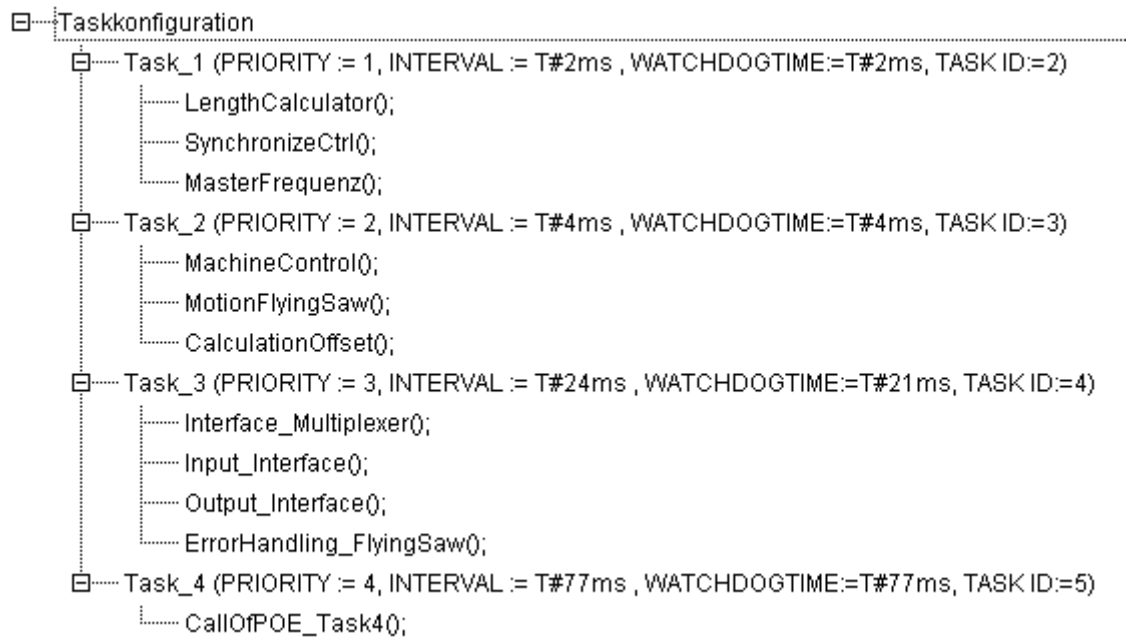
8.2.2 Default setting of the ECS hardware outputs

Digital outputs (X6)			
terminal	signal	value/meaning	
A1	DCTRL_bClnh_b	Reglersperre	
		0V	Controller is inhibited
		24V	Controller is enabled
Analog outputs (X6)			
terminal	signal	value/meaning	
62		(not used)	
63		(not used)	
Master frequency output (X8)			
terminal	signal	value/meaning	
X8		(not used)	



8.3 Task management

In the programmable variant of the Prepared Solution the different functionalities are called in different tasks as a function of the processing priority. The following tasks have already been added:



Task 1 contains the main core of the application. This task must always be processed with the highest priority, even if the user adds further tasks. It is called at 2ms intervals.

Task 2 calls the machine sequence in which the axis control is applied. In addition calculations are made and the master frequency evaluated using the normalisation factor.

Task 3 contains the multiplexer and the error management.



TIP!

The user can add further tasks and call custom programs in these additional tasks. Here the following conditions are to be observed:

- No task is allowed to have a higher priority and a shorter interval time than task 1.
- No user tasks are allowed to have a higher priority and/or a shorter interval time than task 2.
- Further networks can be added to the existing program blocks (e.g. POU *MotionControl*), as long as the processing times for the task called do not cause any task overflow.



9 Dimensioning aspects

9.1 Resolution of the system

In order to determine the resolution, the relationship between measuring wheel and motor must be considered.

In the following, the resolution is taken into consideration:

Measuring wheel parameters:

- Encoder constant [inc/rev]
- Measuring wheel constant [units/rev]

$$\text{encoder resolution}[\text{inc}/\text{units}] = \frac{\text{encoder constant}[\text{inc}/\text{umdr}]}{\text{measuring wheel constant}[\text{units}/\text{umdr}]}$$

Drive parameters:

- Feed constant [units/r]
- Gearbox factor numerator
- Gearbox factor denominator

$$\text{motor resolution}[\text{umdr}/\text{units}] = \frac{\text{gearbox factor numerator}}{\text{gearbox factor denominator} * \text{feedconstant}[\text{units}/\text{umdr}]}$$

Total resolution:

$$\text{resolution}[\text{inc}/\text{umdr}] = \frac{\text{motor resolution}[\text{umdr}/\text{units}]}{\text{encoder resolution}[\text{inc}/\text{units}]}$$



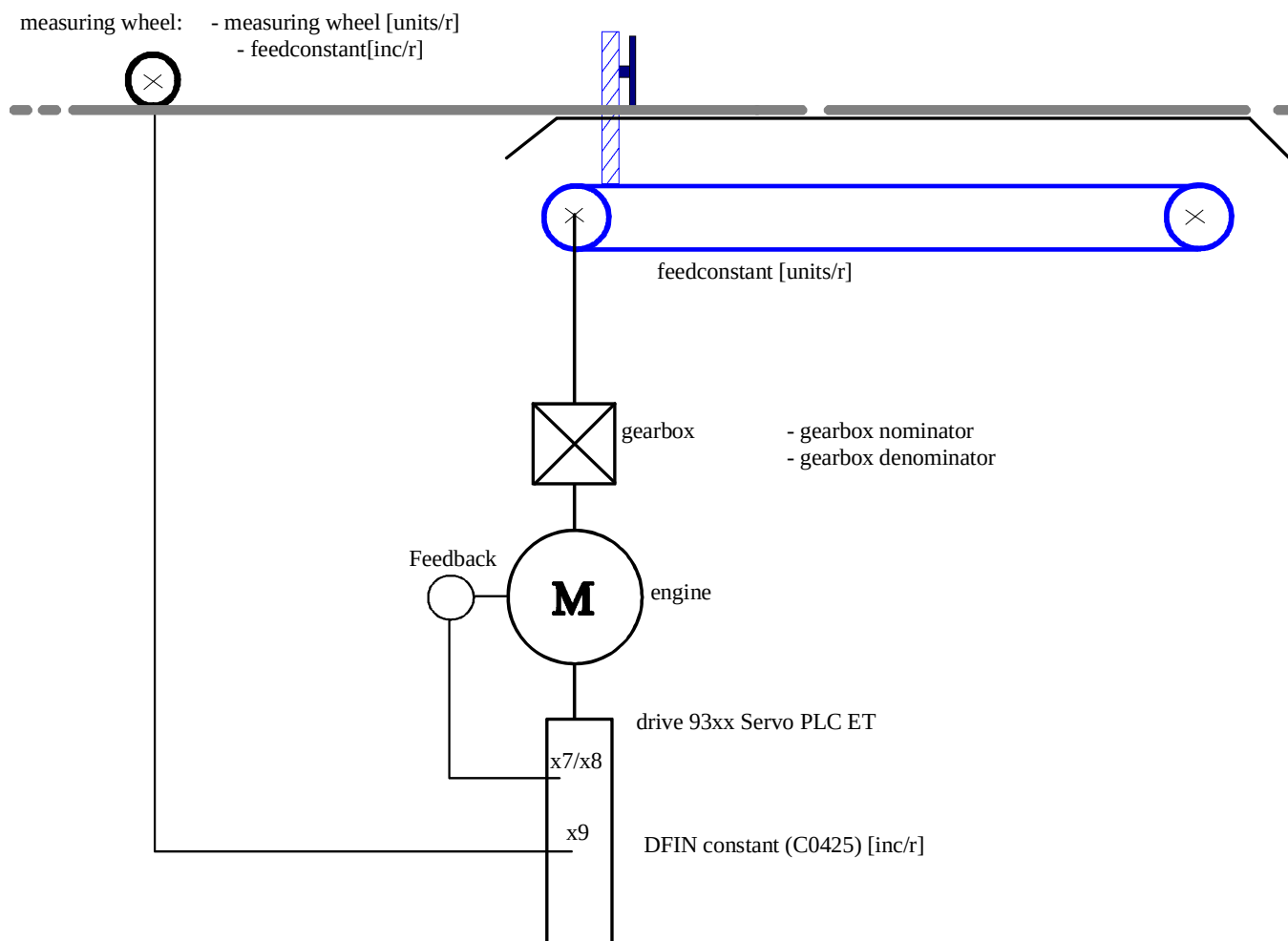
9.2 Axis normalisation

For the normalisation of the flying saw, two concepts for a master frequency source will be considered.

1. Master frequency source measuring wheel
2. Master frequency source Servo / Servo PLC (master frequency coupling)

9.3 Master frequency source measuring wheel

Process example with measuring wheel for master frequency determination





Normalisation is calculated as follows:

Parameters for the normalisation:

- **Measuring wheel:** *measuring wheel constant (C3005/000) [units/r]*
 Encoder constant (C3006/000) [inc/r]



TIP!

Measuring wheel constant: e.g. a measuring wheel with $d = 250\text{mm}$, measuring wheel constant $= 250\text{mm} \cdot \pi = 785.4\text{mm/r}$

- **Mechanical system:** *Feed constant (C1204/000) [units/r]*
 Gearbox
 Gearbox factor numerator (C1202/000)
 Gearbox factor denominator (C1203/000)
- **Controller:** *DFIN master frequency constant (C0425/000) [inc/r]*



TIP!

The normalisation factor is automatically calculated in the application and entered in the codes C3003/000 and C3004/000.



Parameters for normalising ***controller 2***:

- Gearbox factor numerator (C1202/000)
- Gearbox factor denominator (C1203/000)
- Feed constant (C1204/000) [units/r]
- DFIN master frequency constant (C0425/000) [inc/r]

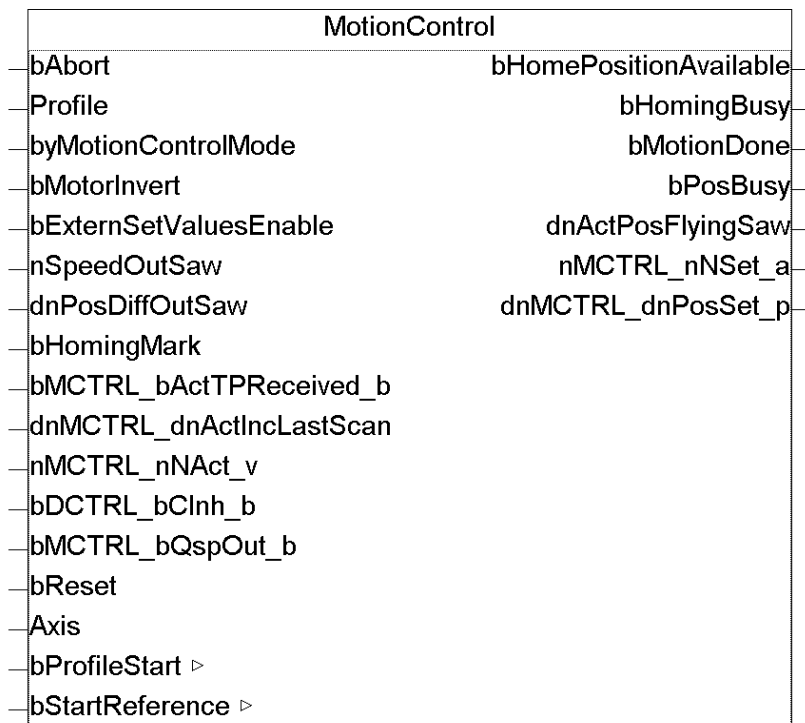
10 Description of the function blocks

10.1 Function block *MotionControl*

In the FlyingSaw application the function block *MotionControl* undertakes the positioning tasks and the referencing.

During synchronisation to the material speed the setpoints are fed through and switched to the interface MCTRL.

The function block *MotionControl* represents the interface between application and drive control (MCTRL).



Task information

Can be called in:	☐ Cyclic task	☒ Time-controlled task (INTERVAL)	☐ Event-controlled task (EVENT)	☐ Interrupt task
-------------------	--------------------------------------	---	--	---

FlyingSaw

Description of the function blocks

Inputs (Variable type: VAR_INPUT)		
Identifier	Data type	Value/meaning
bAbort	BOOL	This input triggers the cancelling of positioning. The input is handled within the application in the MachineControl. The flying saw is positioned in the initial position during return movement. FALSE -> TRUE Cancel the positioning
Profile	DINT	Positioning profile ➤ Data structure; the elements of the structure describe the complete profile
byMotionControlMode	DINT	Positioning mode ➤ In the application absolute positioning is performed during movement to the initial position and constant travel during manual control. Positioning modes used: 0 Absolute positioning 2 Continuous constant travel
bMotorInvert	BOOL	Entry of the motor mounting direction FALSE not inverted TRUE inverted
bExternSetValuesEnable	BOOL	Switch to external setpoints for synchronisation ➤ The external setpoints are looped through 1:1 to the related outputs.
nSpeedOutSaw	INT	Setpoint for the speed for the synchronisation ➤ For this setpoint to be actively switched to the interface, the variable <i>bExternSetValuesEnable</i> must be set to TRUE. ➤ <i>nSpeedOutSaw</i> -> <i>MCTRL_nNSet_a</i>
dnPosDiffOutSaw	DINT	Setpoint for the positional deviation for the synchronisation ➤ For this setpoint to be actively switched to the interface, the variable <i>bExternSetValuesEnable</i> must be set to TRUE. ➤ <i>dnPosDiffOutSaw</i> -> <i>MCTRL_dnPosSet_p</i>
bHomingMark	BOOL	Homing switch
bMCTRL_bActTPReceived_b	BOOL	Input for the MCTRL touch probe
dnMCTRL_dnActIncLastScan	DINT	Phase difference in [inc] between latching time and starting time for the task
nMCTRL_nNAct_v	INT	Actual speed for the actual phase integrator ➤ The input is to be connected to the system variable <i>MCTRL_nNAct_v</i> (speed value for the feedback system) in SB MCTRL.
bDCTRL_bCInh_b	BOOL	Signal for the controller inhibit ➤ The input is to be connected to the system variable <i>DCTRL_bCInh_b</i> in SB DCRTL.

Inputs (Variable type: VAR_INPUT)		
Identifier	Data type	Value/meaning
bMCTRL_bQspOut_b	BOOL	Drive is in quick stop ➤ This input is to be connected to the system variable <i>MCTRL_bQspOut_b</i> in SB MCTRL.
bReset	BOOL	Reset positioning
Axis	AXIS_REF	Machine parameters ➤ Data structure with elements that contain the machine parameters in the internal measuring system. ➤ Connect this input to a global variable written by the FB L_MCMachineData.

FlyingSaw

Description of the function blocks

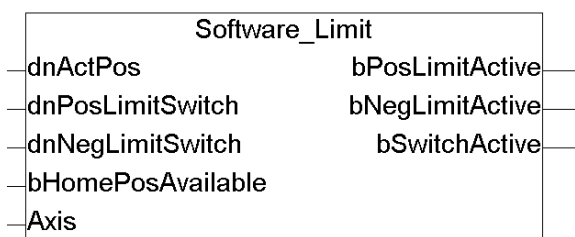
Inputs outputs (Variable type: VAR_IN_OUT)		
Identifier	Data type	Value/meaning
bProfileStart	BOOL	Start a positioning This input signal is automatically reset.
bStartReference	BOOL	Start homing This input signal is automatically reset

Outputs (Variable type: VAR_OUTPUT)		
Identifier	Data type	Value/meaning
bHomePositionAvailable	BOOL	Status signal "Homing done" ➤ The signal is TRUE on successful completion of homing
bHomingBusy	BOOL	Status signal "Homing active" ➤ This signal switches to TRUE when homing is active
bMotionDone	BOOL	Status signal "Positioning has been performed"
bPosBusy	BOOL	Status signal "Positioning is active"
dnActPosFlyingSaw	DINT	Actual phase position of the axis "flying saw" ➤ 65536 inc correspond to one revolution on the motor end
nMCTRL_nNSet_a	INT	Speed setpoint ➤ This output is to be connected to the system variable MCTRL_nNSet_a in SB MCTRL
dnMCTRL_dnPosSet_p	DINT	Positional deviation direct for the position controller ➤ This output is to be connected to the system variable MCTRL_dnPosSet_p in SB MCTRL. The phase controller must be switched active (MCTRL_bPosOn_b = TRUE)

10.2 Function block Software_Limit

This function block monitors the software limit positions set. The block is only active after homing has been performed. This means that during homing only the hardware limit positions (physical limit switches) are actively monitored.

The position entry for the positive and negative software limit switches must not be equal to zero and be within the possible limits so that the software limit positions are monitored.



Task information

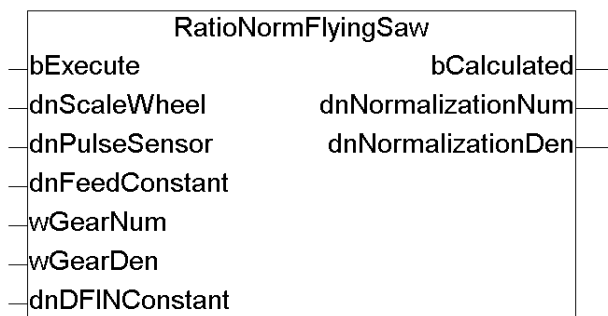
Can be called in:	☐ Cyclic task	☒ Time-controlled task (INTERVAL)	☐ Event-controlled task (EVENT)	☐ Interrupt task
-------------------	--------------------------------------	---	--	---

Inputs (Variable type: VAR_INPUT)		
Identifier	Data type	Value/meaning
dnActPos	DINT	Actual phase position of the axis "flying saw" ➤ 65536 inc correspond to one revolution on the motor end
dnPosLimitSwitch	DINT	Input for the position of the positive software limit switch ➤ The entry is made in the application unit [units] and must have the value > 0. The reference here is the home position. ➤ The software limit position is only active for an entry > 0.
dnNegLimitSwitch	DINT	Input for the position of the negative software limit switch ➤ The entry is made in the application unit [units] and must have a value <0. The reference here is the home position. ➤ The software limit position is only active for an entry <0.
bHomePosAvailable	BOOL	Input for the signal "Homing done" ➤ The software limit position monitoring is only activated once the home position is known
Axis	MC_AXIS_REF	Machine parameters ➤ Data structure with elements that contain the machine parameters in the internal measuring system. ➤ Connect this input to a global variable written by the FB L_MCMachineData.

Outputs (Variable type: VAR_OUTPUT)		
Identifier	Data type	Value/meaning
bPosLimitActive	BOOL	Status signal "Positive software limit position reached"
bNegLimitActive	BOOL	Status signal "Negative software limit position reached"
bSwitchActive	BOOL	Status signal "Software limit position monitoring active"

10.3 Function block *RatioNormFlyingSaw*

This function block calculates the normalisation factor for the flying saw as a function of the machine parameters. The calculation is only allowed to be made in the cyclic task, as fractional rational numbers are processed.



Task information

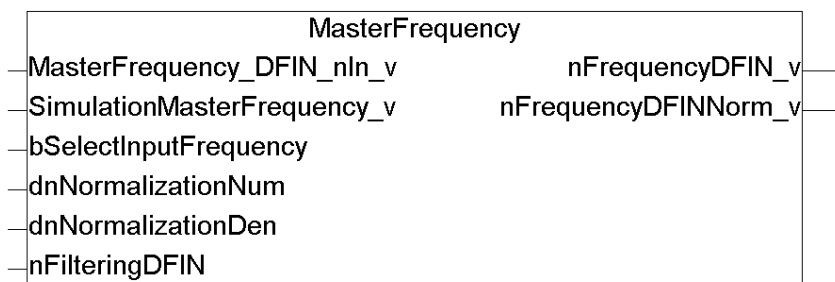
Can be called in:	☒ Cyclic task	☐ Time-controlled task (INTERVAL)	☐ Event-controlled task (EVENT)	☐ Interrupt task
-------------------	---	--	--	---

Inputs (Variable type: VAR_INPUT)		
Identifier	Data type	Value/meaning
bExecute	BOOL	Perform calculation ➤ If the factor is to be calculated, this input must be activated FALSE -> TRUE Triggers a calculation
dnScaleWheel	DINT	Circumference of the measuring wheel ➤ 10000 = 1.0000 [unit]
dnPulseSensor	DINT	Number of pulses from the encoder at the measuring wheel ➤ 10000 = 1.0000 [inc/rev]
dnFeedConstant	DINT	Feed constant ➤ 10000 = 1.0000 [unit/rev]
wGearNum	WORD	Gearbox factor numerator
wGearDen	WORD	Gearbox factor denominator
dnDFINConstant	DINT	DFIN master frequency constant (C0425/000) ➤ Entry is made in [inc/rev]

Outputs (Variable type: VAR_OUTPUT)		
Identifier	Data type	Value/meaning
bCalculated	BOOL	Status signal "Factor calculated"
dnNormalizationNum	DINT	Value for the numerator
dnNormalizationDen	DINT	Value for the denominator

10.4 Function block *Master Frequency*

This function block prepares the master frequency for the flying saw and during this process takes into account important factors like the normalisation factor for the flying saw. It is also possible to filter the input signal and to simulate the master frequency.



Task information

Can be called in:	☐ Cyclic task	☒ Time-controlled task (INTERVAL)	☐ Event-controlled task (EVENT)	☐ Interrupt task
-------------------	--------------------------------------	--	---	---

Inputs (Variable type: VAR_INPUT)		
Identifier	Data type	Value/meaning
MasterFrequency_DFIN_nIn_v	INT	This input is connected to the signal DFIN_nIn_v from the system controller. The master value is acquired in inc/ms and processed in the function block.
SimulationMasterFrequency_v	INT	A master value can be simulated with this variable. The master value is also acquired in inc/ms and processed in the function block. The simulation enables the functions to be tested without receiving a fixed master value from a master. The entry is made in [inc/ms]
bSelectInputFrequency	BOOL	This input makes it possible to switch between the master value simulation and the signal DFIN
		FALSE The master value from the input MasterFrequency_DFIN_nIn_v is used
		TRUE The master value from the input
dnNormalizationNum	DINT	Numerator for the normalisation factor. The normalisation factor is calculated in the function block <i>RatioNormFlyingSaw</i> .
dnNormalizationDen	DINT	Denominator for the normalisation factor. The normalisation factor is calculated in the function block <i>RatioNormFlyingSaw</i> .
nFilteringDFIN	INT	Master value filtering. The filter time is defined in the unit [ms]. Only the output signal <i>nFrequencyDFINNorm_v</i> is filtered.

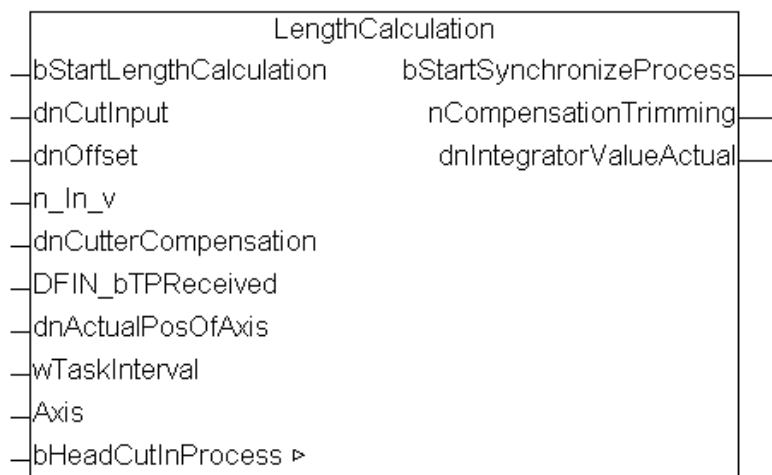
FlyingSaw

Description of the function blocks

Outputs (Variable type: VAR_OUTPUT)		
Identifier	Data type	Value/meaning
nFrequencyDFIN_v	INT	Output of the frequency for the master value. In the simulation mode the value defined on the input <i>SimulationMasterFrequency_v</i> is output. Otherwise the master value from the input <i>MasterFrequency_DFIN_nIn_v</i>. The output variable is also evaluated using the filter time from the input <i>nFilteringDFIN</i>. The filtering is only active if the master value is defined using input X9 on the servo axis. The filter time is not taken into account in the simulation.
nFrequencyDFINNorm_v	INT	Output of the normalised master frequency. In addition to the output variable <i>nFrequencyDFIN_v</i> this output is adapted to the axis parameters using the normalisation factor. The filtering is only active if the master value is defined using input X9 on the servo axis. The filter time is not taken into account in the simulation.

10.5 Function block *LengthCalculation*

Based on the length entered, the block *LengthCalculation* calculates the start signals for the synchronisation to the master speed.



Task information

Can be called in:	☐ Cyclic task	☒ Time-controlled task (INTERVAL)	☐ Event-controlled task (EVENT)	☐ Interrupt task
-------------------	--------------------------------------	---	--	---

Inputs (Variable type: VAR_INPUT)		
Identifier	Data type	Value/meaning
bStartLengthCalculation	BOOL	Start the length calculator
		TRUE Start the length calculator
		FALSE Stop the length calculator. The length calculator is initialised with zero internally.
dnCutInputInc	DINT	Input for specifying the set length. The set length is defined in increments.
dnOffset	DINT	Input for the offset. The offset is calculated in the function block <i>OffsetCalculation</i> and must be connected to this input.
n_In_v	INT	Master frequency specified in the unit [inc/ms]. The master frequency is calculated using the normalisation factor and must be connected to the output of the function block <i>MasterFrequency</i> .
dnCutterCompensation	DINT	Input for width of cut compensation. The width of cut is defined in the application unit [unit].
DFIN_bTPReceived	BOOL	Input for the DFIN touch probe. The variable is to be connected to the input DFIN_bTPReceived_b on the SB DFIN
dnActualPosOfAxis	DINT	Input for the actual position of the flying saw. If a top cut is performed away from the initial position, this position is added to the calculation and taken into account on the next cut.
wTaskInterval	WORD	Task interval. This variable must be linked to System_Flag SYSTEM_wTaskInterval
Axis	MC_AXIS_REF	Machine parameters ➤ Data structure with elements that contain the machine parameters in the internal measuring system. ➤ Connect this input to a global variable written by the FB L_MCMachineData

FlyingSaw

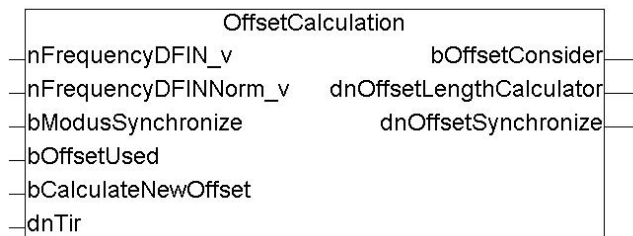
Description of the function blocks

Inputs/outputs (Variable type: VAR_IN_OUT)		
Identifier	Data type	Value/meaning
bHeadCutInProcess	BOOL	Top cut in process. The length calculator is initialised with actual values by triggering this variable. The variable is automatically reset in the block.

Outputs (Variable type: VAR_OUTPUT)		
Identifier	Data type	Value/meaning
bStartSynchronizeProcess	BOOL	"Start synchronisation" signal. The length has been reached and the synchronising process starts.
nCompensationTrimming	INT	Correction that is applied to the setpoint. In this way an error caused by the system propagation time is compensated.
dnIntegratorValueActual	DINT	Output of the actual length.

10.6 Function block *Offset Calculation*

This block calculates the offset, or the offset increments for the block **SynchronizeControl** so that the flying saw is never moved faster than the master speed in synchronous operation.



Task information

Can be called in:	☐ Cyclic task	☒ Time-controlled task (INTERVAL)	☐ Event-controlled task (EVENT)	☐ Interrupt task
-------------------	--------------------------------------	---	--	---

Inputs (Variable type: VAR_INPUT)		
Identifier	Data type	Value/meaning
nFrequencyDFIN_v	INT	Input frequency for the master frequency. The master frequency is the basis for the calculation of the offset increments. This input variable is connected to the function block MasterFrequency.
nFrequencyDFINNorm_v	INT	Normalised input frequency for the master frequency. The master frequency is the basis for the calculation of the offset increments. This input variable is connected to the function block MasterFrequency.
bModusSynchronize	BOOL	Here the mode for the flying saw is defined. In principle a differentiation is made between two modes. The "Synchronous" mode in which the saw blade is never faster than the master value and the "Oversynchronous" mode is which the saw unit briefly "overtakes" the master value on starting until all increments have been caught up. FALSE Flying saw "Oversynchronous" mode
bOffsetUsed	BOOL	Information that the offset has been taken into account in the cut. This variable must be handled in the machine sequence. FALSE Offset for an actual cut has not yet been used.
bCalculateNewOffset	BOOL	Information that a new offset must be calculated for a subsequent cut. This variable must be handled in the machine sequence. FALSE Do not calculate a new offset
dnTir	DINT	Saw unit acceleration time. This variable is to be connected to the function block SynchronizeControl from the library FlyingSawV0100.

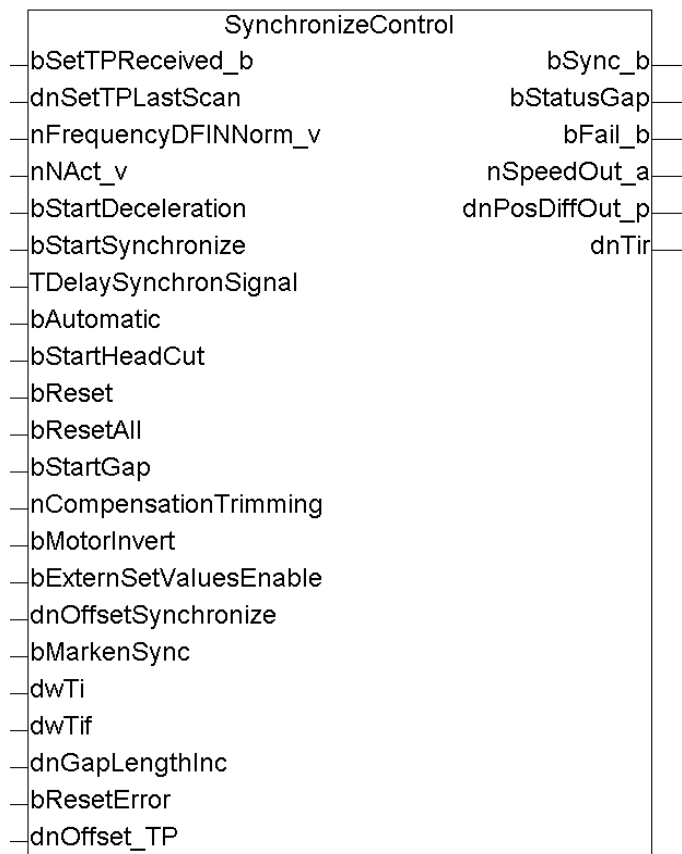
FlyingSaw

Description of the function blocks

Inputs (Variable type: VAR_OUTPUT)		
Identifier	Data type	Value/meaning
bOffsetConsider	BOOL	Offset for the length calculator is to be taken into account. The variable must be connected to the block LengthCalculator from the library FlyingSawV0100.lib.
dnOffsetLengthCalculator	DINT	Offset for the length calculator. The variable is to be connected to the block LengthCalculator from the library FlyingSawV0100.lib.
dnOffsetSynchronize	DINT	Offset for the synchronisation process. The variable is to be connected to the block SynchronizeControl from the library FlyingSawV0100.lib.

10.7 Function block *Synchronize Control*

The main function of the flying saw is undertaken by this block. Among other aspects the phase and speed-dependent synchronisation are handled and the setpoint passed to the block ***MotionControl***.



Task information

Can be called in:	☐ Cyclic task	☒ Time-controlled task (INTERVAL)	☐ Event-controlled task (EVENT)	☐ Interrupt task
-------------------	--------------------------------------	---	--	---

FlyingSaw

Description of the function blocks

Inputs (Variable type: VAR_INPUT)		
Identifier	Data type	Value/meaning
bSetTPReceived_b	BOOL	Input for the touch-probe. This variable is connected to the system variable <i>DFIN_bTPReceived_b</i> from the system block <i>DFIN</i> .
dnSetTPLastScan	DINT	Phase difference between latch point and task start time. This variable is connected to the system variable <i>DFIN_dnIncLastScan_p</i>
nFrequencyDFINNorm_v	INT	Input for the normalised master frequency. This variable is connected to the variable <i>nFrequencyDFIN_v</i> from the function block <i>MasterFrequency</i> in the library <i>FlyingSawV0100.lib</i> .
nNAct_v	INT	Actual speed for the actual phase integrator. This variable is connected to the system variable <i>MCTRL_nNAct_v</i> from the system block <i>MCTRL</i> .
bStartDeceleration	BOOL	Using this variable the saw unit is decelerated after a cut and then positioned at the home position. The variable is processed logically in the machine sequence and is typically activated by the sawing action itself.
		FALSE No deceleration of the saw unit
bStartSynchronize	BOOL	Start a synchronisation process. The input is connected to the variable <i>bStartSynchronizprozess</i> from the function block <i>LengthCalculation</i> in the library <i>FlyingSawV0100.lib</i> .
		FALSE No start
TDelaySynchronSignal	TIME	Delay for the synchronised signal. In this way settling after the synchronising process is taken into account. The setting is made in the unit [ms].
bAutomatic	BOOL	Input information that automatic operation has started. This input is to be connected to the status automatic.
bStartHeadCut	BOOL	Start signal for performing a top cut
bReset	BOOL	The integrators for the synchronising process are reset when the variable is set.
bResetAll	BOOL	In addition to <i>bReset</i> , when this variable is set the positional deviation is set to zero.
bStartGap	BOOL	Start signal for making a gap in the material
nCompensationTrimming	INT	Correction that is added to the setpoint. This correction is used to compensate for an error caused by the system propagation time. This input is to be connected to the block <i>LengthCalcualtion nCompensationTrimming</i>
bMotorInvert	BOOL	Entry of the motor mounting direction
		FALSE not inverted TRUE inverted
bExternSetValuesEnable	BOOL	Switch to external setpoints for synchronisation ➤ The setpoints from the synchronising process are looped through 1:1 to the related outputs.
dnOffsetSynchronize	DINT	Offset for "synchronous" synchronisation. The value is calculated online by the function block <i>OffsetCalculation</i> . This input variable must be connected to the function block <i>OffsetCalculation</i> .

FlyingSaw

Description of the function blocks

Inputs (Variable type: VAR_INPUT)			
Identifier	Data type	Value/meaning	
bMarkenSync	BOOL	Activation of the mark synchronisation.	
		TRUE	Mark synchronisation active
		FALSE	Mark synchronisation not active
dwTi	DWORD	Flying saw acceleration time. The entry is made in the unit [ms]	
dwTif	DWORD	Flying saw deceleration time. The entry is made in the unit [ms]	
dnGapLengthInc	DINT	Length of the gap. The entry is made in the unit [inc]	
bResetError	BOOL	In case of an error, the block and as a result all active integrators are reset. The variable must be included in the application's reset handling.	
dnOffset_TP	DINT	Distance between the touch probe signal detected and the initial position of the flying saw. The entry is made in the unit [inc]	

Inputs (Variable type: VAR_OUTPUT)		
Identifier	Data type	Value/meaning
bSync_b	BOOL	Status signal "Axis synchronised". The signal is output when the axis is moving in phase and speed synchronism with the master value.
bStatusGap	BOOL	Status signal "Gap made". The signal is output when a gap has been made in the material.
bFail_b	BOOL	Status signal "Error". The signal is output when an error has been detected during the synchronous process.
nSpeedOut_a	INT	Actual speed for the actual phase integrator ➤ This input is to be connected to the function block <i>MotionControl</i>.
dnPosDiffOut_p	DINT	Positional deviation direct for the position controller ➤ This output is to be connected to the function block <i>MotionControl</i>
dnTir	DINT	Actual flying saw acceleration time. The signal is needed for the online calculation of the offset value and must be connected to the function block <i>OffsetCalculation</i> .

10.8 Function block *VersionHandling*

The function block *VersionHandling* provides three versions for indication in the related codes:

- Version of the Prepared Solution
- Version of the application library (depending on the Prepared Solution)
- Version of the basic library (is used in all Prepared Solutions)

The function block calculates the three elements for the internal data array from the major and subversions using the following formula:

$adwVersion[1] := ((dwProRelease * 100 + dwProLevel) * 100) + dwProServicePack;$

$adwVersion[2] := ((dwLib1Release * 100 + dwLib1Level) * 100) + dwLib1ServicePack;$

$adwVersion[3] := ((dwLib2Release * 100 + dwLib2Level) * 100) + dwLib2ServicePack;$

The field variable *adwVersion* is intended for indication in display codes with two decimal positions. In this way the user can straightforwardly determine the version states for the project and libraries.

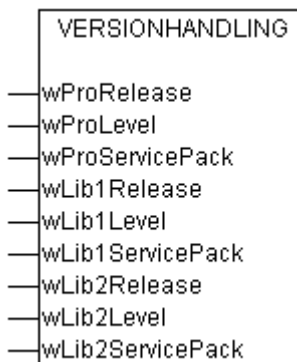
On the display of the code values the versions are indicated so that they are easy for the user to read (e.g. using GDC):

The screenshot shows the 'Global Drive Control - [Flying Saw V010001*]' window. On the left is a 'Parameter menu' with a tree view containing 'FCODE', 'Multitasking', 'Identify', 'Drive', 'LECOM', and 'Software' (highlighted). On the right is a table with columns: Code, Text, Value, and Unit. The table contains three rows of version data.

Code	Text	Value	Unit
C3999 001	version Prepared Solution	10001	
C3999 002	version library flying saw	10001	
C3999 003	version library multiplexer	10001	

At the bottom right of the window, status indicators show 'RSP IMP On line 001'.

Interface



FlyingSaw

Description of the function blocks

Task information

Can be called in:	☒ Cyclic task	☒ Time-controlled task (INTERVAL)	☒ Event-controlled task (EVENT)	☒ Interrupt task
-------------------	---	---	---	--

Variable names

Inputs (Variable type: VAR_INPUT)		
Identifier	Data type	Meaning
wProRelease	WORD	main version for the project:: the value entered must be in the range between 0 and 99.
wProLevel	WORD	subversion for the project:: the value entered must be in the range between 0 and 99.
wProServicePack	WORD	version of service-pack: the value entered must be in the range between 0 and 99.
wLib1Release	WORD	main version for library 1 (e.g. application library): the value entered must be in the range between 0 and 99.
wLib1Level	WORD	subversion for library 1 (e.g. application library): the value entered must be in the range between 0 and 99.
wLib1ServicePack	WORD	version of service-pack: the value entered must be in the range between 0 and 99.
wLib2Release	WORD	main version for library 2 (e.g. basic library): the value entered must be in the range between 0 and 99.
wLib2Level	WORD	subversion for library 2 (e.g. basic library): the value entered must be in the range between 0 and 99.
wLib2ServicePack	WORD	version of service-pack: the value entered must be in the range between 0 and 99.

Internal (Variable type: VAR)		
Identifier	Data type	Meaning
awVersion	ARRAY [1..3] OF DWORD	Data field with indication of the three versions: the first and the second number is the main versio, the third and the fourth is the subversion and the fith and the sixth is the number of the service-pack. Example: main version 1.0, subversion 0.0, service-pack 1.0, display: 10001 The values are calculated using the following formulas: $\text{adwVersion}[1] := ((\text{dwProRelease} * 100 + \text{dwProLevel}) * 100) + \text{dwProServicePack};$ $\text{adwVersion}[2] := ((\text{dwLib1Release} * 100 + \text{dwLib1Level}) * 100) + \text{dwLib1ServicePack};$ $\text{adwVersion}[3] := ((\text{dwLib2Release} * 100 + \text{dwLib2Level}) * 100) + \text{dwLib2ServicePack};$

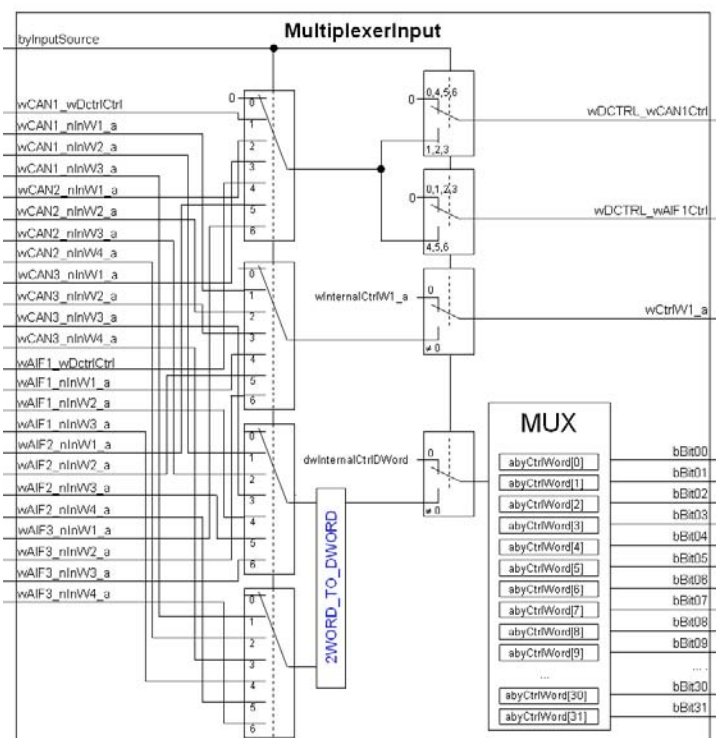
10.9 Function block *MultiplexerInput*

The function block *MultilexerInput* makes it possible to select a data interface for controlling the output bits. The following data sources can be selected using the input byte *byInputSource*:

- Internal data source in the form of control words (3 items of process data: word 1 ... 3)
- CAN1 (4 items of process data: word 0 ... 3)
- CAN2 (4 items of process data: word 0 ... 3)
- CAN3 (4 items of process data: word 0 ... 3)
- AIF1 (4 items of process data: word 0 ... 3)
- AIF2 (4 items of process data: word 0 ... 3)
- AIF3 (4 items of process data: word 0 ... 3)

The data words 2 and 3 are interpreted in bits and can be mapped to the output signals (*bBit00 ... bBit31*) as required using a multiplexer also included. For this purpose there is an element in the internal index array *abyCtrlWord[0..31]* for each of these output signals. Using this array it is defined for each output bit *bBit00 ... bBit31* from which source in words 2 and 3 the information for this output signal is drawn (multiplexer).

Interface



Task information

Can be called in:	☒ Cyclic task	☒ Time-controlled task (INTERVAL)	☒ Event-controlled task (EVENT)	☒ Interrupt task
-------------------	---	---	---	--

Variable names

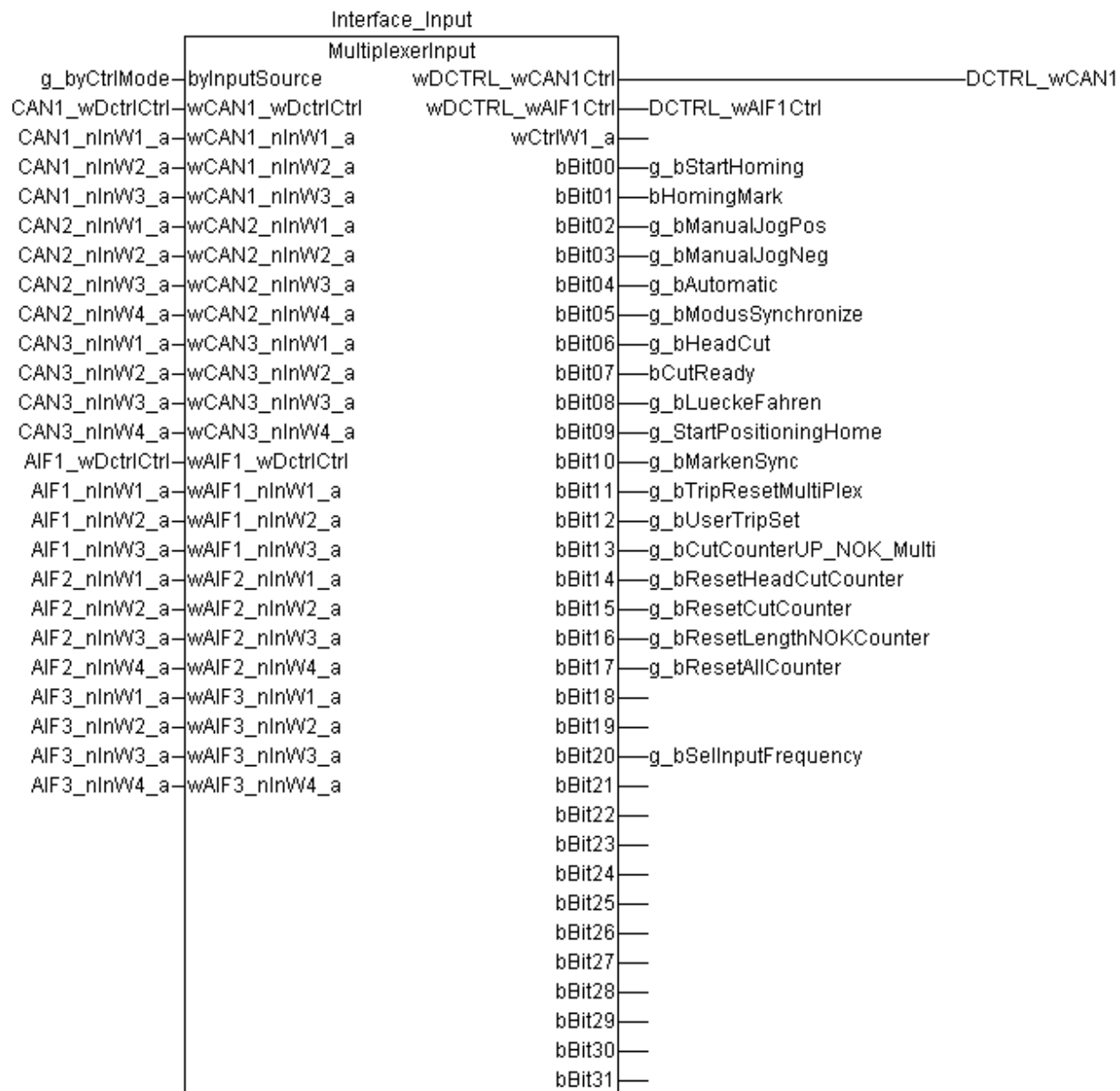
Inputs (Variable type: VAR_INPUT)		
Identifier	Data type	Meaning
byInputSource	BYTE	Selection of the control source for the application:
		0 Control code
		1 CAN1
		2 CAN2
		3 CAN3
		4 AIF1
		5 AIF2
		6 AIF3
wCAN1_wDctrlCtrl	WORD	Input for the system variable <i>CAN1_wDctrlCtrl</i>
wCAN1_nInW1_a	WORD	Input for the system variable <i>CAN1_nInW1_a</i>
wCAN1_nInW2_a	WORD	Input for the system variable <i>CAN1_nInW2_a</i>
wCAN1_nInW3_a	WORD	Input for the system variable <i>CAN1_nInW3_a</i>
wCAN2_nInW1_a	WORD	Input for the system variable <i>CAN2_nInW1_a</i>
wCAN2_nInW2_a	WORD	Input for the system variable <i>CAN2_nInW2_a</i>
wCAN2_nInW3_a	WORD	Input for the system variable <i>CAN2_nInW3_a</i>
wCAN2_nInW4_a	WORD	Input for the system variable <i>CAN2_nInW4_a</i>
wCAN3_nInW1_a	WORD	Input for the system variable <i>CAN3_nInW1_a</i>
wCAN3_nInW2_a	WORD	Input for the system variable <i>CAN3_nInW2_a</i>
wCAN3_nInW3_a	WORD	Input for the system variable <i>CAN3_nInW3_a</i>
wCAN3_nInW4_a	WORD	Input for the system variable <i>CAN3_nInW4_a</i>
wAIF1_wDctrlCtrl	WORD	Input for the system variable <i>AIF1_wDctrlCtrl</i>
wAIF1_nInW1_a	WORD	Input for the system variable <i>AIF1_nInW1_a</i>
wAIF1_nInW2_a	WORD	Input for the system variable <i>AIF2_nInW2_a</i>
wAIF1_nInW3_a	WORD	Input for the system variable <i>AIF1_nInW3_a</i>
wAIF2_nInW1_a	WORD	Input for the system variable <i>AIF3_nInW1_a</i>
wAIF2_nInW2_a	WORD	Input for the system variable <i>AIF2_nInW2_a</i>
wAIF2_nInW3_a	WORD	Input for the system variable <i>AIF2_nInW3_a</i>
wAIF2_nInW4_a	WORD	Input for the system variable <i>AIF2_nInW4_a</i>
wAIF3_nInW1_a	WORD	Input for the system variable <i>AIF3_nInW1_a</i>
wAIF3_nInW2_a	WORD	Input for the system variable <i>AIF3_nInW2_a</i>
wAIF3_nInW3_a	WORD	Input for the system variable <i>AIF3_nInW3_a</i>
wAIF3_nInW4_a	WORD	Input for the system variable <i>AIF3_nInW4_a</i>

FlyingSaw

Description of the function blocks

Outputs (Variable type: VAR_OUTPUT)		
Identifier	Data type	Meaning
wDCTRL_wCAN1Ctrl	WORD	Output of the control word for the system variable <i>DCTRL_wCAN1Ctrl</i>
wDCTRL_wAIF1Ctrl	WORD	Output of the control word for the system variable <i>DCTRL_wAIF1Ctrl</i>
wCtrlW1_a	INT	Output of the data word (depending on the value on <i>byInputSource</i> derived from the variables <i>wInternalCtrlW1_a</i> , <i>wCAN1_nInW1_a</i> , <i>wCAN2_nInW2_a</i> , <i>wCAN3_nInW2_a</i> , <i>wAIF1_nInW1_a</i> , <i>wAIF2_nInW2_a</i> or <i>wAIF3_nInW2_a</i>)
bBit00 bBit01 bBit02 bBit03 ... bBit30 bBit31	BOOL	Control bits 0 to 31: The individual bits are formed from the internal double control word <i>dwCtrlWord</i> . Here any output bit <i>bBit00</i> , <i>bBit01</i> ... <i>bBit31</i> can be assigned to any bit in the double control word (<i>dwCtrlWord.0</i> ... <i>dwCtrlWord.31</i>). The internal index variable <i>abyCtrlWord[i]</i> (<i>i</i> = 0 ... 31) is used for the assignment; the index <i>i</i> addresses the required output bit <i>bBit00</i> , <i>bBit01</i> ... <i>bBit31</i> and the value of <i>abyCtrlWord[i]</i> the required source bit in the double control word (<i>dwCtrlWord.0</i> ... <i>dwCtrlWord.31</i>). The value range allowed for <i>abyCtrlWord[i]</i> is 0 ... 31.

Example wiring for Prepared Solution:



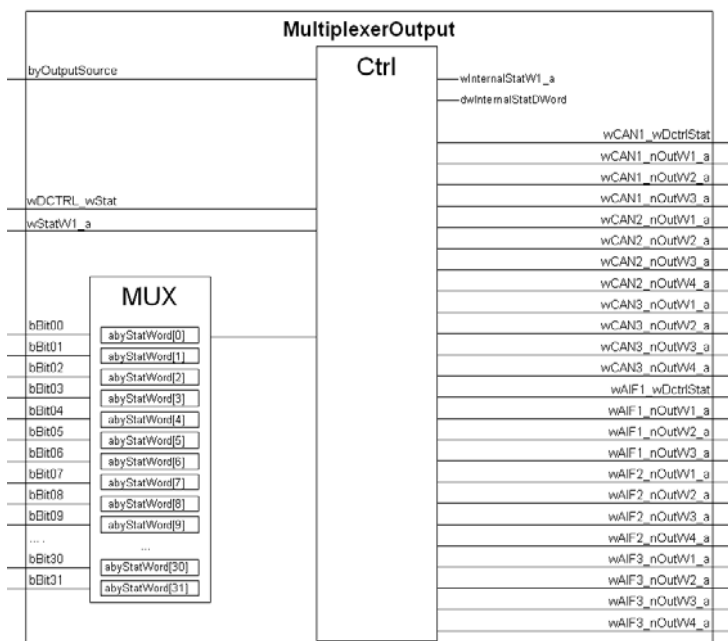
10.10 Function block *MultiplexerOutput*

The *MultiplexerOutput* function block makes it possible to select a data interface for the output of status information. The following data interfaces can be selected using the input byte *byOutputSource*:

- Internal data source in the form of status words (3 items of process data: word 1 ... 3)
- CAN1 (4 items of process data: word 0 ... 3)
- CAN2 (4 items of process data: word 0 ... 3)
- CAN3 (4 items of process data: word 0 ... 3)
- AIF1 (4 items of process data: word 0 ... 3)
- AIF2 (4 items of process data: word 0 ... 3)
- AIF3 (4 items of process data: word 0 ... 3)

The data words 2 and 3 are interpreted in bits and can be mapped from any of the input signals (*bBit00 ... bBit31*) as required using a multiplexer also included. For this purpose there is an element in the internal index array *abyStatWord[0..31]* for each of these output signals. Using this array it is defined for each bit of this double status word from which source *bBit00 ... bBit31* the information for this double status word is drawn (multiplexer).

Interface



Task information

Can be called in:	☒ Cyclic task	☒ Time-controlled task (INTERVAL)	☒ Event-controlled task (EVENT)	☒ Interrupt task
-------------------	---	---	---	--

FlyingSaw

Description of the function blocks

Variable names

Inputs (Variable type: VAR_INPUT)		
Identifier	Data type	Meaning
byOutputSource	BYTE	Selection of the target interface for the status information for the application:
		0 Status code
		1 CAN1
		2 CAN2
		3 CAN3
		4 AIF1
		5 AIF2
		6 AIF3
wDCTRL_wStat	WORD	Input for the system status word (system variable <i>DCTRL_wStat</i>)
wStatW1_a	INT	Definition of the first data word as required (depending on the value on <i>byOutputSource</i> this value is output to the variable <i>wInternalStatW1_a</i> , <i>wCAN1_nOutW1_a</i> , <i>wCAN2_nOutW2_a</i> , <i>wCAN3_nOutW2_a</i> , <i>wAIF1_nOutW1_a</i> , <i>wAIF2_nOutW2_a</i> or <i>wAIF3_nOutW2_a</i>)
bBit00 bBit01 bBit02 bBit03 ... bBit30 bBit31	BOOL	Definition by bit of the status information (bits 0 to 31): The individual bits are formed from the internal double status word <i>dwStatWord</i> . Here any input bit <i>bBit00</i> , <i>bBit01</i> ... <i>bBit31</i> can be assigned to any bit in the double status word (<i>dwStatWord.0</i> ... <i>dwStatWord.31</i>). The internal index variable <i>abyStatWord[i]</i> (<i>i</i> = 0 ... 31) is used for the assignment; the index <i>i</i> defines the required bit in the double status word (<i>dwStatWord.0</i> ... <i>dwStatWord.31</i>) and the value of <i>abyStatWord[i]</i> the required input bit <i>bBit00</i> , <i>bBit01</i> ... <i>bBit31</i> for this output bit. The value range allowed for <i>abyStatWord[i]</i> is 0 ... 31.

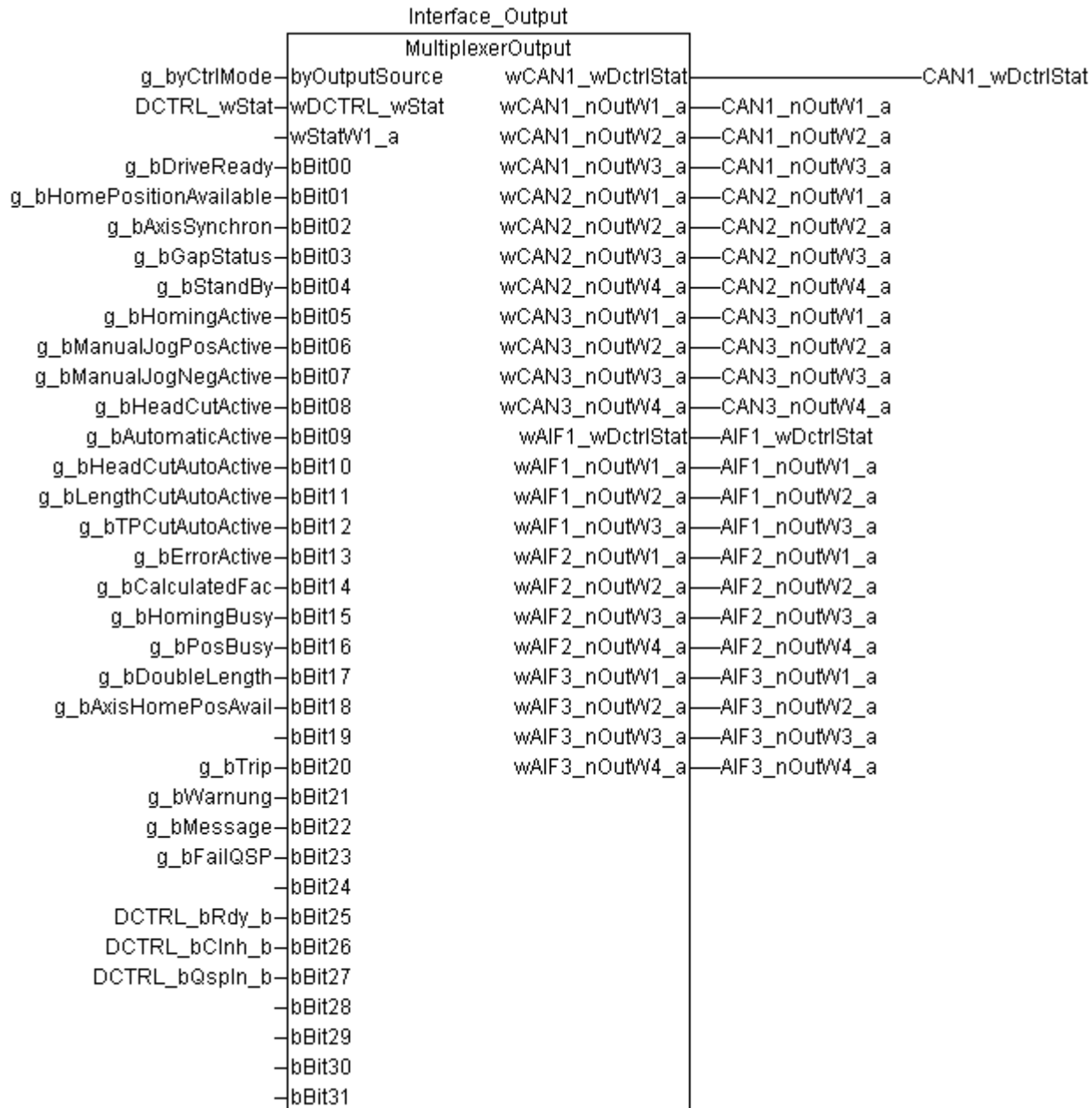
FlyingSaw

Description of the function blocks

Outputs (Variable type: VAR_OUTPUT)		
Identifier	Data type	Meaning
wCAN1_wDctrlStat	WORD	Output for the system variable CAN1_wDctrlStat
wCAN1_nOutW1_a	WORD	Output for the system variable CAN1_nOutW1_a
wCAN1_nOutW2_a	WORD	Output for the system variable CAN1_nOutW2_a
wCAN1_nOutW3_a	WORD	Output for the system variable CAN1_nOutW3_a
wCAN2_nOutW1_a	WORD	Output for the system variable CAN2_nOutW1_a
wCAN2_nOutW2_a	WORD	Output for the system variable CAN2_nOutW1_a
wCAN2_nOutW3_a	WORD	Output for the system variable CAN2_nOutW3_a
wCAN2_nOutW4_a	WORD	Output for the system variable CAN2_nOutW4_a
wCAN3_nOutW1_a	WORD	Output for the system variable CAN3_nOutW1_a
wCAN3_nOutW2_a	WORD	Output for the system variable CAN3_nOutW2_a
wCAN3_nOutW3_a	WORD	Output for the system variable CAN3_nOutW3_a
wCAN3_nOutW4_a	WORD	Output for the system variable CAN3_nOutW4_a
wAIF1_wDctrlStat	WORD	Output for the system variable AIF1_wDctrlStat
wAIF1_nOutW1_a	WORD	Output for the system variable AIF1_nOutW1_a
wAIF1_nOutW2_a	WORD	Output for the system variable AIF1_nOutW2_a
wAIF1_nOutW3_a	WORD	Output for the system variable AIF1_nOutW3_a
wAIF2_nOutW1_a	WORD	Output for the system variable AIF2_nOutW1_a
wAIF2_nOutW2_a	WORD	Output for the system variable AIF2_nOutW2_a
wAIF2_nOutW3_a	WORD	Output for the system variable AIF2_nOutW3_a
wAIF2_nOutW4_a	WORD	Output for the system variable AIF2_nOutW4_a
wAIF3_nOutW1_a	WORD	Output for the system variable AIF3_nOutW1_a
wAIF3_nOutW2_a	WORD	Output for the system variable AIF3_nOutW2_a
wAIF3_nOutW3_a	WORD	Output for the system variable AIF3_nOutW3_a
wAIF3_nOutW4_a	WORD	Output for the system variable AIF3_nOutW4_a

Internal (Variable type: VAR)		
Identifier	Data type	Meaning
wInternalStatW1_a	INT	Internal status word for the indication of the actual data word (input signal wCtrlW1_a): The internal status word wInternalStatW1_a is assigned with byOutputSource = 0 the value on input wCtrlW1_a.
dwInternalStatDWord	DWORD	Internal variable for the indication of the double status word: the double status word is formed from the input signals bBit00, bBit01 ... bBit31 using the multiplexer, if byOutputSource = 0 is pre-selected.

Example wiring for Prepared Solution:



11 Appendix

11.1 Possible error sources

11.1.1 Slip at the measuring wheel or at the material infeed

A very probable cause is that the material infeed and the measuring wheel are not correctly positioned in relation to each other and "slip" occurs at this point resulting in the incorrect indication of the material speed to the controller. Slip can be recognised by the set lengths not passing through correctly and the saw carriage not moving in "synchronism" with the material.

11.1.2 Interference on the master encoder signal

Electromagnetic interference coupled onto the encoder cable can also affect the cutting accuracy.

11.1.3 Incorrectly set synchronisation ratio / normalisation factor

An incorrectly set synchronisation ratio (normalisation factor) can cause significant cutting tolerances that will become particularly large when the line speed is changed. To rectify the problem the normalisation factor should be checked again.



TIP!

During use measuring wheels are subject to wear resulting in a reduction in their circumference. You can correct for this wear using the code C3005/000

11.2 Global variables

11.2.1 Global

Global variables (Variable type: VAR_GLOBAL)		
Identifier	Data type	Value/meaning
g_dnScaleWheel	DINT	Circumference of the measuring wheel
g_dnPulseSensor	DINT	Encoder pulses from the measuring wheel
g_nFrequencyDFIN	INT	Input frequency of the measuring wheel
g_nFrequencyDFINNorm	INT	Standardised input frequency of the measuring wheel
g_bStartlengthCalculator	BOOL	Start the length calculator
g_dnCutLengthInput	DINT	Set length
g_dnGapLengthInput	DINT	Length of the gap
g_dnGapLength_p	DINT	Length of the gap in increments
g_bMarkenSyncAxtive	BOOL	Activate mark synchronisation
g_bOffsetUsed	BOOL	Offset taken into account during a length cut
g_bCalculateNewOffset	BOOL	Calculate new offset
g_dnTir	DINT	Flying saw acceleration time
g_bOffsetConsider	BOOL	Offset value has been taken into consideration
g_dnOffsetLengthCalculator	DINT	Offset for the length calculator
g_dnOffsetSynchronize	DINT	Offset for the synchronisation
g_bStartDecelerationSaw	BOOL	Start of the deceleration ramp
g_nCompensationTrimming	INT	Increments that are compensated during the synchronisation
g_nCompensationTrimmingUsed	INT	Compensation completed
g_dnCutterKompensation	DINT	Width of cut compensation
DelaySynchronSignal	INT	Delay on the synchronised signal
g_bReset	BOOL	Reset
g_bResetSync	BOOL	Reset the synchronisation
g_bExternSetValuesEnable	BOOL	Switch to external setpoints
g_dnAbortCalcOffset_units	DINT	Calculation of the offset is interrupted just before synchronising
g_dnAbortCalcOffset_Inc	DINT	Calculation of the offset is interrupted just before synchronising
g_dnActualValueLengthCalculator	DINT	Actual length calculator value
g_nSpeedOutSaw	INT	Flying saw set speed
g_dnPosDiffOutSaw	DINT	Setpoint for the phase controller
g_bAutomaticFlag	BOOL	Automatic mode is active
g_bWatchLengthTP	BOOL	Mark monitoring active
g_ManualProfile	Profile	Profile for jogging
g_SecondProfile	Profile	Profile for the movement to the initial position
g_BackProfile	Profile	Profile for return positioning after a synchronous movement
g_FreeDriveLimitSwitch	Profile	Move clear from limit switch
g_Profile	Profile	Profile data
g_Axis	Axis	Axis data

Global variables (Variable type: VAR_GLOBAL)		
g_bAbort	BOOL	Cancel the positioning
g_bReference	BOOL	Start homing
g_bAxisHomePosAvail	BOOL	Initial position reached
g_byMotionControlMode	BYTE	Positioning mode
g_bProfileStart	BOOL	Start a positioning
g_bManual	BOOL	Start manual operation
g_bMotionDone	BOOL	Positioning complete
g_bMotorInvert	BOOL	Motor mounting position
g_dnMaxFollowError	DINT	Maximum following error
g_bFollowError	BOOL	Following error resolution
g_bHeadCutInProzess	BOOL	Top cut in length operation

11.2.2 VarCounter_FS

Global variables (Variable type: VAR_GLOBAL)		
Identifier	Data type	Value/meaning
g_bCountUpHeadCut	BOOL	Increment top cut counter
g_bCountDownHeadCut	BOOL	Decrement top cut counter
g_bResetHeadCutCounter	BOOL	Reset top cut counter
g_bCutCounterUP	BOOL	Increment cut counter
g_bCutCounterDOWN	BOOL	Decrementing cut counter
g_bResetCutCounter	BOOL	Reset cut counter
g_bCutCounterUP_NOK	BOOL	Increment scrap counter
g_bResetLengthNOKCounter	BOOL	Decrement scrap counter
g_bResetAllCounter	BOOL	Reset all counters
g_nCounterHeadCut	INT	Number of top cuts
g_nCutCounterOK	INT	Number of pieces cut
g_nCutCounterNOK	INT	Number of scraps pieces that have been separated out using immediate cuts

11.2.3 VarErrorHandling

Global variables (Variable type: VAR_GLOBAL)		
Identifier	Data type	Value/meaning
g_bTrip	BOOL	Trip active
g_bWarnung	BOOL	Warning active
g_bMessage	BOOL	Message active
g_bFailQSP	BOOL	Axis is in QSP

11.2.4 VarInterfaceFlyingSaw

Global variables (Variable type: VAR_GLOBAL)		
Identifier	Data type	Value/meaning
g_bDriveReady	BOOL	Drive ready signal
g_bStartHoming	BOOL	Start homing
g_bHomingMark	BOOL	Homing cam
g_bManualJogPos	BOOL	Inching in positive direction
g_bManualJogNeg	BOOL	Inching in negative direction
g_bStartPositioningHome	BOOL	Move to initial position
g_bAutomatic	BOOL	Start automatic operation
g_bCutReady	BOOL	Cut done signal
g_bHeadCut	BOOL	Trigger top cut
g_bStartGap	BOOL	Make gap in the material
g_bTPUnrealised	BOOL	Mark not detected during mark control
g_bMarkenSync	BOOL	Activate mark synchronisation
g_bModusSynchronize	BOOL	Synchronisation mode
g_bTripReset	BOOL	Reset error
g_bTripresetMultiPlex	BOOL	Reset error using multiplexer
g_bUserTripSet	BOOL	Set user error
g_bImperialUnits	BOOL	Switch between imperial and metric measuring system
g_nSimulationFrequency	INT	Simulation speed
g_bSelInputFrequency	BOOL	Switch between DFIN and simulation
g_dnFollowErrorLimt	DINT	Following error limit
g_bHomePositionAvailable	BOOL	Home position known
g_bGapStatus	BOOL	Gap made in the material
g_bAxisSynchron	BOOL	Axis synchronised signal
g_wActualFaultNumber	WORD	Actual error number
g_bHomingBusy	BOOL	Homing is active
g_bPosBusy	BOOL	Positioning is active
g_bDoubleLength	BOOL	Double length detected
g_dnActPosFlyingSaw	DINT	Actual position of the axis
g_byCtrlMode	BYTE	Multiplexer mode
g_nFiltering	INT	Master speed filtering
g_dwTiFlyingSaw	DWORD	Flying saw acceleration time
g_dwTifFlyingSaw	DWORD	Flying saw deceleration time
g_dnOffsetTP	DINT	Distance between mark detection and the initial position
g_bSoftwareLimitsActive	BOOL	Software limit switches active
g_bCutCounterUP_NOK_Multi	BOOL	Operate scrap counter via the multiplexer

11.2.5 VarLimitsSwitches

Global variables (Variable type: VAR_GLOBAL)		
Identifier	Data type	Value/meaning
g_bLimitSwitchPos	BOOL	Positive hardware limit switch
g_bLimitSwitchNeg	BOOL	Negative hardware limit switch
g_bPosLimitSwitch	BOOL	Positive software limit switch
g_bNegLimitSwitch	BOOL	Negative software limit switch
g_dnPosLimitSwitch	DINT	Position of the positive software limit switch
g_dnNegLimitSwitch	DINT	Position of the negative software limit switch

11.2.6 VarNormFactor

Global variables (Variable type: VAR_GLOBAL)		
Identifier	Data type	Value/meaning
g_bCalculatedFac	BOOL	Normalisation factor calculated
g_dnNormalizationNumerator	DINT	Normalisation factor numerator
g_dnNormalizationDenominator	DINT	Normalisation factor denominator

11.2.7 VarOperationVisu

Global variables (Variable type: VAR_GLOBAL)		
Identifier	Data type	Value/meaning
g_bStartHomingVisu	BOOL	Start of homing via the visualisation
g_bManualVisu	BOOL	Start of manual operation via the visualisation
g_bManualJogPosVisu	BOOL	Positive inching of the axis via the visualisation
g_bManualJogNegVisu	BOOL	Negative inching of the axis via the visualisation
g_bAutomaticVisu	BOOL	Start automatic via the visualisation
g_bHeadCutVisu	BOOL	Trigger top cut via the visualisation
g_bCutReadyVisu	BOOL	Trigger cut done signal via the visualisation

11.2.8 VarStatusMachine

Global variables (Variable type: VAR_GLOBAL)		
Identifier	Data type	Value/meaning
g_bStandBy	BOOL	"Standby" mode
g_bHomingActive	BOOL	"Homing active" mode
g_bManualJogPosActive	BOOL	"Inching positive" mode
g_bManualJogNegActive	BOOL	"Inching negative" mode
g_PositioningActive	BOOL	"Positioning" mode
g_bHeadCutActive	BOOL	"Top cut" mode
g_bAutomaticActive	BOOL	"Automatic" mode
g_bHeadCutAutoActive	BOOL	"Top cut in automatic operation" mode
g_bLengthCutAutoActive	BOOL	"Length cut" mode

Global variables (Variable type: VAR_GLOBAL)		
g_bTPCutAutoActive	BOOL	"Mark cut" mode
g_bErrorActive	BOOL	Error detected
g_wState	Program state	Status of the application

11.2.9 VarVersion

Global variables (Variable type: VAR_GLOBAL)		
Identifier	Data type	Value/meaning
C_wFlyingSawProjectVersionER	WORD	main version of the project
C_wFlyingSawProjectVersionEL	WORD	subversion of the project
C_wFlyingSawProjectVersionESP	WORD	service-pack-version of the project

11.3 Codes of the Prepared Solution

11.3.1 Table of application codes

Along with system codes (see documentation 9300ET ServoPLC), with the Prepared Solution important settings for the machine function can be made using application codes from the range C3000/000.

Code	Possible settings:		Comment	
	Default	Selection		
C3000	0	0: Not inverted 1: Inverted		Inversion of the motor mounting position
C3001	0	0: Not inverted 1: Inverted		Inversion of the master frequency (DFIN). Necessary if the measuring wheel is mounted counter-rotating.
C3003	1	0... {1} ... 65536		Gearbox factor numerator of measuring wheel
C3004	1	0... {1} ... 65536		Gearbox factor denominator of measuring wheel
C3005	1	0... {1} ...		Circumference of the measuring wheel
C3006	1	0... {1} [units]} 2147483647		Number of increments from the encoder connected to the measuring wheel
C3007	7500000	0... {1} [inc/rev]} 2147483647		Entry of the set cut length. The entry is made in the format cut length in units * 10000. Example: 1234.56 mm set length, entry in code: 12345600 units.
C3009	100	0... {1[ms]} ... 32767		Delay on the signal "axis synchronised"
C3010	500	0... {1[inc/ms]} ... 32767		Simulation speed (simulation of the master frequency input)
C3012	1500	0... {1} [ms]} 4294967295		Flying saw acceleration time, during the synchronising process
C3013	2500	0 {1} [ms]} 4294967295		Flying saw deceleration time, after the synchronous movement
C3014	5.0000	0.0000 {1 [unit]} 214000.0000		Length of the gap that is to be made after processing the material
C3015	0.0000	0.0000 {1 [unit]} 214000.0000		Mark detection offset. Distance between mark detection and flying saw initial position.
C3016	0.0000	0.0000 {1 [unit]} 214000.0000		Width of cut compensation
C3017	0	0: Not activated 1: Activated		Monitoring of the mark detection. Length between the marks is monitored using the length calculator and a cut "forced" if a mark is not detected.
C3100	500.0000	-214000.0000... {0.0001} [units/s]} 214000.0000		Speed for manual movement of the axis (inching operation, valid for positive and negative inching)
C3101	750.0000	-214000.0000... {0.0001} [units/s^2]} 214000.0000		Acceleration for manual movement of the axis (inching operation, valid for positive and negative inching)
C3102	750.0000	-214000.0000... {0.0001} [units/s^2]} 214000.0000		Deceleration for manual movement of the axis (inching operation, valid for positive and negative inching)
C3200	0.0000	-214000.0000... {0.0001} [units]} 214000.0000		Position of the return movement profile. Movement to the initial position. The initial position should be home position!
C3201	1200.0000	-214000.0000... {0.0001} [units/s]} 214000.0000		Speed for the return movement profile for the initial position movement
C3202	750.0000	-214000.0000... {0.0001} [units/s^2]} 214000.0000		Acceleration for the return movement profile for the initial position movement
C3203	750.0000	-214000.0000... {0.0001} [units/s^2]} 214000.0000		Deceleration for the return movement profile for the initial position movement
C3213	0	0: >_Rn_MP/TP 1: <_Rn_MP/TP 8: >_MP/TP 9: <_MP/TP		Definition of the homing mode: <u>Symbology:</u> > Movement in positive direction < Movement in negative direction Lp Positive limit switch Ln Negative limit switch Rp Positive edge on the homing switch Rn Negative edge on the homing switch MP/TP Zero pulse from the motor feedback system or touch-probe edge on a digital input
C3218	10.0000	0.0001... {0.0001} [units]} ... 214748.0000		Entry of the following error shutdown limit
C3223	0.0000	0... {0.0001} [units]} ... 214748.0000		Limit position for the positive software limit <u>Comment:</u> software limits are only active if the home position for the axis is known and the limits are not set to zero.

	Possible settings:			
C3224	0.0000	-214748.0000... {0.0001 [units]}	0	Limit position for the negative software limit <u>Comment:</u> software limits are only active if the home position for the axis is known and the limits are not set to zero.
C3225	100.0000	-214748.0000... {0.0001 ... 214748.0000 [inc]}		Home offset
C3242	100	1 {1[rpm]}	... 16000	Speed for the homing
C3252	5.00	0.01 {0.01[s]}	... 650.00	Ramp times for the homing: These ramp times relate to C0011/000.
C3300	500.0000	-214000.0000... {0.0001 ... 214000.0000 [units/s]}		Speed for the return movement profile for the axis positioning at the initial position.
C3301	7500.0000	-214000.0000... {0.0001 ... 214000.0000 [units/s^2]}		Acceleration for the return movement profile for the axis positioning at the initial position.
C3302	7500.0000	-214000.0000... {0.0001 ... 214000.0000 [units/s^2]}		Deceleration for the return movement profile for the axis positioning at the initial position.
C3501	0	0: metric (1[s_unit] = 1[mm]) 1: imperial (1[s_unit] = 1[inch])		Selection of the measuring system: This code has no effect on the functionality, but affects only the visualisations in the programmable variant of the Prepared Solution.
C3990	-	1: Error state 10: Standby state Manual operation 20: Positive manual control 21: Negative manual control 22: Homing active 30: Automatic "flying saw" 40: Top cut is being performed 41: Length cut active 42: Mark cut active 43: Move to initial position 50: Initialisation 100:		Display code: state of the "flying saw" Prepared Solution
C3998	-	0... {1} ... 65535		Display code: actual error message, this contains system and application error messages.
C3999	-	0 ... {0.01} ... 99.99		Display code: version of the Prepared Solution. The first and the second number is the main version, the third and the fourth is the subversion and the fifth and the sixth is the number of the service-pack.
C4000	0	0: Internal control words 1: CAN1 2: CAN2 3: CAN3 4: AIF1 5: AIF2 6: AIF3		Preselection of the control interface: All three axes in the system can be controlled using the control interface.

Possible settings:			
C4010			Configuration codes for the control interface: here each sub-code defines where the control information is to be drawn for specific control functionality: For information on the control functionalities for the default settings, please see chapter 3.8.1
1	0	0:	Control bit 0
2	1	1:	Control bit 1
3	2	2:	Control bit 2
4	3	3:	Control bit 3
5	4	4:	Control bit 4
6	5	5:	Control bit 5
7	6	6:	Control bit 6
8	7	7:	Control bit 7
9	8	8:	Control bit 8
10	9	9:	Control bit 9
11	10	10:	Control bit 10
12	11	11:	Control bit 11
13	12	12:	Control bit 12
14	13	13:	Control bit 13
15	14	14:	Control bit 14
16	15	15:	Control bit 15
17	16	16:	Control bit 16
18	17	17:	Control bit 17
19	18	18:	Control bit 18
20	19	19:	Control bit 19
21	20	20:	Control bit 20
22	21	21:	Control bit 21
23	22	22:	Control bit 22
24	23	23:	Control bit 23
25	24	24:	Control bit 24
26	25	25:	Control bit 25
27	26	26:	Control bit 26
28	27	27:	Control bit 27
29	28	28:	Control bit 28
30	29	29:	Control bit 29
31	30	30:	Control bit 30
32	31	31:	Control bit 31
C4012		For information on the status information for the default settings, please see chapter 3.8.1	Configuration codes for the status interface: here each sub-code defines where the status information is to be drawn for a specific status bit:
1	0	0:	1: Status bit 0
2	1	1:	2: Status bit 1
3	2	2:	3: Status bit 2
4	3	3:	4: Status bit 3
5	4	4:	5: Status bit 4
6	5	5:	6: Status bit 5
7	6	6:	7: Status bit 6
8	7	7:	8: Status bit 7
9	8	8:	9: Status bit 8
10	9	9:	10: Status bit 9
11	10	10:	11: Status bit 10
12	11	11:	12: Status bit 11
13	12	12:	13: Status bit 12
14	13	13:	14: Status bit 13
15	14	14:	15: Status bit 14
16	15	15:	16: Status bit 15
17	16	16:	17: Status bit 16
18	17	17:	18: Status bit 17
19	18	18:	19: Status bit 18
20	19	19:	20: Status bit 19
21	20	20:	21: Status bit 20
22	21	21:	22: Status bit 21
23	22	22:	23: Status bit 22
24	23	23:	24: Status bit 23
25	24	24:	25: Status bit 24
26	25	25:	26: Status bit 25
27	26	26:	27: Status bit 26
28	27	27:	28: Status bit 27
29	28	28:	29: Status bit 28
30	29	29:	30: Status bit 29
31	30	30:	31: Status bit 30
32	31	31:	32: Status bit 31

	Possible settings:		
C4135	0	0 4294967295 {1}	Control code: Using the control code the flying saw can be controlled (with C4000/000 = 0). The individual bits of the control code can be defined as required using the configuration codes C4010/001 ... 0032. The factory-set bit assignment is given in code C4010/xxx.
C4136	-	0 4294967295 {1}	Indication of the actual application control word: Dependent on C4000/000 the actual application control word is displayed here. The meaning of the individual bits of the application control word can be defined as required using the configuration codes C4010/001 ... 0032. The factory-set bit assignment is given in code C4010/xxx.
C4150	0	0 4294967295 {1}	Status code (display code): Using the status code the complete application can be monitored (with C4000/000 = 0). The individual bits of the status code can be defined as required using the configuration codes C4012/001 ... 0032. The factory-set bit assignment is given in code C4012/xxx.
C4500	-	0... 32767 {1}	Cut counter indication: All pieces completely processed are counted here. The cut counter is decremented when the scrap counter has been triggered
C4501	-	0... 32767 {1}	Scrap counter indication: All pieces cut out of the material with an immediate cut are counted here.
C4502	-	0... 32767 {1}	Top cut counter: The number of top cuts is counted here.
C5000	-	-2147480000... 2147480000 {0.0001[units]}	Indication of the actual position of the axis

11.3.2 Code initialisation values

Unlike the initialisation values for the default setting, the following codes are initialised via the project

Code	Default setting	Comment
C0425/000	6	DFIN number of increments 16384 inc/rev.
C0428/000	1	Touch-probe via digital input X5/E5 (DFIN)
C0911/000	0	Touch-probe via zero pulse (MCTRL)
C2104/000	1	Auto-start after mains power-up
C4010/002	1,0000	Control bit 1 is read from input bit 1
C4010/003	2,0000	Control bit 2 is read from input bit 2
C4010/004	3,0000	Control bit 3 is read from input bit 3
C4010/005	4,0000	Control bit 4 is read from input bit 4
C4010/006	5,0000	Control bit 5 is read from input bit 5
C4010/007	6,0000	Control bit 6 is read from input bit 6
C4010/008	7,0000	Control bit 7 is read from input bit 7
C4010/009	8,0000	Control bit 8 is read from input bit 8
C4010/010	9,0000	Control bit 9 is read from input bit 9
C4010/011	10,0000	Control bit 10 is read from input bit 10
C4010/012	11,0000	Control bit 11 is read from input bit 11
C4010/013	12,0000	Control bit 12 is read from input bit 12
C4010/014	13,0000	Control bit 13 is read from input bit 13
C4010/015	14,0000	Control bit 14 is read from input bit 14
C4010/016	15,0000	Control bit 15 is read from input bit 15
C4010/017	16,0000	Control bit 16 is read from input bit 16
C4010/018	17,0000	Control bit 17 is read from input bit 17
C4010/019	18,0000	Control bit 18 is read from input bit 18
C4010/020	19,0000	Control bit 19 is read from input bit 19
C4010/021	20,0000	Control bit 20 is read from input bit 20
C4010/022	21,0000	Control bit 21 is read from input bit 21
C4010/023	22,0000	Control bit 22 is read from input bit 22
C4010/024	23,0000	Control bit 23 is read from input bit 23
C4010/025	24,0000	Control bit 24 is read from input bit 24
C4010/026	25,0000	Control bit 25 is read from input bit 25
C4010/027	26,0000	Control bit 26 is read from input bit 26
C4010/028	27,0000	Control bit 27 is read from input bit 27
C4010/029	28,0000	Control bit 28 is read from input bit 28
C4010/030	29,0000	Control bit 29 is read from input bit 29
C4010/031	30,0000	Control bit 30 is read from input bit 30
C4010/032	31,0000	Control bit 31 is read from input bit 31

11.4 Error messages

For the error messages the Prepared Solutions differentiate between the so-called system error messages (see chapter 11.4.1) and process-based error messages (expanded error messages, see chapter 11.4.2).

11.4.1 System error messages

The system error messages are generated in the operating system in the target system and cover all impermissible, drive-based states (e.g. overcurrent, undervoltage, error in the feedback system, ...).

The following system error messages can occur in the ServoPLC:

Error number	Error code	Cause	Remedy
11	OC1	Real short circuit between motor phases	Disconnect motor and measure between winding connections for short circuits Disconnect supply cable and check between phases for short circuit. Operate inverter without load: the error should now reset itself. If this is not the case, there is an internal short circuit => replace controller or send for repair
		High, capacitive charging current on the motor cable	Use shorter/lower capacitance motor cable Fit choke to the output phases on the controller.
12	OC2	Real short to earth on one of the motor phases	Disconnect motor and check winding connections for short to earth. Disconnect supply cable and check between phases and earth for short circuit. Operate inverter without load: the error should now reset itself. If this is not the case, there is an internal short to earth => replace controller or send for repair
		High, capacitive charging current on the motor cable	Use shorter/lower capacitance motor cable Fit choke to the output phases on the controller.
15	OC5	Frequent/excessively long acceleration processes with overcurrent ($I_{Motor} > I_{Nom, inverter}$)	Check drive dimensioning; if necessary increase drive power.
		Continuous overload with $I_{Motor} > 1.05 \times I_{Nom, inverter}$	
16	OC6	$I^2 \times t$ overload (C0120)	Check drive dimensioning; if necessary increase drive power.
17	OC7	$I^2 \times t$ overload (C0130)	Check drive dimensioning; if necessary increase drive power.
18	OC8	$I^2 \times t$ overload (C0127)	Check drive dimensioning; if necessary increase drive power.
20	OU	Mains voltage too high	Check mains voltage. Check controller for correct adjustment of the mains voltage (code C0173/000) Check drive dimensioning.
		Energy fed back during braking excessive (regenerative operation)	On individual operation of the drive from supply: - Extend deceleration ramps (codes C0013/000, C0105/000) - If possible: set up DC-bus connection to other controllers - Use braking unit or supply unit and brake module or regenerative module - Check controller for correct adjustment of the mains voltage (code C0173/000) - Check drive dimensioning In case of drives with DC bus connection: - Check fuses on all controllers on the DC bus as well as fuses in the brake modules/power supply modules/regenerative modules - Check controller for correct adjustment of the mains voltage (code C0173/000) - Check drive dimensioning
		Energy fed back during continuous operation excessive (regenerative operation)	On individual operation of the drive from supply: - If possible: set up DC-bus connection to other controllers - Use regenerative module - Check drive dimensioning In case of drives with DC bus connection: - Check fuses on all controllers on the DC bus as well as fuses in the brake modules/power supply modules/regenerative modules - Check drive dimensioning

Error number	Error code	Cause	Remedy
30	LU	DC-bus voltage has dropped below the value set in code C0173/000	Check mains voltage. Check mains contactor/fuses. In case of DC supply: check power supply module.
32	LP1	A current-carrying motor phase has failed	Disconnect motor and measure windings. Disconnect supply cable and check for continuity.
		The current threshold is set too low	Increase current threshold in code C0599/000.
		This monitoring is not suitable for - synchronous servo motors - Field frequencies above 480Hz	Deactivate monitoring using code C0597/000 = 3.
50	OH	Ambient temperature too high ($T_U > 40^{\circ}\text{C}$ or 50°C , depending on device type/power reduction)	Leave drive controller to cool down and provide improved ventilation (control cabinet fan?). Check ambient temperature in the control cabinet.
		Heatsinks are heavily soiled	Clean heatsinks and protect against renewed soiling.
		Mounting position of the drive controller	Check the mounting position of the controller and the stipulated minimum distances top and bottom.
51	OH1	Internal temperature > 90 degrees C	Leave drive controller to cool down and provide improved ventilation (control cabinet fan?). Check ambient temperature in the control cabinet
53	OH3	Overtemperature detection by PTC/KTY: motor too hot due to continuous overload/ overcurrent	Check drive dimensioning.
		No PTC/KTY connected	Connect PTC/KTY. Deactivate the OH3 error using code C0583/000 = 3).
54	OH4	Ambient temperature too high ($T_U > 40^{\circ}\text{C}$ or 50°C , depending on device type/power reduction)	Leave drive controller to cool down and provide improved ventilation (control cabinet fan?). Check ambient temperature in the control cabinet.
		Heatsinks are heavily soiled	Clean heatsinks and protect against renewed soiling.
		Mounting position of the drive controller	Check the mounting position of the controller and the stipulated minimum distances top and bottom.
		A value that is too low has been set in the parameters using code C0122/000	Increase value in code C0122/000.
55	OH5	Temperature inside the controller $> C0124$	Leave drive controller to cool down and provide improved ventilation (control cabinet fan?). Check ambient temperature in the control cabinet
57	OH7	Overtemperature detection by PTC/KTY: motor too hot due to continuous overload/ overcurrent	Check drive dimensioning.
		No PTC/KTY connected	Connect PTC/KTY. Deactivate the OH7 error using code C0584/000 = 3).
		A value that is too low has been set in the parameters using code C0121/000	Increase value in code C0121/000.
58	OH8	Overtemperature detection by temperature contact: motor too hot due to continuous overload/ overcurrent	Check drive dimensioning.
		No temperature contact connected	Connect temperature contact to terminals T1/T2. Deactivate the OH8 error using code C0585/000 = 3).
61	CE0	Malfunction on the transfer of control commands over the AIF interface (e.g. PROFIBUS DP)	Fix automation module firmly. Fasten automation module in place.
62	CE1	CAN-IN1 channel is not receiving any data or is receiving erroneous data	Check CAN bus cable (connector X4) Check CAN terminating resistor (on the first and last CAN station) Check configuration of the CAN node sending on CAN-IN1 Check power supply to the CAN node sending on CAN-IN1 Check monitoring time in code C0357/001 and increase if necessary
63	CE2	CAN-IN2 channel is not receiving any data or is receiving erroneous data	Check CAN bus cable (connector X4) Check CAN terminating resistor (on the first and last CAN station) Check configuration of the CAN node sending on CAN-IN2 Check power supply to the CAN node sending on CAN-IN2 Check monitoring time in code C0357/002 and increase if necessary

Error number	Error code	Cause	Remedy
64	CE3	CAN-IN3 channel is not receiving any data or is receiving erroneous data	Check CAN bus cable (connector X4) Check CAN terminating resistor (on the first and last CAN station) Check configuration of the CAN node sending on CAN-IN3 Check power supply to the CAN node sending on CAN-IN3 Check monitoring time in code C0357/003 and increase if necessary
65	CE4	Controller has received too many erroneous messages over the CAN bus and has disconnected itself from the bus	Check baud rate of all nodes Check wiring: - Check bus terminating resistors - Check shield connection on the cables - Check PE connection - Check bus load (e.g. using software tool PCANView) - Reduce baud rate (CAUTION: pay attention to CAN bus cable lengths)
66	CE5	CAN time-out (gateway function C0370)	Check settings in C0370.
70	U15	Undervoltage internal 15 V supply voltage	Check servo axis voltage supply.
71	CCr	Processor is overloaded or program execution problem	Reduce processor load: remove unnecessary function blocks from the processing table (C0465/0xx)
		Heavy interference on the control cables	Lay control cables screened
		Earth loop in the wiring	Wire as per EMC requirements (see System Manuals "9300", p. 4-34)
72	Pr1	Error on transferring the parameter set CAUTION: default setting is loaded automatically!	Set the required parameters and save safe against mains failure in the EPROM using code C0003/000
74	PEr	Error in the internal program execution	Read parameter set and send together with the source project to Lenze for further analysis, consultation with Lenze necessary (service hotline: (+49) 5154 82-1111)
75	Pr0	Invalid data in the EEPROM on loading the parameters from the EEPROM to the RAM CAUTION: default setting is loaded automatically!	Set the required parameters and save safe against mains failure in the EPROM using code C0003/000, then switch mains/shutdown the control voltage once
79	PI	Error on transferring the parameters from the EEPROM to the RAM	Read parameter set and send together with the source project to Lenze for further analysis, consultation with Lenze necessary (service hotline: (+49) 5154 82-1111)
		Use of a parameter set in the GDC software that does not match the device (9300E/ET).	Use parameter set that matches the controller in the GDC software, transfer this again to the controller and save safe against mains failure in the controller using C0003/000, then switch mains/shutdown the control voltage once
82	Sd2	Resolver cable partially or entirely open circuit/disconnected	Check resolver cable for wire breakage Check resolver
		Resolver not fitted/not connected	Disable monitoring via the code C0586/000 = 3, if a resolver is not used
83	Sd3	Encoder cable open circuit	Disconnect cable from X9 and check for wire breakage
		Pin 8 on input X9 open circuit	Apply +5V DC to pin 8 on input X9 OR: Disable monitoring using code C0587/000 = 3
85	Sd5	Current master value on X6 (terminals 1 and 2) less than 2mA	Check cable for wire breakage Check master current value encoder Check setpoint configuration using C0034/000 OR: Disable monitoring using code C0598/000 = 3
86	Sd6	Motor temperature sensor on X8 is providing undefined values	Check cable on X8 for correct seating and wire breakage Disconnect thermocouple (PTC/KTY) and measure resistance (must not be ∞) OR: Disable monitoring using code C0594/000 = 3
87	Sd7	Absolute value encoder with RS485 interface is not sending any data	Check cable on X8 for correct seating and wire breakage Check absolute value encoder for correct function Set supply voltage to 8.0V using C0421/000 Encoder with HIPERFACE interface not connected CAUTION: after rectifying the cause of the error, fully isolate the servo controller

Error number	Error code	Cause		Remedy
88	Sd8	On the sin/cos encoder or the controller (terminal X8) the connector has been removed (open circuit) or is not correctly fitted		Check the connector on the controller (terminal X8) and on the encoder system for correct seating
		In the connecting cable to the sin/cos encoder there is a fault		Check the connector assignment and core assignment on the cable for correct pin assignment
		Fault in the encoder electronics		Replace faulty encoder type.
				Note: Particularly for monitoring the encoder on a synchronous machine set using Code C0580 = 0 the error response "error" (TRIP).
89	PL	Error during rotor position adjustment		
91	EEr	A digital signal linked to the system variable <i>DCTRL_bTripSet_b</i> is carrying the state TRUE		Check signal source in system variable <i>DCTRL_bTripSet_b</i> . The "Eer" error can be reset if this signal source has adopted the state FALSE.
105	H05			Consultation with Lenze necessary (service hotline: (+49) 5154 82-1111) Reset only possible by mains switching, if error cannot be reset: replace controller or send for repair
107	H07	An incorrect power section has been detected during the initialisation of the controller		Consultation with Lenze necessary (service hotline: (+49) 5154 82-1111) Reset only possible by mains switching, if error cannot be reset: replace controller or send for repair
108	H08	Extension board not fitted correctly or not supported by the program		Check the settings in the system controller as to which extension board is selected.
110	H10	Sensor for the heatsink temperature measurement is providing undefined values, an internal thermocouple may be faulty		Consultation with Lenze necessary (service hotline: (+49) 5154 82-1111) Reset only possible by mains switching, if error cannot be reset: replace controller or send for repair
111	H11	Sensor for the internal temperature measurement is providing undefined values, an internal thermocouple may be faulty		Consultation with Lenze necessary (service hotline: (+49) 5154 82-1111) Reset only possible by mains switching, if error cannot be reset: replace controller or send for repair
122	CE11	FIF-CAN1	FIF-CAN1_IN (monitoring time can be adjusted using C2457/1)	Check baud rate of all nodes Check wiring: - Check bus terminating resistors - Check shield connection on the cables - Check PE connection - Check bus load (e.g. using software tool PCANView) Reduce baud rate (CAUTION: pay attention to CAN bus cable lengths)
		CANaux1	CANaux1_IN (monitoring time can be adjusted using C2457/1)	
123	CE12	FIF-CAN2	FIF-CAN2_IN (monitoring time can be adjusted using C2457/2)	Check baud rate of all nodes Check wiring: - Check bus terminating resistors - Check shield connection on the cables - Check PE connection - Check bus load (e.g. using software tool PCANView) Reduce baud rate (CAUTION: pay attention to CAN bus cable lengths)
		CANaux2	CANaux2_IN (monitoring time can be adjusted using C2457/2)	
124	CE13	FIF-CAN3	FIF-CAN3_IN (monitoring time can be adjusted using C2457/3)	Check baud rate of all nodes Check wiring: - Check bus terminating resistors - Check shield connection on the cables - Check PE connection - Check bus load (e.g. using software tool PCANView) Reduce baud rate (CAUTION: pay attention to CAN bus cable lengths)
		CANaux3	CANaux3_IN (monitoring time can be adjusted using C2457/3)	
125	CE14	FIF-CAN	BUS-OFF status FIF-CAN (too many faulty telegrams received)	Check baud rate of all nodes Check wiring: - Check bus terminating resistors - Check shield connection on the cables - Check PE connection - Check bus load (e.g. using software tool PCANView) Reduce baud rate (CAUTION: pay attention to CAN bus cable lengths)
		CANaux	BUS-OFF status FIF-CAN (too many faulty telegrams received)	

Error number	Error code	Cause	Remedy
126	CE15	CAN aux. Communication error in the gateway function (C0370, C0371) via CAN-AUX	Check baud rate of all nodes Check wiring: - Check bus terminating resistors - Check shield connection on the cables - Check PE connection - Check bus load (e.g. using software tool PCANView) Reduce baud rate (CAUTION: pay attention to CAN bus cable lengths)
190	NErr	Drive mechanically overloaded (e.g. active load on hoists excessive)	Check drive dimensioning Check mechanism for stiffness or jamming Increase ramp times to reduce the dynamics
		Torque limit reached	Increase torque limit: - Directly using system variables MCTRL_nHiMLim_a, MCTRL_nLoMLim_a - Indirectly using current limit in code C0022/000 - Check the motor data (codes C0081/000 to C0092/000)
		Fault in the speed feedback system	Check the speed feedback: - Check the parameter setting for the speed feedback system (C0025/000, C0495/000) - Check rotor position and re-adjust if necessary (C0058/000, C0095/000) - Check the wiring/polarisation of the speed feedback system
		Tolerance window C0576/000 set too low	Increase the tolerance window in code C0576/000.
200	NMAX	Active load (e.g. on hoists) excessive	Check current limit in code C0022/000 and increase if possible (CAUTION: pay attention to motor current limit!) Check drive dimensioning
		Torque limit reached	Increase torque limit: - Directly using system variables MCTRL_nHiMLim_a, MCTRL_nLoMLim_a - Indirectly using current limit in code C0022/000
		In case of speed feedback: the actual speed/rotor position is being measured incorrectly	Check selection of the feedback system (code C0025/000) Check the motor data (codes C0081/000 to C0092/000) Check rotor position and re-adjust if necessary (C0058/000, C0095/000)
201	Overrun Task1 Id2	The runtime for the related task is taking longer than defined in the watchdog time (as programmed in the task configuration in DDS)	Reduce the load on the task by moving sub-routines that are not time critical to lower priority tasks, tasks with a greater interval time or to the cyclic task Increase the interval time for the task. Check the arithmetic operation for division by 0. Check your program code for infinite loops (REPEAT/UNTIL, WHILE, FOR loops) and recursive calls and remove these structures. On the use of free CAN objects: reduce the number of telegrams received per unit time with identifiers that do not correspond to the system blocks CAN1, CAN2 or CAN3. In case of activated data consistency: deactivate the default setting for data consistency in DDS (Zielsystem (Target system) menu command) and use the library <i>Lenze32BitTransferDrv.lib</i> .
202	Overrun Task2 Id3		
203	Overrun Task3 Id4		
204	Overrun Task4 Id5		
205	Overrun Task5 Id6		
206	Overrun Task6 Id7		
207	Overrun Task7 Id8		
208	Overrun Task8 Id9		
209	Floatfehler Sys-Task Id0	An error has occurred during a floating point operation in the system task.	Consultation with Lenze necessary (service hotline: (+49) 5154 82-1111)
210	Floatfehler PLC_PRG Id1	An error has occurred during a floating point operation in the cyclic task (PLC_PRG or a POE called).	Check all floating point operations in the cyclic task and sub-routines called by them. Remove/prevent divisions by 0.
211	Floatfehler Task1 Id2	An error has occurred during a floating point operation in the cyclic task (PLC_PRG or a POE called).	Check all floating point operations in the cyclic task and sub-routines called by them. Remove/prevent divisions by 0.
212	Floatfehler Task2 Id3		
213	Floatfehler Task3 Id4		
214	Floatfehler Task4 Id5		
215	Floatfehler Task5 Id6		

Error number	Error code	Cause	Remedy
216	<i>Floatfehler Task6 Id7</i>		
217	<i>Floatfehler Task7 Id8</i>		
218	<i>Floatfehler Task8 Id9</i>		
219	<i>Overrun PLC_PRG Id0</i>	The cyclic task is taking too long or cannot be completed.	Check your program code for infinite loops (REPEAT/UNTIL, WHILE, FOR loops) and recursive calls and remove these structures.
220	<i>NoCredits</i>	The target system does not support the technology libraries contained in the project (e.g. positioning technology, electronic cam).	Check code C2115/000: if no technology units are displayed here, contact your Lenze sales representative or the Lenze service hotline (+49) 5154 82-1111.
230	<i>NoPrg</i>	There is no PLC program saved in the target system for execution.	Transfer suitable PLC program to the target system using DDS (PRO file) or GDLoader (BIN file).
231	<i>Unallowed Lib</i>	The target system does not support a library used in the source project.	Use the library for the target system. Ensure the target system version supports the version of the libraries used.
232	<i>NoCamData</i>	Motion profiles (Cam data) not available	Check the data in the Cam file
240	<i>TxBuffer Overflow</i>	On the usage of the library <i>LenzeCanDrv.lib</i> too many CAN telegrams are being sent. The transmit data memory is full.	Reduce the CAN telegrams sent using the function block <i>L_CanPdoTransmit</i> – Increase the interval time for the task – Select time-controlled instead of event-controlled transmission using the <i>byTransmitMode</i> – Increase the transmission cycle on the input <i>tRepeatTime</i>.
			Increase the transmission rate on the system bus using the code C0351/000.
241	<i>RxBuffer Overflow</i>	On the use of the library <i>LenzeCanDrv.lib</i> too many CAN telegrams are being received and cannot be processed that quickly in the PLC program. The receive data memory is full.	Reduce the number of telegrams received to the free CAN objects that are further processed by function blocks of type <i>L_CanPdoReceive</i> . Call sub-routines in which received, free CAN telegrams are further processed at shorter interval times.
250	<i>NoApplFlash</i>	The data memory for additional data/application data is missing	Replace the actual target system with a suitable target system that contains the memory module for additional/application data. Contact your Lenze sales representative or the Lenze service hotline (+49) 5154 82-1111.
251	<i>ChckErr (ApplRAM)</i>	The checksum for the application data in the RAM does not match the reference checksum.	Contact your Lenze sales representative or the Lenze service hotline (+49) 5154 82-1111.
252	<i>DwnldErr (ApplFlash)</i>	During the transfer of the application data to the non-volatile application data memory an error occurred.	Check the data connection: – Check the bus connectors and terminating resistors for correct seating and correct polarity. – Check the parameters set for the transmission path for freedom from ambiguity (e.g. several CAN nodes with same bus address, baud rate, bus load too high). – If necessary remove nodes from the system bus to reduce the bus load.
			The source file for the additional data is corrupt (impermissible change): re-generate the source data (e.g. LC9 file with profile data) and repeat the transfer.
260	<i>LifeGd Event</i>	The checksum for the application data in the flash EPROM does not match the reference checksum.	Contact your Lenze sales representative or the Lenze service hotline (+49) 5154 82-1111.

11.4.2 Application error messages

Error number	Error code	Cause	Remedy
400	<i>PosLimit Switch</i>	Positive limit switch actuated	Move clear from the positive limit switch: Acknowledge the error message. Move the drive in the negative direction, e.g. using manual jogging until clear of the positive limit switch
		Limit switch signal not connected to terminal E2	Set the level inversion for the digital input E2 to inverse logic (C0114/002 = 1) or Hard wire the terminal E2 to 24V DC
401	<i>NegLimit Switch</i>	Negative limit switch actuated	Move clear from the negative limit switch: Move the drive in the positive direction, e.g. using manual jogging until clear of the negative limit switch
		Limit switch signal not connected to terminal E1	Set the level inversion for the digital input E1 to inverse logic (C0114/001 = 1) or Hard wire the terminal E1 to 24V DC
402	<i>PosSWLimit</i>	Positive software limit position has reached/passed	Move the drive in the negative direction to the permissible traversing range. Set the limit value for the positive software limit position (C3223/000) to a higher value.
403	<i>NegSWLimit</i>	Negative software limit position has reached/passed	Move the drive in the positive direction to the permissible traversing range. Set the limit value for the negative software limit position (C3224/000) to a lower value.
405	<i>FollError</i>	The drive cannot follow the setpoint, the actual following error is greater than the value set in the parameters in C3218/001.	- Increase current threshold in C0022/000 and as a result increase the maximum torque (CAUTION: pay attention to motor heating!) - Reduce dynamic requests (accelerations/decelerations) - Increase following error limit in C3218/001 - Check drive dimensioning as necessary
410	<i>DriveNotRdy</i>	Automatic operation has been interrupted by a controller inhibit set by the user.	- Controller inhibit can be triggered using terminal 28, a bus system (AIF or CAN bus), the control codes C0040/000 and C0135/000 or the system variables <i>DCTRL_bCinh1_b</i> and <i>DCTRL_bCinh2_b</i>. - Re-enable the target system, acknowledge the error and return the drive to the required operating mode.
419	<i>StateBusError</i>	Another drive axis connected to the state bus has a fault. All controllers connected are set to the error status <i>StateBusError</i> using the terminals ST.	- Identify the drive in the state bus interconnection that has caused the error, rectify the cause of the error and reset the error on this drive - Then reset the state bus error on all other drives that are connected to the state bus.
420	<i>OU_MCTRL Error</i>	Automatic operation has been interrupted by a brief overvoltage on the DC bus.	An overvoltage on the DC bus is detected using the system variable <i>MCTRL_bOverVoltage_b</i>: if this error occurs proceed as follows: - Acknowledge the error. - Set the drive back to the required operating mode. To determine the cause of the error, refer to error message no. 20 (OU) in the system error messages.
421	<i>LU_MCTRL Error</i>	Automatic operation has been interrupted by undervoltage on the DC bus.	An undervoltage on the DC bus is detected using the system variable <i>MCTRL_bUnderVoltage_b</i>: if this error occurs proceed as follows: - Acknowledge the error. - Set the drive back to the required operating mode. To determine the cause of the error, refer to error message no. 30 (LU) in the system error messages.

11.4.3 User-defined error messages

Error number	Error code	Cause	Remedy
500	<i>UserError</i>	The related input <i>bUserError</i> on the function block <i>ErrorHandling</i> has been activated by a positive edge. The signal causing the error has been programmed on the user's side.	Evaluate the logic providing the signal on the input <i>bUserError</i> .