User Manual Flying Saw

applications
& TOOLS

SIMOTION Flying Saw V2.1

SIEMENS

SIROT- ID: 37841424

We reserve the right to make technical changes to this product.

Copyright

General Notes

Note The standard applications are not binding and do not claim to be complete regarding the circuits shown, equipping and any eventuality. The standard applications do not represent customer-specific solutions. They are only intended to provide support for typical applications. You are responsible in ensuring that the described products are correctly used. These standard applications do not relieve you of the responsibility in safely and professionally using, installing, operating and servicing equipment. When using these standard applications, you recognize that we cannot be made liable for any damage/claims beyond the liability clause described. We reserve the right to make changes to these standard applications at any time without prior notice. If there are any deviations between the recommendations provided in these standard applications and other Siemens publications - e.g. catalogs - then the contents of the other documents have priority.

Warranty, liability and support

If the application is provided free of charge the following shall apply:

We shall not be liable for the information contained in this document.

Any and all further rights and remedies against Siemens AG for whatsoever legal reason, shall be excluded; this shall refer in particular to claims for loss of production, loss of use, loss of orders or profit and other direct, indirect or consequential damage.

The aforesaid shall not apply if liability is mandatory, e.g. in accordance with the Product Liability Act, in cases of intent, gross negligence by directors and officers of Siemens AG or in the case of willful hiding of a defect.

These limitations of liability shall also apply for the benefit of the Siemens AG's subcontractors, suppliers, agents, directors, officers and employees.

This Contract shall be subject to German law if customer’s place of business is in Germany. If customer’s place of business is outside of Germany the Contract shall be subject to Swiss law. The application of the UN Convention on Contracts for the International Sale of Goods (CISG) shall be excluded.

If the application is provided in return for payment the alternative shall apply which fits the respective business case:

/ Alternative 1: (internal business)
If not explicitly stated otherwise below, the "Terms and Conditions for Deliveries and Services for Siemens Internal Transactions", valid at the time of sale, are applicable.

/ Alternative 2: (domestic business of Siemens AG)
If not explicitly stated otherwise below, the "General License Conditions for Software Products for Automation and Drives for Customers with a Seat or registered Office in Germany", valid at the time of sale, are applicable.

/ Alternative 3: (direct export business of Siemens AG)

If not explicitly stated otherwise below, the "General License Conditions for Software Products for Automation and Drives for Customers with a Seat or Registered Office outside of Germany", valid at the time of sale, are applicable.

It is not permissible to transfer or copy these standard applications or excerpts of them in unmodified form without first having prior explicit authorization from Siemens Industry Sector in writing.

For questions regarding this application please contact us at the following e-mail address:
mailto:tech.team.motioncontrol@siemens.com

Qualified personnel

In the sense of this documentation qualified personnel are those who are knowledgeable and qualified to mount/install, commission, operate and service/maintain the products which are to be used. He or she must have the appropriate qualifications to carry-out these activities

e.g.:

- Trained and authorized to energize and de-energize, ground and tag circuits and equipment according to applicable safety standards.
- Trained or instructed according to the latest safety standards in the care and use of the appropriate safety equipment.
- Trained in rendering first aid.

There is no explicit warning information in this documentation. However, reference is made to warning information and instructions in the Operating Instructions for the particular product.

Reference regarding export codes

AL: N

ECCN: N

Foreword

Standard SIMOTION application

A standard SIMOTION application comprises the following components:

- One or several software objects or code blocks with defined interfaces that can be simply integrated into other software projects – without requiring any significant programming – in order to fulfill a precisely defined technology task there. **(Core functions)**
- A software project based on a demonstration case to show the functionality and possible uses of the standard application - including the associated WinCCflex screen for demonstration. **(Demonstration project)**
- A document to describe the functionality, background information and handling of the standard application. Further, its use as demonstration model is explained. **(Description)**

Document structure

The documentation of this application is sub-divided into the following main section:

Section	Description	Note
Prerequisites and objectives	In the first section you can obtain an overview of this standard application. This Section explains the prerequisites and the objective when using this application. Some of the uses of this standard application are explained as well as situations where this application cannot be used.	
The application example as demosystem	Section “Application example as demosystem” is interesting if you wish to use this standard application for demonstration purposes. Here, you are provided with information about how you can download this application from your PC/PG to the demonstration case step-by-step and how it then used.	
Integrating the core functions	Section “Integrating the core function” provides you with all of the necessary steps to integrate the core functions of the standard application into your user project. Preparations and parameterizing operations are explained. Further, you are also told how to integrate the core functions into your application step-by-step. In addition, tips are provided on how to use the core functions.	
Program description of the technology template	Section “Program description” is interesting if you wish to expand/adapt the functionality of the core functions provided for your particular application.	

Appendix	Section “Appendix” provides you with detailed information including a detailed fault/error description, a description of a test program for the core standard application functions as well as references. A feedback sheet is also provided which you can use to give us your comments and suggestions on this document.	

Index of contents

Prerequisites and objectives

Content

Section A will gives a detailed overview about each implemented function or function block which is ready to apply.
The application presented in this manual provides some use-cases. Further, some applications will be listed for which this standard application cannot be used. In addition, the performance limits of this standard application will be shown.

Objective

The part of this document should provide the user with the following information:

- The objective and purpose of this standard application
- List several applications
- Indicate the performance limits of this standard application.

Subjects discussed

Chap.	Title	Contents
1	Basic information	The necessary prerequisites to use the standard “flying saw” application. The use of this application. Software components and code blocks of the standard “flying saw” application.
2	Uses	Tasks and properties of the core functions of the standard “flying saw” application Exclusions and restrictions Hardware components required
3	Structure and function	Type of design and physical quantities for which the “flying saw” core function can be used. Type of cams that the “flying saw” core functions calculate and used. Operating states at the FlyingSaw FB.

1 Basic information

1.1 Prerequisites

1.1.1 Target group

The Converting Toolbox standard application “Flying Saw” is intended to provide SIMOTION programmers and users with help when engineering and programming flying saw machinery.

1.1.2 Knowledge base

To use this standard application, please make yourself familiar with SCOUT, technology objects and technology function calls in SIMOTION.

This document does not discuss these a.m. topics and limits itself to provide information and data how to use this standard application.

1.1.3 Technical environment

This Flying Saw application can be used unmodified, only at a SIMOTION D- and SINAMICS-demonstration case.

1.2 Objective and purpose of this standard application

1.2.1 Task description

A continuous web - fed with an infeed unit - shall be cut into identical parts and sections. Preventing any impacts on the cut the cutting equipment must be synchronized on the line speed of the web. This allows the web to be cut “on the fly”.

Why processing / machining “on the fly”?

"On the fly" processing / machining is required if:

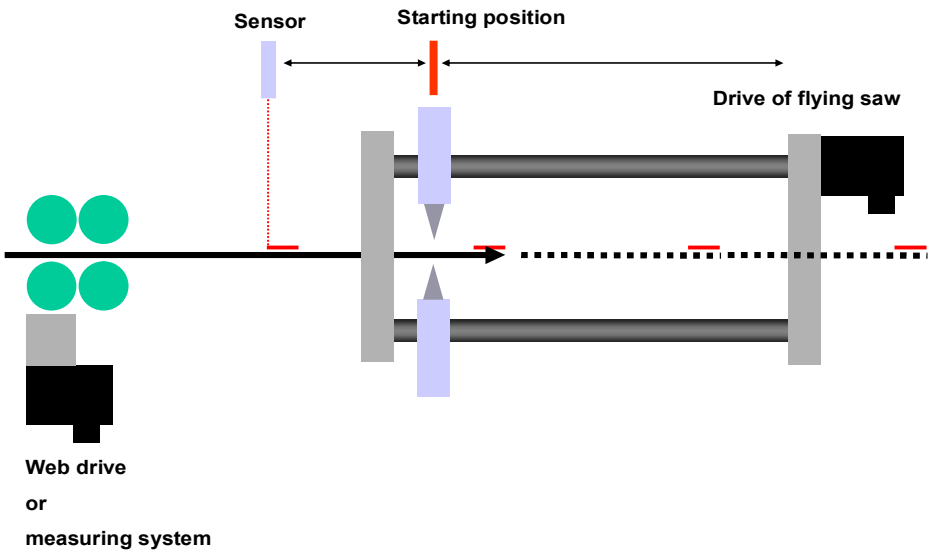
- Stopping the material web due to sub-processes which would lead to quality loss, increased waste or lower productivity. (e.g. extrusion line)
- It is impossible to stop the material web because associated deceleration- or acceleration-forces cannot be controlled.
- The individual processing / machining steps are too different so that a common processing step would take too much time and therefore would make the process ineffective.

Principle design of a "flying saw"

The material is fed-in through a draw unit with a constant velocity.

The position of the cut on the material web is either detected with a print mark sensor or an adjustable length setpoint is given by the machine controller. At the same time the synchronous position is defined, wherefrom the cutting equipment will match with the material speed and the cutting sequence can be started. After cutting has been completed, the cutting equipment moves back into start position where it is waiting for the next cutting position.

Fig. 1-1: Principle of operation of the flying saw



1.2.2 Solution using the standard application SIMOTION Flying Saw

The SIMOTION Flying Saw application introduced in this document helps the user to meet the assigned requirements for developing a flying saw solution as quickly as possible.

The standard application includes a function block which controls the flying saw and implements the motion control part via parameters corresponding to the particular application.

The user program only has to control the material feed and to supply the core function with the correct parameters.

Additional function of the standard application

Furthermore the standard application provides the optional feature to create a gap with the cutting equipment between the cutted pieces

This function can be activated by setting parameters in the core function.

1.2.3 Advantages of the standard application SIMOTION Flying Saw

The SIMOTION Flying Saw application offers the following advantages:

Quick programming

The standard application SIMOTION Flying Saw enables simple and quick implementation of flying saw functionalities.

The core functions provided in the standard application can be copied into the user's application. Addional and necessary engineering steps will be explained in this maual.

Automatic motion control

Using the technology functions of SIMOTION the core motion control functions of the standard application can be realized The user only programs an execution sequence which corresponds to required machine function.

Possibility of adaptation

The standard application contains all source codes and comments. Thus existing core functions can quickly and simply be extended with customized functions.

1.3 Components included in the standard application

The standard SIMOTION Flying Saw application is implemented as a library.

This library is part of the two projects which will be delivered with the standard application software package.

The first project can be used for a demonstration machine, on a SIMOTION D demonstration case and on a PC with WinCC Flexible 2008 for visualization.

The demonstration program fulfills the following tasks:

- Operating mode manager for the complete (demonstration) machine
- The control of the (demonstration) machine
- The core functions of the flying saw functionality with absolute gearing
- All of the machine functions that are relevant for the demonstration case environment can be simulated
- The (demonstration) machine is displayed on the WinCCflex screen

The second project (TestFB) enables the user to get quick access to the standard application Flying Saw (FB-FlyingSaw) as well as obtaining know how for commissioning and diagnosting the SIMOTION Flying Saw application.

The TestFB program is limited to the following tasks:

- Presentation of FB FlyingSaw from a user's point of view.
- Controlling of all interfaces of the standard application.
- Visualization of all relevant data of standard application.
- Graphical display of synchronous operations.

In this way the core functions of the standard SIMOTION Flying Saw application can quickly and simply be integrated into user projects.

2 Areas of use

2.1 Applications

2.1.1 Available controls

The standard SIMOTION Flying Saw application can be applied on any SIMOTION control system.

2.1.2 Tasks that can be implemented using the core functions

The core functions provide interfaces for “on the fly” processes at:

- Cutters
- Perforators
- Embossers
- Sealers
- Saws
- Drilling machines
- Painting machines
- Stamping units

Definition

In the following text, the **flying saw** term is used to represent all of the other conceivable machine versions.

The term **saw** is used for all parts of the machine where direct contact is made to the material when “cutting”.

2.1.3 Properties and features of the core “flying saw” functions

The following properties and features were implemented and can be used in a user program:

- **Precisely synchronizing at a specific position**
  The flying saw is synchronized with the material, so it reaches its velocity at a parameterized position.
- **Flying axis**
  During cutting, the flying saw and the material move with the same velocity.
- **Adjustable rest position of the flying saw**
  The rest position of the flying saw can be adjusted between two cuts using a parameter.
- **“Cutting” at a print mark**
  The cutting position is detected using a probe and print mark.
- **“Cutting” to length**
  The cut position is determined by comparing the length of material that has been transported since the last cut with the selected cut length.
- **Immediate cut**
  A cut is initiated using an operator function, independent of the material length that has been passed since the last cut.
- **Creating a gap**
  After a cut has been made, the cutted material will be moved with the cutting equipment (saw) in the same direction of the material flow until a parameterizable distance (gap) to the non-cutted material has been achieved. After this step the cutting unit (saw) returns to its start position.
- **Gear factor**
  A gear factor can be set between the measurable material web velocity and the material web velocity effective when cutting.

  E.g. this feature is required when cutting wound paper cores (formers). The wound material is skewed and therefore the measured material web velocity (with a measuring wheel) does not match to the resulting core (former) velocity.

2.2 Principle of operation

2.2.1 Possibilities for determination the cutposition

To determine the cutting postion the block has two different techniques:

CP_CALCULATED:	Free cutting. Cut position is calculated by breaking material web down by cutlength
CP_MEASURED:	Detection of cutpositions at a measuring point including storage of the measured values in a buffer

CAUTION	**The selection can be changed only when the application is switched off and the material axis or leading value is switched off.**

CP_CALCULATED

As a matter of principle the material is allways cut at position „cutlength“.

Precondition:

- The material axis is generated as a linear axis **without** modulo length.

First cut at standstill:

- The material position is set to „cutlength“.
- After the cut carried out, the „cutlength“ will be subtracted from the actual material position.
  (→ The result is: position = 0 when material keeps at standstill!)

First cut out of the motion:

- The material position is set to a position < „cutlength“ according to the synchronization requirements, so that the cutting equipment is synchronized when the material position has reached the „cutLength“.
- After cut and positioning back to startpos is completed, the „cutlength“ will be subtracted from the actual material position.

All further cuts:

- After the cut and positioning back to startpos has been completed, the „cutlength“ will be subtracted from the actual material position.

”OnTheSpotCut“ is possible at any time when the Flying Saw (cutting equipment) is at standstill.

CP_MEASURED

The positions, onwhich the material shall be cut, are detected with a measuring sensor in front of the Flying Saw then extrapolated to the synchronizing position and memorized in a ring buffer. The Flying Saw application will read these position values from the buffer, starts the synchronizing procedure and deletes the postion values when the cut has been completed at that position.

Precondition:

- The material axis is generated as a linear axis **with** modulo length.
- A measuring sensor is mandatory when CPM_DETECTED is activated.
- A ring buffer system is activated.

CPM_DETECTED:

- A measuring sensor is mandatory!
- With the activation of the automatic mode the position of the master value is set to position = 0.
- The application is waiting for the first measuring value, extrapolates this value to the synchronizing position and writes it into the ring buffer.
- Next the measuring system will be prepared for the next measuring value by extrapolation with the cutlength and activated with a sensivity range.
- Once a measuring value is detected this value will be extrapolated to the synchronizing position and written into the ring buffer.
- In case a measured value was not detected, an error counter will be incremented. The last measured value will be increased by the cutlength and extrapolated to the synchronizing position. This value will be written as the new measured value into the ring buffer.
- After the next measured value has successfully been detected the error counter will be set to zero. The counter overflow is monitored.

CPM_SIMULATED:

- No measuring sensor is necessary.
- With the activation of the automatic mode the position of the master value will be set to position = 0.and automatically the first „measured value“ will be extrapolated to the synchronizing position and written into the ring buffer.
- All further „measured values“ will be determined by adding „cutlength“ and extrapolation to the synchronizing position. The resulting value will be written into the ring buffer.

“OnTheSpotCuts” to this effect are not possible. As all cuts have to be written into the ring buffer, these values can only be executed after a certain material run time which is determined by the synchronizing length and the distance between the sensor and the Flying Saw.

2.2.2 Synchronizing

The user can select one of three synchronization types:

- Time related
  - BY_TIME

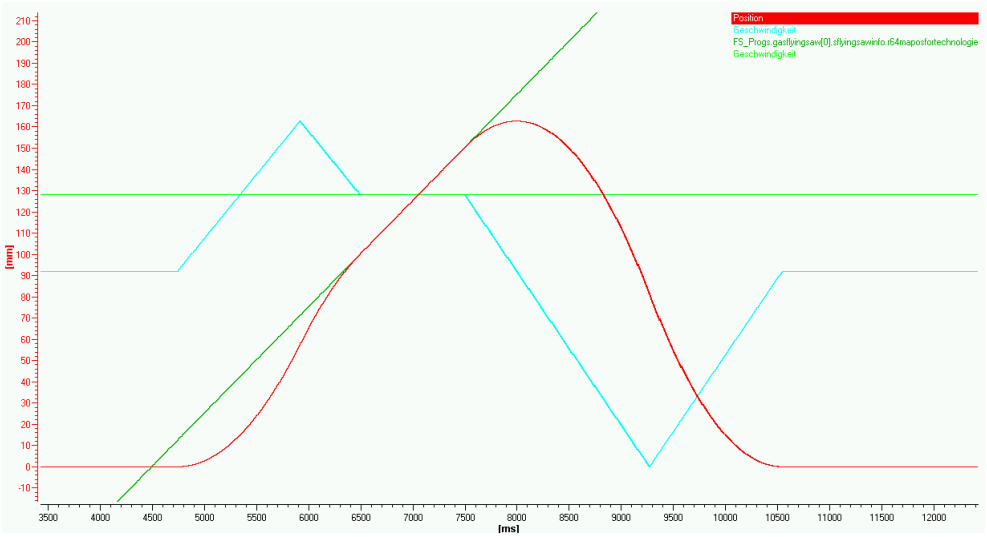
- Position-related
  - BY_LEADING_AXIS_VALUE
  - SYMMETRIC

Time-related synchronization

Synchronization motion is obtained from the dynamic values parameterized at the axis (velocity, acceleration and jerk).

With dynamic values big enough, the saw will wait for the start of the synchronization motion and then accelerates to a velocity higher than the material velocity in order to catch-up with the synchronous point.

Fig. 2-1: Synchronization with overshooting velocity



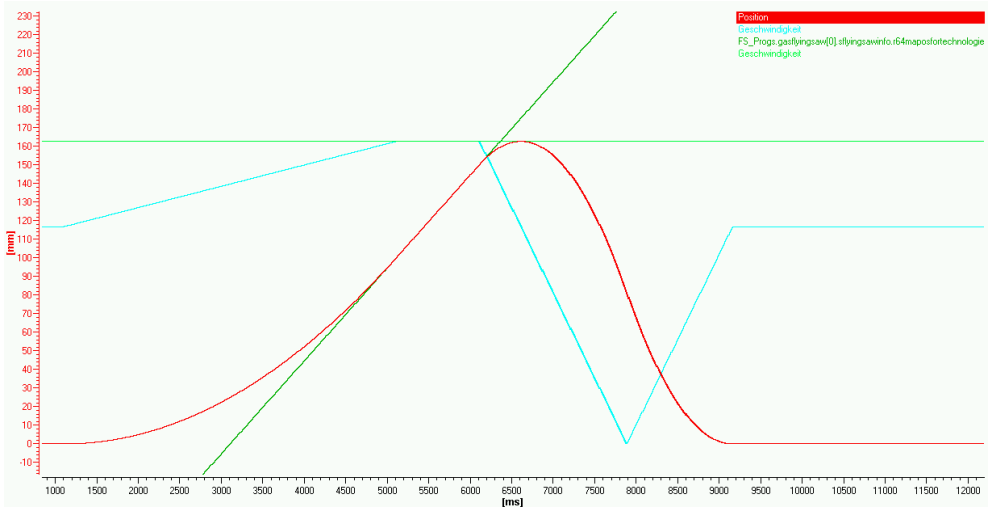
With dynamic values too small, the saw won't be able to start and the error message „Sync not possible“ appears.

Position-related synchronization

The characteristic of the synchronizing movement is dependent on “SyncLength” entered.

If SYMMETRIC is selected, the distance is dimensioned in that way that symmetrical synchronization will be achieved without any overspeed. The cutting equipment starts with the synchronization motion so that the synchronous point only “catches-up” when the material velocity is reached. In this case the “SyncLength” is set set to 2 x (SyncPos – StartPos).

Fig. 2-2: Synchronization with continuous acceleration



If the „SyncLength“ is selected too short, the saw will not be able to start and the error message „Sync not possible“ appears.

If the „SyncLength“ is selected too long or the saw doesn’t reach startposition in time, the saw will not be able to start and the error message „for Sync to late“ appears. In this case the saw will cut automatical the next reachable integer multiple of cutlength. (e.g. cutlength = 400mm, the next possible cutlength will be 800mm, 1200mm …!) If the operator doesn’t agree with these lengths, he can use the “on the spot cut” function.

2.3 Application environment

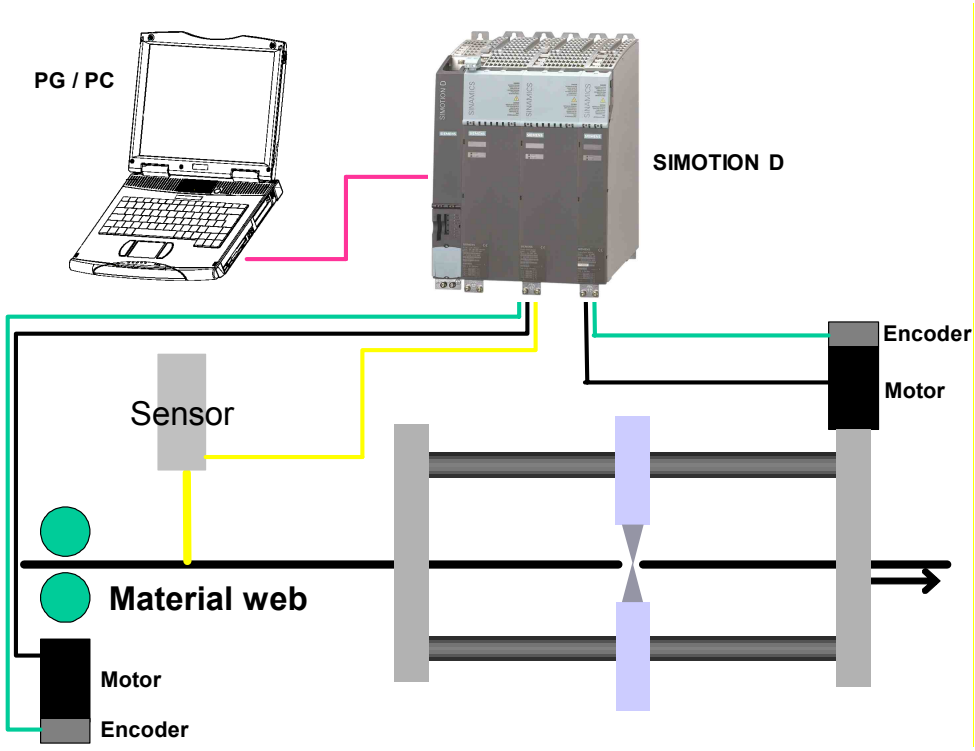
The standard application SIMOTION Flying Saw can be used at

- Real machine
- Demo application (e.g demo case or drives/motors without mechanical coupling to a real machine in test environment!)

CAUTION	**The User has to select the real application environment by using the bit sUsersInterface.boRealMachine.**

2.3.1 Real Machine (sUsersInterface.boRealMachine = TRUE!)

Fig. 2-3: Application environment of the standard application SIMOTION Flying Saw Real Machine

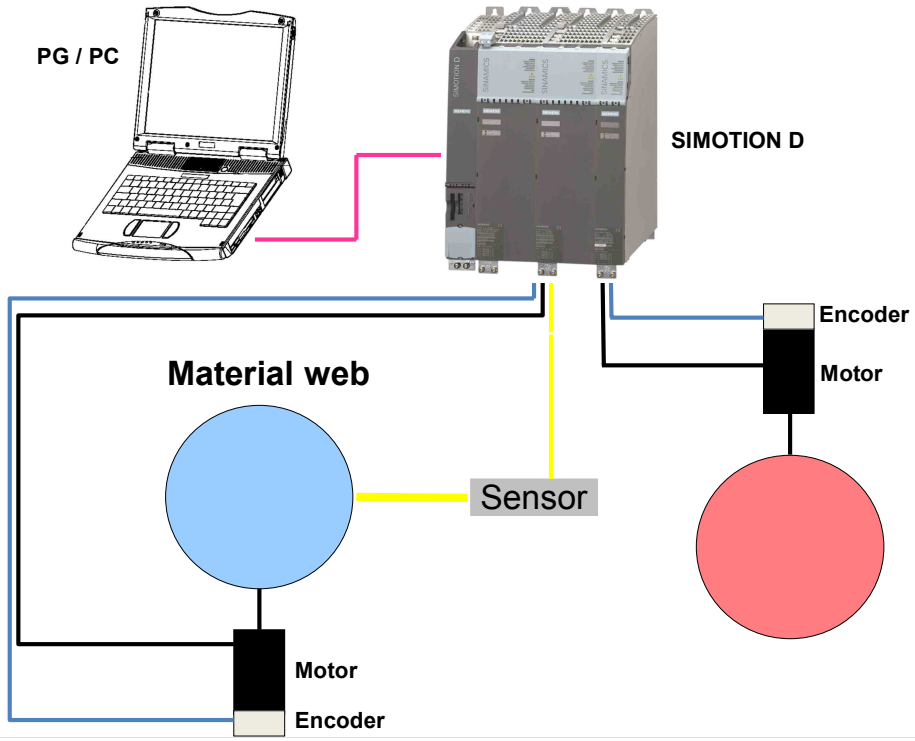


A SIMOTION system can be used to control the material feed and the drive of the flying saw. As an alternative, it is also possible to couple the flying saw via a machine encoder.

In this environment modulo parameters of the flying saw axis will be deactivated.

2.3.2 **Demo application (**sUsersInterface.boRealMachine = FALSE!)

Fig. 2-4 Application environment of the standard application SIMOTION Flying Saw Demo application



In this environment the FlyingSaw axis will be configured as a modulo axis with:

- modulo length = EndPos + 500.0
- startposition = – (distance to sensor +50.0).

Thereby at the demo environment no homing of the FlyingSaw axis is.required.

3 Structure and function

3.1 Design of the "flying saw"

3.1.1 General design

The bases of the flying saw application for example consists of a spindle axis, on which the cutting unit is mounted.

The following physical variables of the flying saw are in use (see white boxes):

Fig 3-1 Physical variables of the application Flying Saw

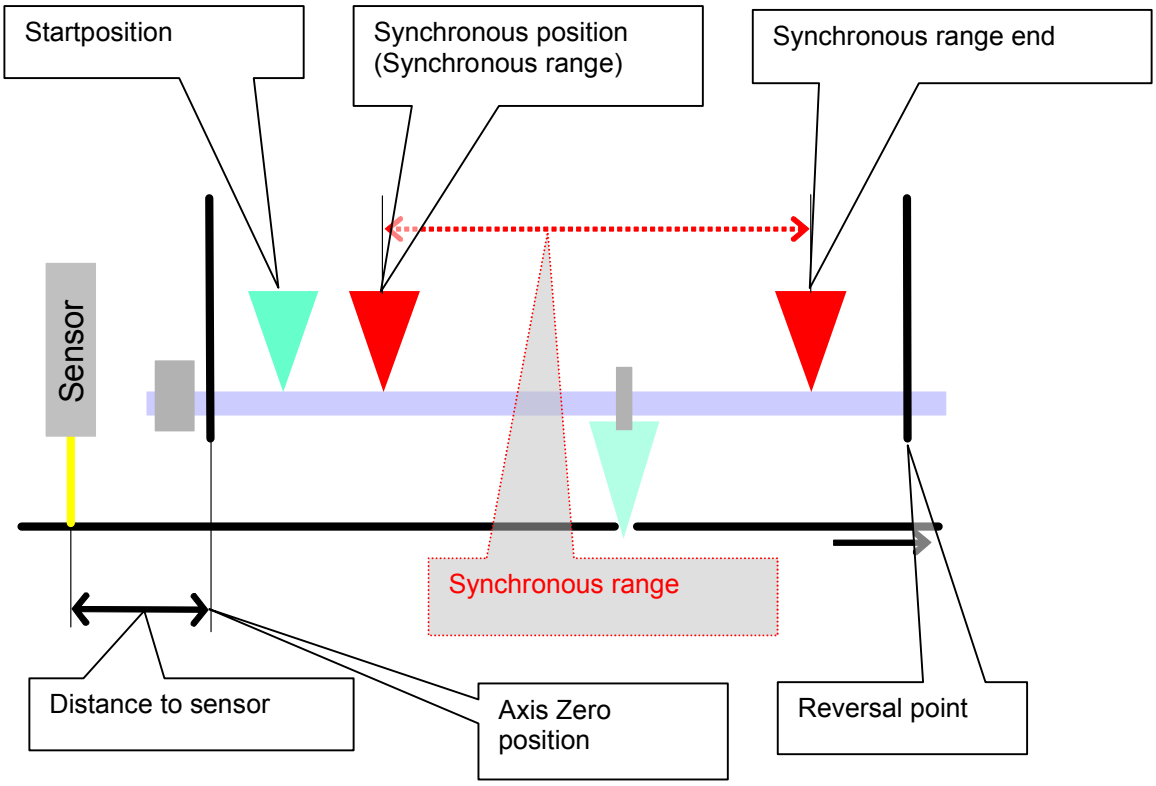


Table 3-1: Explanation of the parameterizable physical quantities

Physical quantity	Description
Distance to Sensor *sUsersInterface.* *r64DistanceToSensor* [mm]	Distance between the axis zero point of the flying saw and the sensor to detect the material and print mark
Startposition *sUsersInterface.* *r64StartPos* [mm]	Start position of the cutting unit referred to the axis zero point of the flying saw.
Synchronous position *sUsersInterface.* *r64SyncPos* [mm]	Position at which the cutting unit reaches the specified material position and material velocity. Starting point of the synchronous range
Synchronous range end *sUsersInterface.* *r64EndSyncPos* [mm]	End of the synchronous range. If cutting has still not been completed at this position, then the cutting is interrupted.
Reversal point *sUsersInterface.* *r64EndPos* [mm]	End point of the traversing range of the flying saw (e.g. software limit switch!)

Synchronous range

The synchronous range and therefore the cutting range are located between the synchronous position (***sUsersInterface.r64SyncPos***) and synchronous range end (***sUsersInterface.r64EndSyncPos***).

3.2 Operating modes of the application SIMOTION Flying Saw

The application automatically controls the flying saw axis to implement the required level of functionality.

This functionality is sub-divided into five operating states so that the sequences at the flying saw are implemented as a result of the modes or also as a result of the transitions between the states.

This means the user only influences the flying saw by specifying the required operating mode with the required parameters.

Fig. 3-2: Operating modes

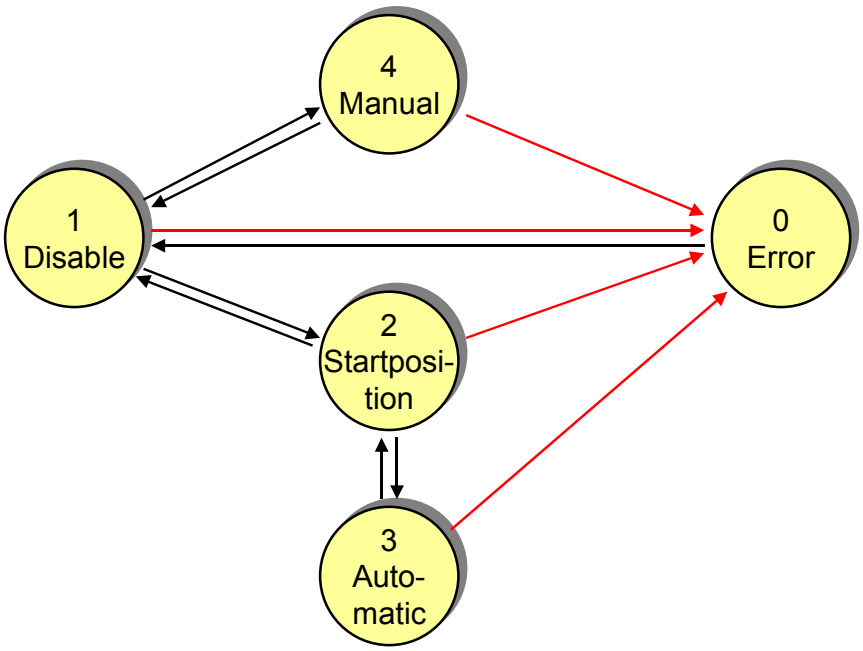


Table 3-2: Operating modes of the Application SIMOTION Flying Saw

Operating mode	Description
ERROR (0)	An error was detected and the flying saw was brought into a safe state. The error is still present and can be evaluated by the user. The axis of the flying saw is stopped and deactivated.
DISABLE (1)	The flying saw is ready for use, but is in the safe operating state. All errors that are available have been successfully acknowledged. The axis of the flying saw is in the state after being reset, deactivated and stopped (disabled).
STARTPOSITION (2)	The flying saw is located at the start position and is ready for synchronization to the material web. The axis of the flying saw is activated. The axis waits in the START POSITION operating mode.
AUTOMATIC (3)	The flying saw has been synchronized to the material web, and is cutting the material as specified.
MANUAL (4)	The flying saw is in the manual mode, the user can manually move the axis. The axis of the flying saw is activated. The axis waits in the MANUAL operating mode.

These operating modes are available in the user program to implement machine functions and can be transferred to the *FBFlyingSaw* function block via input parameters.

The function block displays the currently reached state using output parameters.

The application example as demonstration system

Contents

All of the necessary steps to commission the standard application SIMOTION Flying Saw as demonstration system are explained in Section B.

Preparations and parameterizing operations are also explained. Further, you are told how you can use the WinCCflex man-machine interface (screen) of the application example step-by-step.

Objective

Section B of this document provides the reader with the following

- The prerequisites to use this standard SIMOTION application as a demonstration system
- Preparations and parameterizing operations are explained
- Describes the steps necessary when presenting this standard application.
- Provides tips for using this standard application.

Subjects discussed

Chap.	Title	Contents
4	Installing the hardware and software	Safety information and instructions Components and their interconnections required for the presentation Installation of standard SIEMENS software Downloading the user program in SIMOTION D435 Downloading drive parameters in SINAMICS
5	Operator control of the application example	Brief instructions on how to use the demonstration system Detailed operating instructions

4 Installing the hardware and software

4.1 Regarding your safety

4.1.1 Safety information and instructions

Pictograms, signal words and text

Every piece of safety information/instruction in this document is designated by text graphics – comprising pictogram and signal word, and supplemented by explanatory text. A clear classification according to the degree of the potential hazard is provided as a result of the combination of pictogram and signal word. Safety information/instructions are provided in front of the information regarding activities to be executed.

Classification

There are **three different stages** regarding safety information/instructions. These are designated **by the same pictogram**. They differ by the signal word.

Danger

This safety information/instruction indicates an immediate hazard. If the information/instruction is not carefully followed, this results in severe bodily injury or even death.

Warning

This safety information/instruction indicates a potential hazard. If the information/instruction is not carefully followed, this can result in severe bodily injury or even death.

Caution

This safety information/instruction indicates a potentially hazardous situation, which can result in slight to average bodily injury. This pictogram/text word can also warn about potential material damage.

4.1.2 Responsibilities of the operator

Correct use

The correct use of the application components exclusively relates to the open-loop and closed-loop control of test set-ups that were adapted to the power/performance of the application components. In order that the application functions perfectly, the required standard SIMATIC components as well as also the necessary hardware and software components must be installed.

The company/person operating the system may only make changes to the application components after having received written authorization from the suppliers.

Misuse

The following are considered to be misuse:

- Inadmissible loads applied to the application components.
- Any application deviating from the use specified above, or applications that go beyond the specified use.
- Non-observance of the safety information and instructions.
- If faults that could have a negative impact on the safety are not immediately resolved/removed.
- Any changes/modifications to equipment/devices that are used to ensure perfect function and operation, unrestricted use as well as active or passive safety.
- If recommended hardware and software components are not used.
- If the application components are not in a perfect technical condition are not operated conscious of safety and hazards, and not taking into account all of the instructions provided in the documentation.

The manufacturer assumes no liability for incorrect use (misuse).

Responsible for monitoring

The company or person operating the system is responsible in continually monitoring the overall technical status of the application components (defects and damage that can be externally identified as well as changes in the operating behavior).

The company/person operating the system is responsible in ensuring that the application is only operated in a perfect state. He must check the state of the application components before they are used and must ensure that any defect is removed before commissioning.

Qualification of personnel

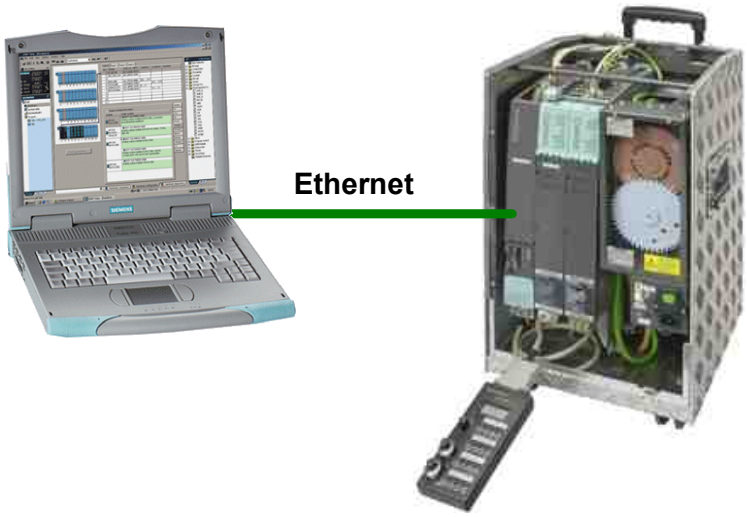
The operating company/person may only deploy trained, authorized and reliable personnel. In so doing, all safety regulations must be carefully observed.

Personnel must receive special instructions regarding the hazards/dangers that can occur.

4.2 Hardware structure and mounting/installation

Overview

Fig. 4-1: Hardware components (without power cable!)



The demonstration project can be commissioned using a standard SIMOTION D435 demonstration case and a PC connected via Ethernet.

Table 4-1: Hardware components

Hardware element	Diagram	Order No./MLFB and functions
Training case, SIMOTION D435 with SINAMICS S120		
SIMOTION D demonstration and training case		6ZB2 470-0AE00 The SIMOTION D training case comprises standard components (SIMOTION D435, two SINAMICS axes with motors) and has two axes. These are used to demonstrate the application. The case is already pre-configured and connected-up. It only has to be connected to the HMI system via PROFIBUS.
Communications		
Ethernet Cross Link Cable		The cable is used to establish a connection between the HMI system (PG/PC) and the SIMOTION D435 training case.
HMI system		
PG/PC with MPI interface		- The PG/ PC is used as the HMI display screen.

Procedure

Please proceed as follows to configure and install the hardware components for the application example:

Table 4-2 Hardware configuration and mounting

No.	Action	Comment
1	Connect the Ethernet interface of your PG/PC with the lower Ethernet interface IE2 (contact X130) of SIMOTION D435 with an Ethernet Cross-Link cable.	
2	Connect the SIMOTION D training case to the power supply.	
3	Power-up all of the equipment/devices.	

4.3 Installing the standard SIEMENS software

Note If the application is only to be used to demonstrate and present, then it is only necessary to install WinCC Flexible 2008 Runtime with 256 power tags.

Minimum required releases

Table 4-3 Versions

Component	Version.
STEP 7	V5.5.03.42
SIMOTION SCOUT	V4.3.1.17
WinCC Flexible	2008 SP3 HF3
WinCC Flexible Runtime	2008 SP3 HF3

4.4 Downloading the user program and parameterizing the drive in the SIMOTION D demonstration case

4.4.1 De-archiving the SIMOTION project

- Open SIMOTION SCOUT
- De-archive the SIMOTION project and open it using SIMOTION SCOUT

4.4.2 Resetting SIMOTION D435 to the factory settings

In order to obtain a fixed starting point for the description on how to download the user program into the demonstration case, restore the factory setting at the demonstration case as described below:

- Power-down the demonstration case
- Set the mode switch SIMOTION D435 to setting 3 (MRES)
- Power-up the demonstration case
- When RDY lights green and STOP lights orange, set the SIMOTION D435 mode switch to the 0 position (RUN)
- RDY and RUN are green

If the factory setting has been established, (restored), the SIMOTION D435 has the standard IP address 169.254.11.22 at the Ethernet interface IE2 (X130).

4.4.3 Setting the PG/PC interface

Please proceed as follows to set the PG/PC interface:

- In SIMOTION SCOUT open the interface configuration using EXTRAS → SET PG/PC INTERFACE...

- In the area “Interface Parameter Assignment”, select your Ethernet card/interface of the PG/PC with “TCP/IP → .....”.
- Acknowledge any (possible) alarm with Yes.
- Acknowledge your changes with OK.
- Return to SIMOTION SCOUT and open the network configuration NETPRO using the button or the menu “PROJECT” → OPEN NETPRO.
- Open the properties by double clicking on PG/PC (1).
- The ETHERNET interface being used must be able to be found under the “Assignment” tab (this is already automatically selected). If this is already the case, select this and set the checkmark for S7ONLINE access **active**.
- Please select “Compile and check evrything”
- Close any messages that may be displayed regarding alarms.
- Before you load the project, please close NETPRO and set the Ethernet address of your PG/PC.

4.4.4 Setting the Ethernet address of the PG/PC

Proceed as follows to set the Ethernet address of the PG/PC:

- Open the network connection window of your PG/PC and select the network connection that is used to SIMOTION and open its properties (righthand mouse click → PROPERTIES or select the symbol and then FILE → PROPERTIES).
- In the area “This connection uses the following items” select the protocol “Internet Protocol (TCP/IP)” and open its properties.
- The Ethernet interface IE2 (X130) of SIMOTION D435 has as standard, the IP address 169.254.11.22.
- Select “Use the following IP address” and enter the IP address **169.254.11.23**.
- As “Subnet mask” enter **255.255.0.0**.
- The address set above must be identical to the address for the PG/PC. Check this address by selecting the PG/PC interface in NetPro and if required, adapt this
- Confirm your changes by pressing on **OK** twice.

4.4.5 Loading the hardware configuration after the factory setting

To load the hardware configuration, please proceed as follows:

- Go into the hardware configuration and press the button Load to target device
- Acknowledge with **OK**
- Press on **No** and do not execute a new start
- Close the hardware configuration and change to SIMOTION Scout

4.4.6 Loading the SIMOTION project of the standard application

Proceed as follows to load the SIMOTION project:

- Before you load the project, in the offline mode, under Target system/Target device, select whether both SIMOTION D435 as well as also SINAMICS_Integrated are selected. Please acknowledge any changes with **OK**.
- After starting the download, you will be prompted whether you wish to "Copy RAM to ROM" after you have successfully loaded the project. You must always answer with **Yes,** as otherwise your program must be reloaded again after power ON/OFF. This copying only refers to the SIMOTION part of the project.
- Please acknowledge with **OK** when the loading and copy operations have been completed.
- Acknowledge copy RAM to ROM in SINAMICS_Integrated with **Yes** and acknowledge with **OK** once this has been completed.
- Finally, switch SIMOTION D435 into the RUN operating state. To do this, click on the SIMOTION-CPU and using the righthand mouse key and target device/operating state, go to the operating state display.
- Here, click on the **RUN** button.

SIMOTION D435 is then in RUN and the demonstration case is ready for the presentation/demonstration!

5 Operator control of the application example

The application can be used to present SIMOTION D with SINAMICS and get to know and test the functions of the CPU D435.

Prerequisites

The following prerequisites must be fulfilled to use the application example:

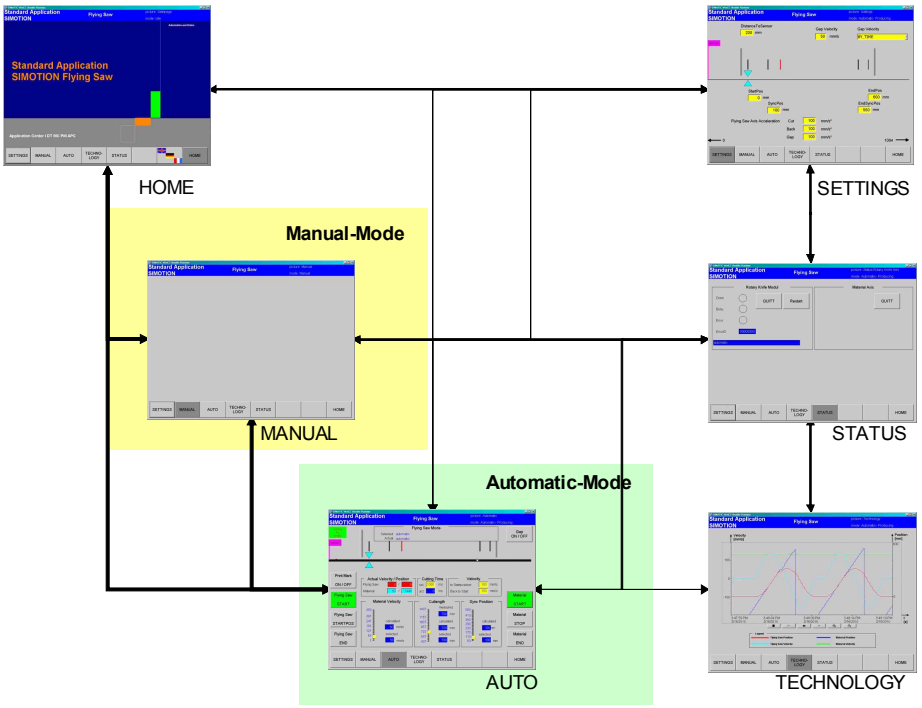
- The SIMOTION project is provided online in SIMOTION D435.
- The parameterization for the applications has been downloaded into SINAMICS (also included in the SIMOTION project!).
- All of the devices are powered-up.
- The SIMOTION D435 has been switched into the “Run” state using the online function of SIMOTION SCOUT.
- At least WinCC Flexible 2008 SP3 Upd2 is installed on the PC/PG.

Note WinCCflex requires the appropriate authorization. Authorization for at least 256 power tags is required.

5.1 Structure overview

Please refer to the following diagram for the basic operator control structure with all of the operator areas of the application.

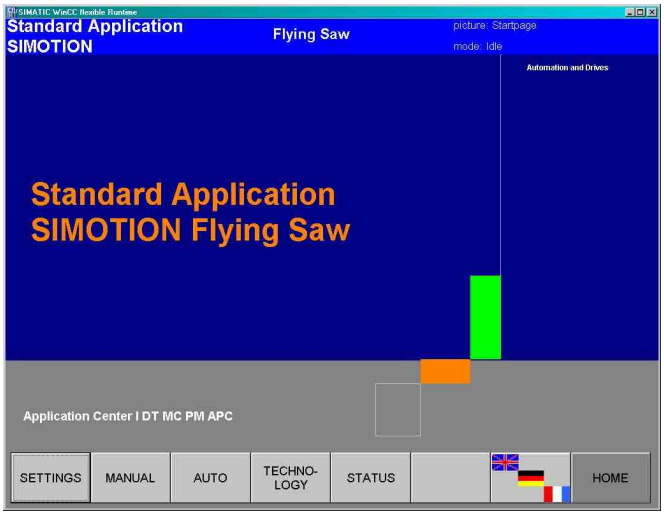
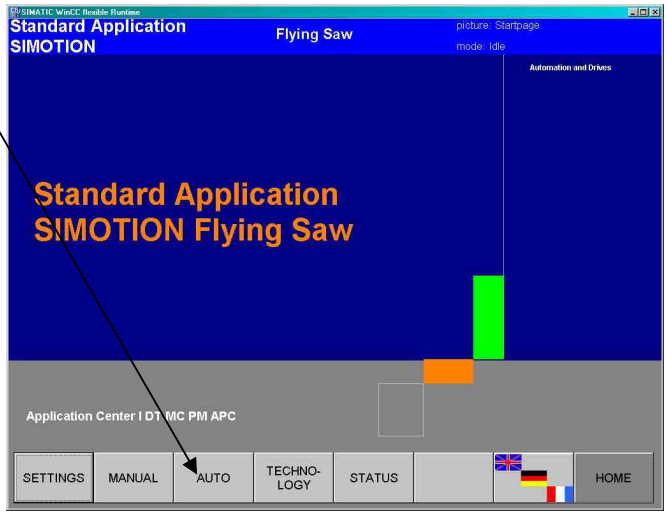
Fig. 5-1: Structure overview to demonstrate the application

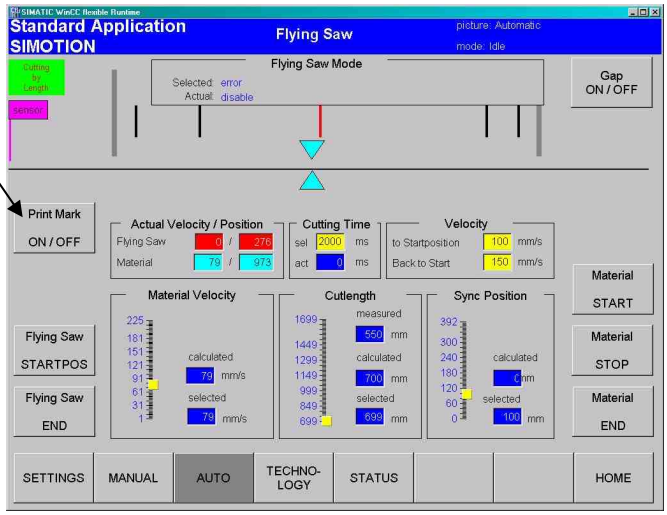
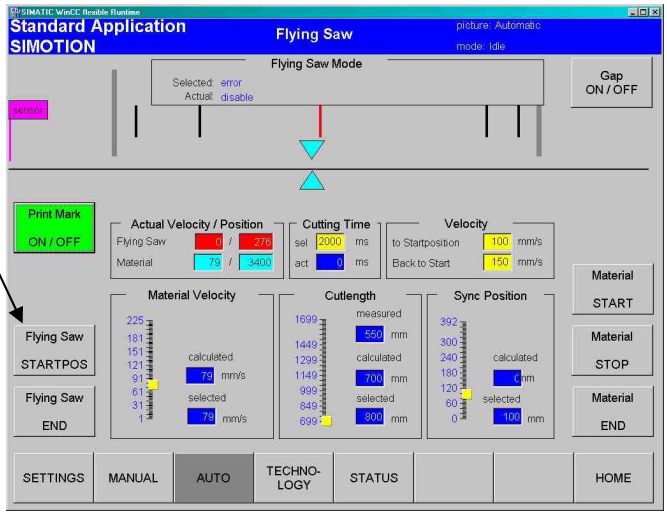
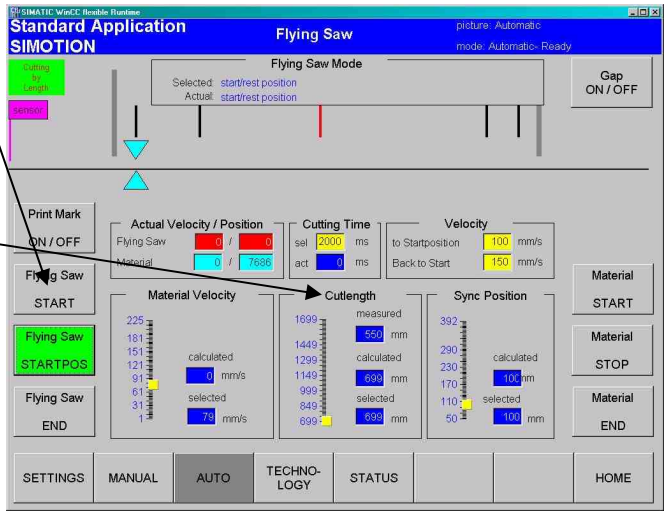


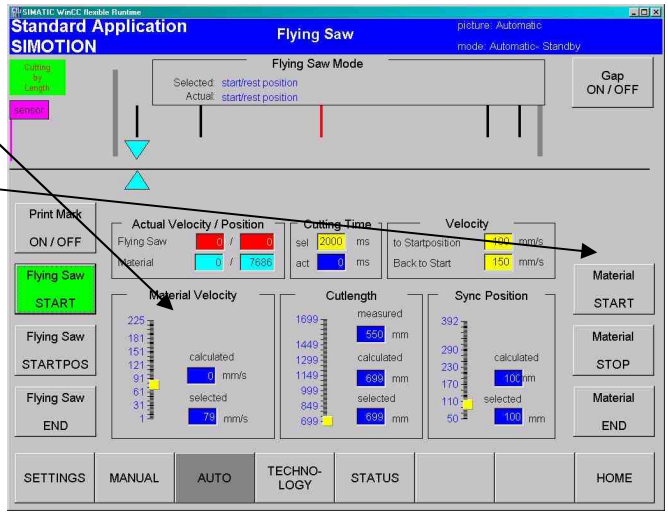
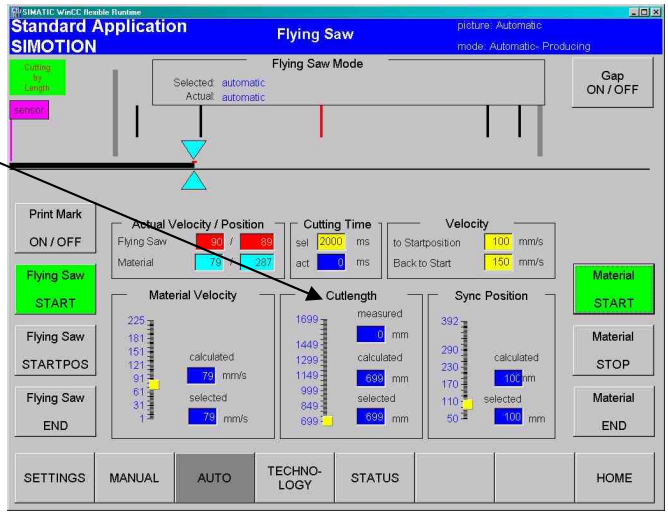
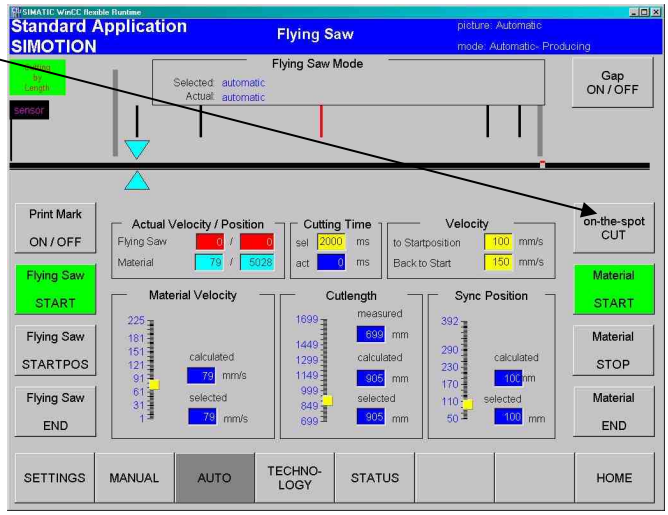
5.2 Brief instructions

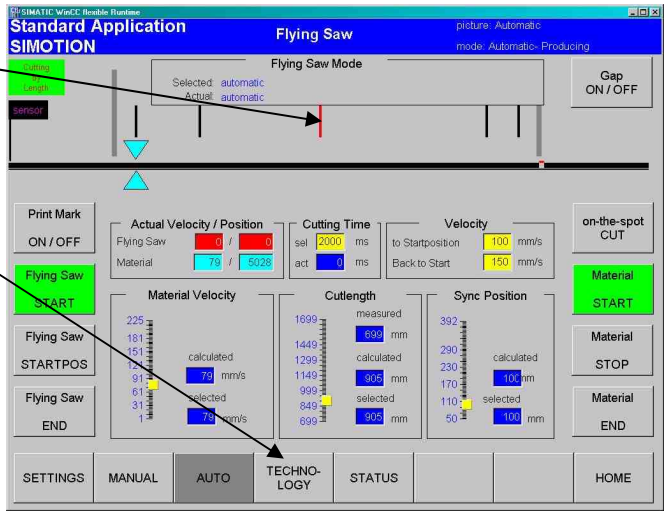
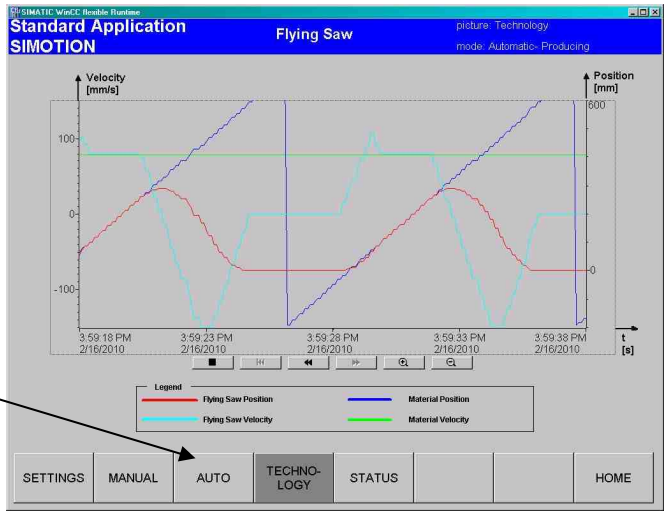
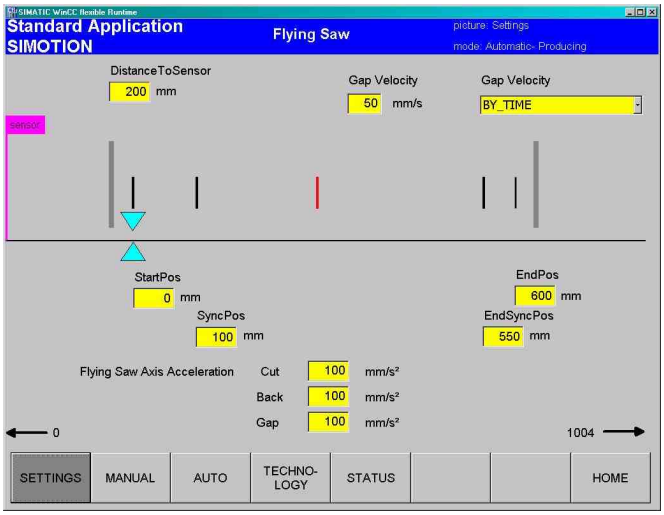
Execute the following steps in the sequence as listed in the following table to demonstrate the application example:

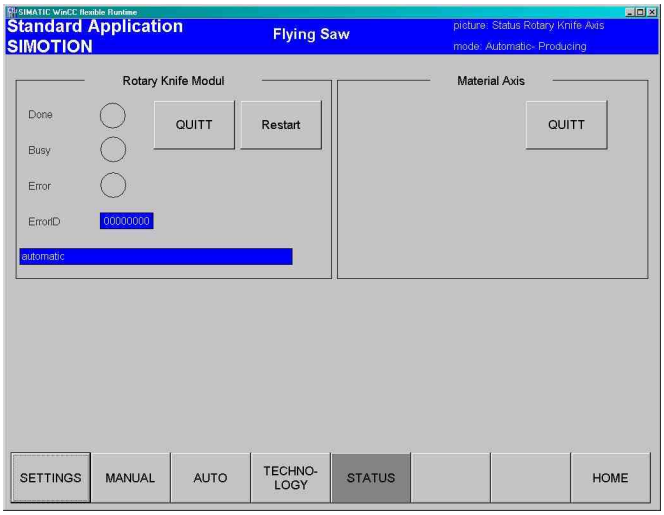
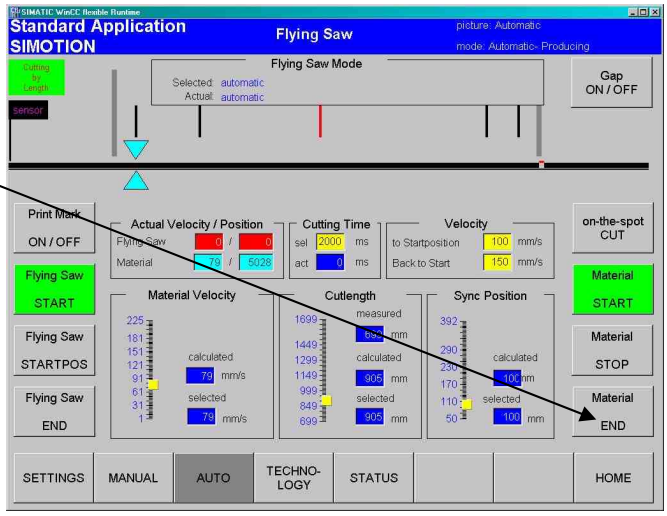
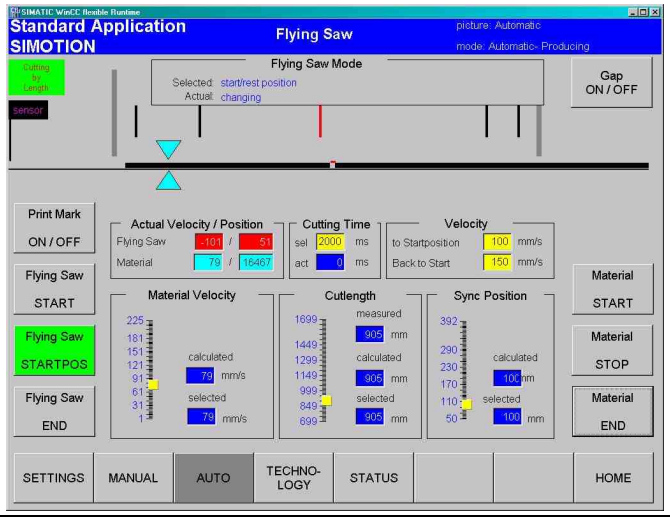
Table 5-1: Brief instructions to demonstrate/present the “flying saw” application example

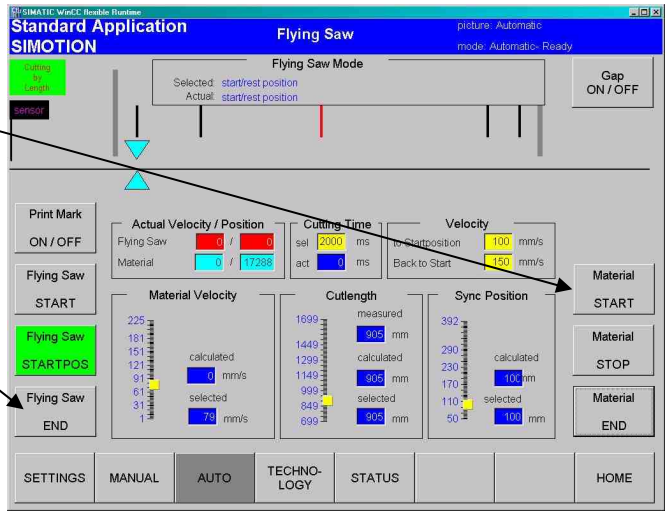
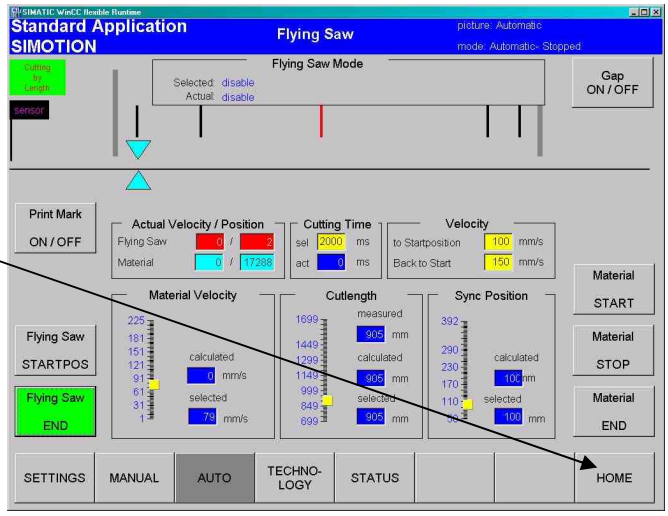
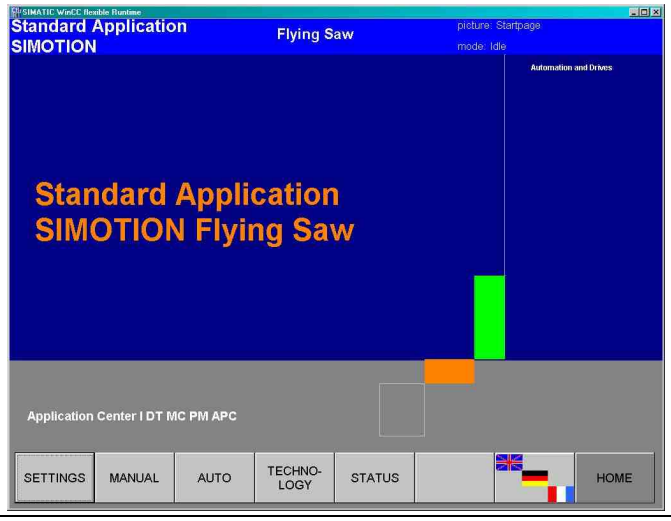
No.	Action	Comment
1	Call the following file "C:\Siemens\Step7\S7Proj\ FlyingSaw_V20\HmiEs\Project_1\ Project_1.WinCCStation.fwx".	
2	If the connection to the CPU was able to be established, the screen form appears at the top left - **mode: Idle.**	Check that your PG/PC is set to correct IP-Address. (**169.254.11.23** with "subnet mask“ **255.255.0.0**).
3	At the bottom left, please click on **AUTO** (3rd button from the left!).	

4	First activate the print mark simulation by pressing the **Print Mark Simulation ON / OFF** button. When cutting “to print mark”, you can only set an integer multiple of 200 mm as cut length (this is inherent to the system).	
5	Next, activate the travel to the starting position using the button **Flying Saw STARTPOS.**	
6	Once the flying saw has reached its starting position, the **Flying Saw START** button is displayed to activate the flying saw. Before you activate the flying saw, the required cut length should be selected using the **Cutlength** slider.	

7	Now you can start the material feed! Select the **Material Velocity** using the Material Velocity slider. You can start the material feed by pressing the **Material START** button.	
8	After the material web starts, the material enters the unit and the print mark simulation generates, after the length, set using the **Cutlength** slider switch, a print mark. The flying saw then makes a cut.	
9	You can immediately generate a print mark using the **on-the-spot CUT** button – which is also then cut by the flying saw.	

10	The red bar indicates how far the flying saw must travel with the material web with the current setting for one cut. Activate the technology screen by pressing the **TECHNOLOGY** button	
11	The following contents are displayed in the technology screen: • The actual material velocity • The actual velocity of the flying saw • The actual position of the flying saw • The position of the material referred to the actual cut You can return to the automatic screen by pressing the **AUTO** button	
12	Machine parameter settings Machine geometry: • Distance to Sensor • StartPosition • StartSyncPosition • EndSyncPosition • EndPosition Acceleration: • Cut • Back • Gap Create a “gap”: • GapVelocity	

13	The actual state of the function block of the application is displayed in the status display. You can read the ErrorID numbers, which are displayed under ErrorID, under **Chapter 8.3.7 Error messages**. Faults that are present can be acknowledged by pressing the **QUITT** button. If the function block is in the **error** state, then press the **Quitt** button until it goes into the **disable** state. Using the **Restart** button, you can reset the flying saw into the initial state after a restart.	
14	The material can be removed by pressing the **Material END** button	
15	After the last cut, the flying saw automatically moves to its starting position and waits for new material.	

16	Once the material has been completely removed from the screen area, then you can either allow new material to enter by pressing the **Material START** button and continue with **Step 8**, or end the “flying saw” by pressing the **Flying Saw END** button.	
17	After the flying saw has been ended, you can return to the welcome screen by pressing the **HOME** button.	
18		

6 Operatoration of the TestFB application

The pupose of this application is to learn the functionality of the FB-FlyingSaw as it should be installed in any user application and how to run the FB. This application also is prepared for the demo suitcase, and can be adapted also directly to the user hardware.

Prerequisites

The following prerequisites must be fulfilled to use the application example:

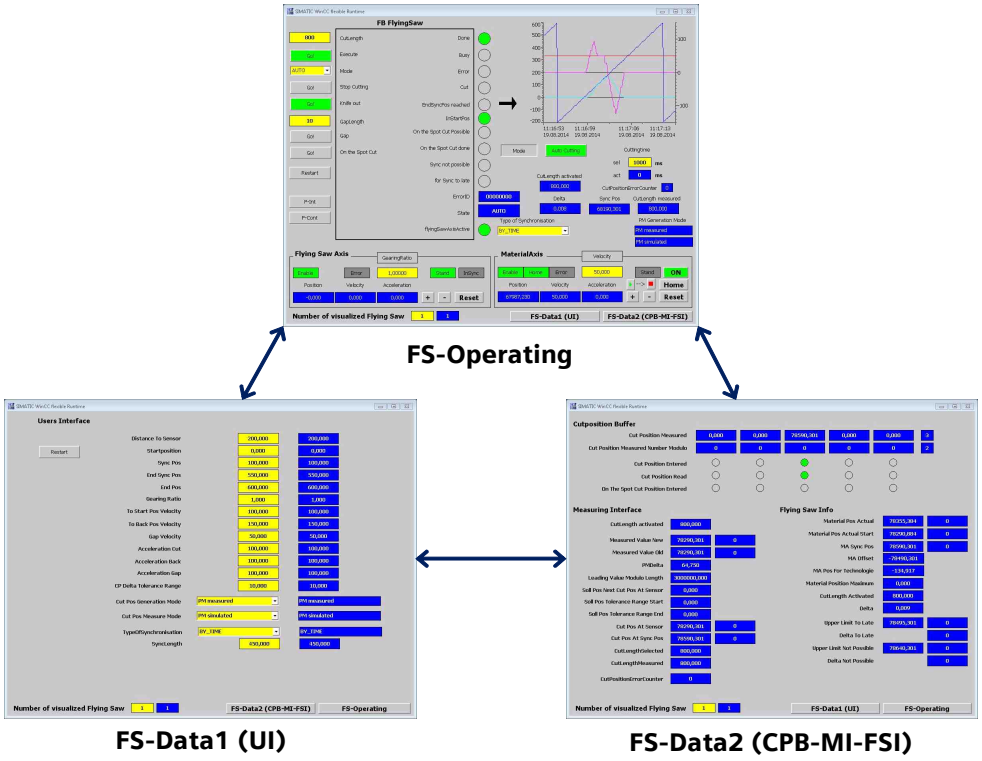
- The SIMOTION project is provided online in SIMOTION D435.
- The parameterization for the applications has been downloaded into SINAMICS (also included in the SIMOTION project!).
- All of the devices are powered-up.
- The SIMOTION D435 has been switched into the “Run” state using the online function of SIMOTION SCOUT.
- At least WinCC Flexible 2008 SP3 Upd2 is installed on the PC/PG.

Note WinCCflex requires the appropriate authorization. Authorization for at least 256 power tags is required.

6.1 Structure overview

Please refer to the following diagram for the basic operator control structure with all of the operator areas of the application.

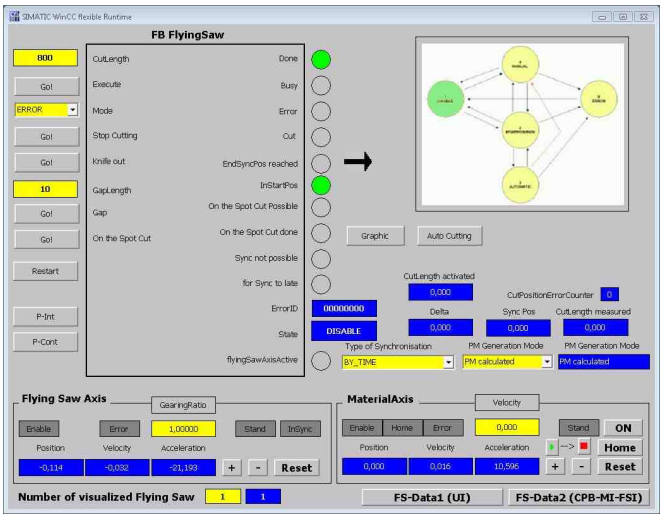
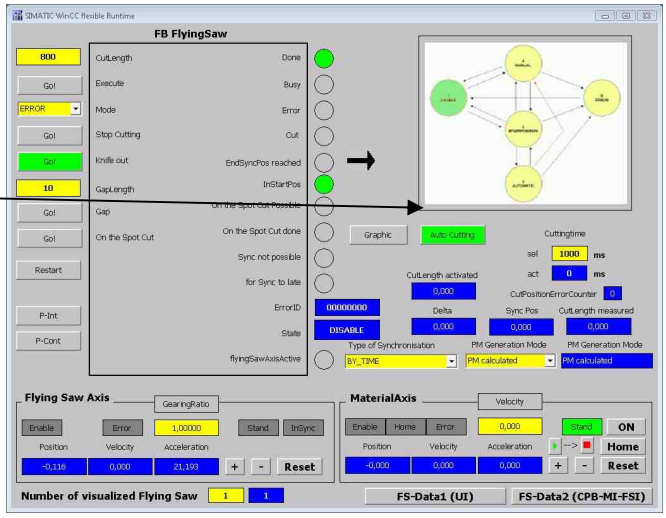
Fig 6-1: Structure overview to demonstrate the application

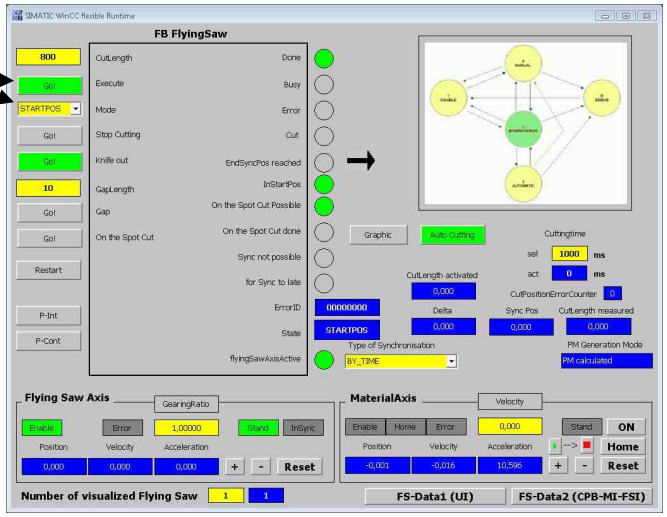
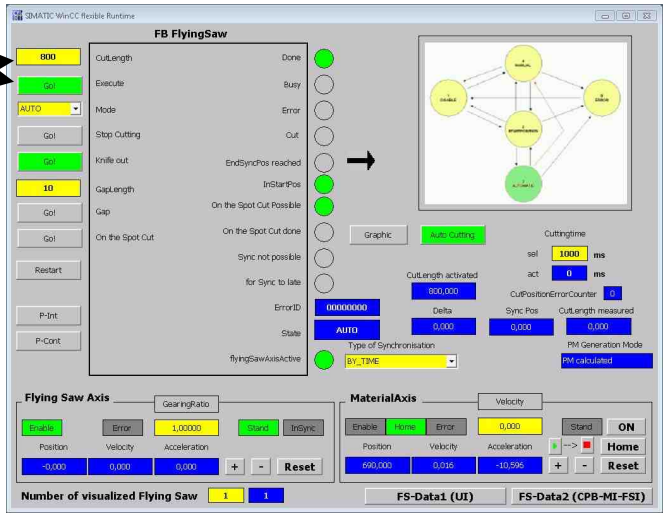
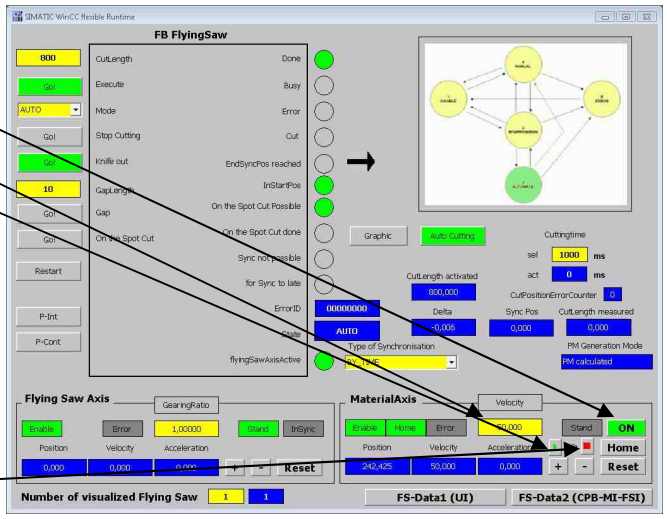


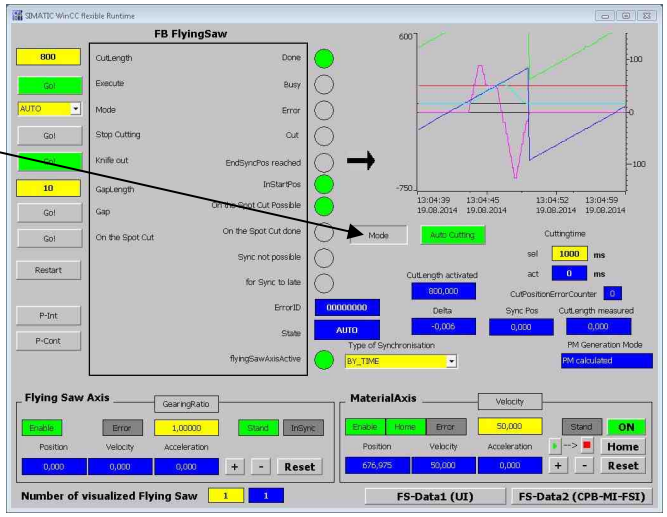
6.2 Brief instructions

Execute the following steps in the sequence as listed in the following table to demonstrate the application example:

Table 6-1: Brief instructions to demonstrate/present the TestFB-Application SIMOTION Flying Saw

Nr.	Aktion	Anmerkung
1	Start screen after activation of WinCC RT	
2	Activation of „Auto Cutting“ If „Auto Cutting“ is activated, all the control signals for the knife will be automatically generated and must not be activated t by the user. This simplifies the handling of the application. This function is activated via the button „Auto Cutting“.	

3	Run FlyingSaw to startposition Select mode „STARTPOS“ Activate button „Execute“ (Edge 0→1!)	
4	Switch FlyingSaw into mode „automatic“ Select mode „AUTO“ Activate button „Execute“ (Edge 0→1!)	
5	Activation of material movement Switch on material axis Select speed value Start movement (Stop movement)	

6	Switchover from the indication of the application state to the indication of the movement curves Switching over will be done with the button „Mode / Graphic“	
7	**Have fun with trying out the application !!!!!!!**	

Integration of the core functions

Content

The section “Intergration of core functions” gives an overview of all necessary steps for implementing the SIMOTION Flying Saw applications into the user’s program.

Preparatory activities and parameterizing works will be defined. Moreover this section also explains step by step how to integrate the application library into the user’s application.

7 Integration in the user program

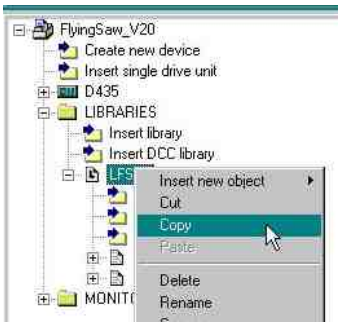
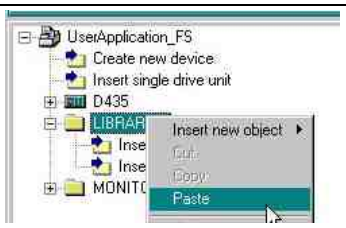
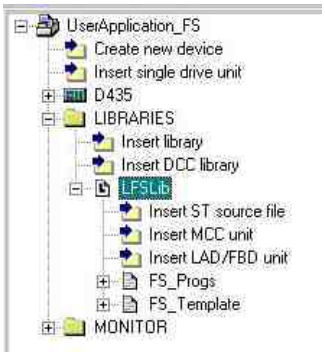
7.1 Importing the source code

The SIMOTION Flying Saw application is part of the *LFSLib* library. The library is available within the SIMOTION project “SIMOTION Flying Saw” as well as XML export. In order to be able to use the functionality of the library, this library must be integrated into the corresponding user project.

7.1.1 Copying from the application example

Two instances of SIMOTION SCOUT are opened in order to copy the *LFSLib* library. The application example is opened or dearchived in the first window and a new user project is created in the second window, which already includes the global library folder.

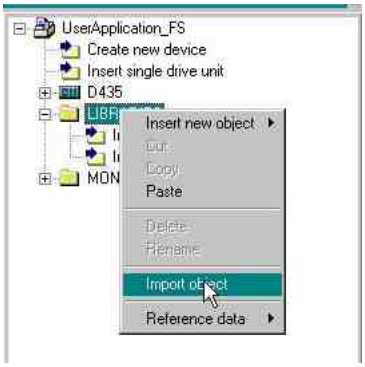
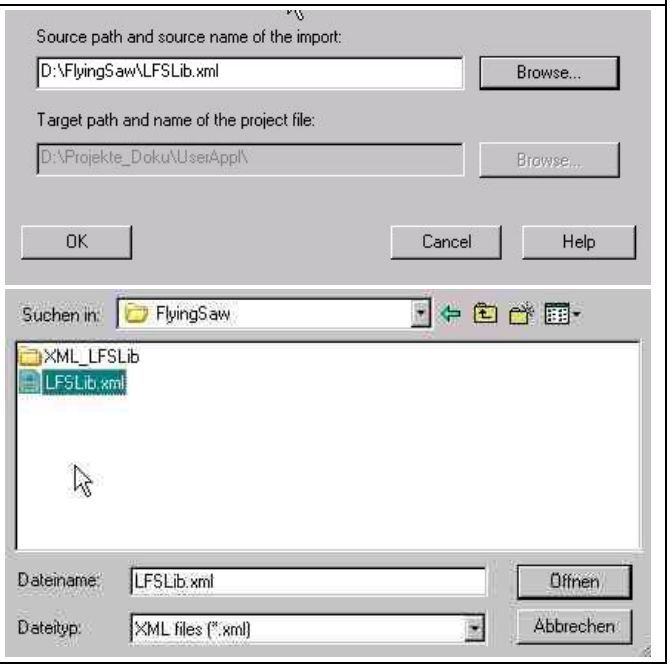
Table 7-1: Copying the library from the application example

No.		
1.	The library is downloaded to the buffer memory by right clicking on the library in the standard application following by **Copy**.	
2.	The library *LFSLib* can be inserted if, in the user project, the **Library** folder is selected and is then run by clicking on it with the righthand mouse key.	
3.	The lower-level objects are also copied.	

7.1.2 Intergration via XML import

Using XML import, the *LFSLib* library and the *FS_Progs* program unit will be linked to the existing user project.

Table 7-2: Importing the XML file

No.		
1.	For the XML import, the **LIBRARIES** folder must be selected in the user project and then executed by right-clicking. Using the **Import object** menu item, a window opens in which the path of the XML file must be specified.	
2.	Using the **Browse** button, the XML file path is specified and therefore the *LFSLib* library is imported into the user project.	

7.1.3 Linking the flying saw functionality

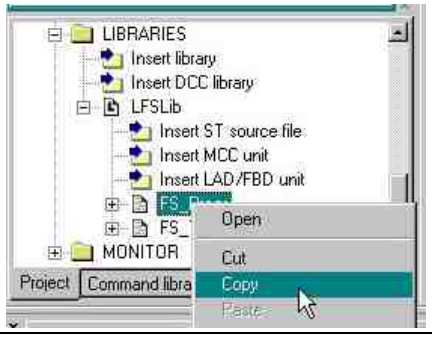
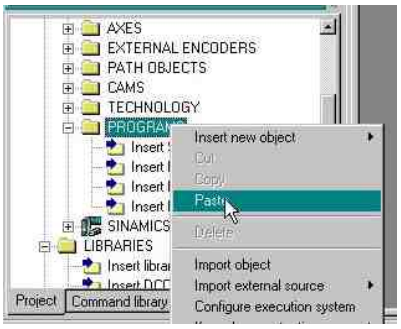
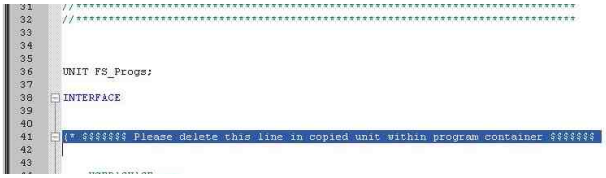
The *LFSLib* library comprises several program units, whereby the unit *FS_Template* contains the functionality of the application in the form of function blocks as well as data type definitions. The reference to the library is established in the unit *FS_Progs* as well as the instances of the function blocks and the data structures will be created. In addition, the unit includes the calls of the instances in those programs which must be integrated in the task system of the user project. The programs are not able to run in the library and may have to be adapted to the particular application (*startupFlyingSaw*).

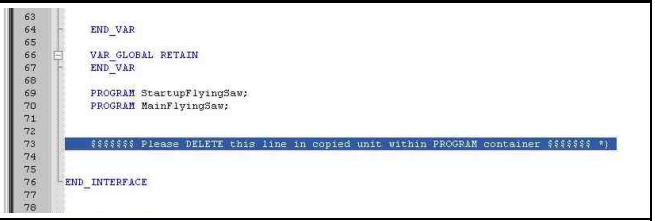
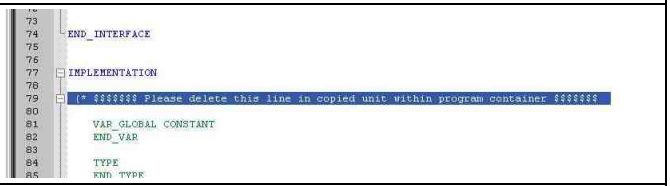
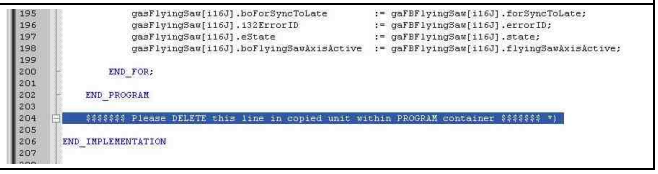
Table 7-3: Programs of the *RK_Progs* unit

Program	Description	Task
startupFlyingSaw	Initialization and parameterization	StartupTask
mainFlyingSaw	Main program	BackgroundTask

After the library has been copied / imported in the library folder, the program unit *FS_Progs* must be copied from the library folder into the program folder of the application – and modified at four positions:

Table 7-4: Copying and modifying the UNIT FS_Progs

No.		
1.	The unit is downloaded into the buffer memory by right-clicking on the program unit *FS_Progs* in the library *LFSLib* followed by ***Copy***.	
2.	Mark the **Program** folder in the user project. Insert the program unit *FS_Progs* by right-clicking **Paste**	
3.	Then, in the program unit *FS-Progs*, within the program container (not in the library!) four program lines must be deleted. Marked with **$$$$$$$**	

No.		
4.		
5.		
6.	Now the program unit can be compiled.	

When using the program unit *FS_Progs*, the user must not link the library by himself. The library is linked using the *FS_Progs* unit.

User program units requiring access to library-functions, library-types, library-function blocks and to data of the standard application, must be linked with the program unit *FS_Progs* in the interface section.

```
INTERFACE
        USES FS_Progs          // Linking the standard application
END_INTERFACE
```

Variable access

To access variables of the SIMOTION Flying Saw application, the following instances are set-up in the global data area in the *FS_Progs* program unit:

Table 7-5: Data interface

Element	Description
gasFlyingSaw	Block interfaces and parameters
gaFBFlyingSaw	Instance(s) of the *FBFlyingSaw*
NUMBER_OF_FLYING_SAWS	Number of FB instances

```
VAR_GLOBAL CONST
    NUMBER_OR_FLYING_SAWS          : INT    := 1;    // number of flying saws
END_VAR
VAR_GLOBAL
    gasFlyingSaw    : ARRAY[0.. NUMBER_OR_FLYING_SAWS -1] OF sFlyingSawType;
    gaFBFlyingSaw   : ARRAY[0.. NUMBER_OR_FLYING_SAWS -1] OF FBFlyingSaw;
END_VAR
```

7.2 Technology objects

7.2.1 Necessary technology objects and synchronism interconnections

To use the SIMOTION Flying Saw application, the following technology objects must be set-up and interconnected as described..
Please take into account the way the cutposition (CP) has to be detected and the way the actual application is used (to control a real machine or for test pruposes at a demo case):

Table 7-6: Technology objects required

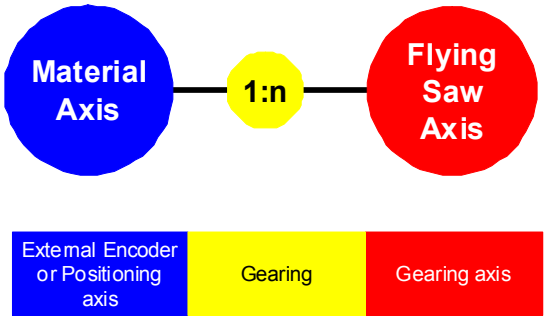
Object	Description		
Web Axis (MaterialAxis, (MaterialEncoder)	To determine the position and velocity of the material web as leading value of the SIMOTION Flying Saw application and to coordinate the motion of the material web axis from the same control.		
	TO	Real positioning or following axis	
		Machine encoder (external encoder)	
	Configuration	Configuration	When CP Generation Mode = CP_CALCULATED Set the axis type as **linear** without modulo length.
			When CP Generation Mode = CP_MEASURED For accuracy reasons, we recommend that the axis type is parameterized as **modulo linear axis** with a modulo length greater than three times the maximum product length.
			All of the other parameters should be set corresponding to the drive and encoder data.
		Mechanical system	For the parameter “distance per spindle revolution”, enter the distance as that value the material web travels within one revolution of the drive roll. Complete other parameters accordingly to the geometry and machine design.
		Limits	For this parameter, enter values that correspond to the system/drive specifications
		Referencing	Parameterize **“not required”**
		Monitoring, closed-loop control	Please set this parameter corresponding to the system specifications.
Flying Saw Axis (FlyingSawAxis)	Drive for the flying saw motion.		
	TO	Real following axis	
	Configuration	Configuration	If the application is used in a “Real Machine” sUsersInterface.boRealMachine = TRUE. Parameterize the axis as a synchronous axis. Set the axis type as **linear without modulo.**
			If the application is used in a test arrangement (e.g. demo case) sUsersInterface.boRealMachine = FALSE Parameterize the axis as a synchronous axis. Set the axis type as **linear with modulo** Using modulo length value sUsersInterface.r64EndPos + 500.0 Using startvalue value - (sUsersInterface.r64DistanceToSensor + 50.0) Thus the axis always is in the actual operating range and must not be referenced at test mode.

Object	Description		
			All of the other parameters should be set corresponding to the drive and encoder data.
		Mechanical system	For the parameter “distance per spindle revolution” enter that distance the “Flying Saw”(cutting equipment) is travelling along the web within one motor revolution. Complete other parameters corresponding to the geometry and the machine design.
		Pre-assignment	Please set this parameter corresponding to the drive dynamics
		Limits	Please set this parameter corresponding to the system specifications
		Referencing	Please set this parameter corresponding to the encoder. (The template assumes a referenced axis. If an incremental encoder is to be used, then the axis must be referenced in the **MANUAL** mode before additional operating modes are selected.)
		Monitoring functions, closed-loop control	Please set this parameter corresponding to the system specifications

7.2.2 Interconnection of technology objects

Synchronism interconnections of the SIMOTION Flying Saw application:

Figure 7-1: Synchronism interconnections required for the *FBFlying Saw*



The material axis (MaterialAxis) is the reference value for the flying saw. This reference value can be detected in two different ways:

1. The material path is controlled by the SIMOTION drive system which also controls the flying saw. In this case, a TO axis is provided. The synchronous connection with the axis of the flying saw (FlyingSawAxis) should have access to the setpoints of the material path which are existing in the system (setpoint coupling)
2. The speed of the material is measured with a machine encoder. In this case, there are no additional setpoints available. The synchronous connection with

the flying saw (FlyingSawAxis) can only access the actual values of the machine encoder (actual value coupling).

The FlyingSawAxis as “synchronous axis” is connected to the MaterialAxis in a 1:n (gearing) synchronous interconnections.

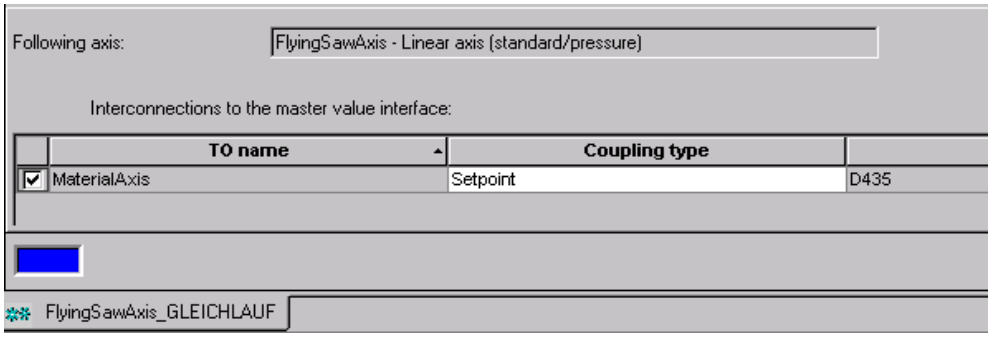
The technology objects and synchronous interconnections (shown in the diagram above) must be set-up and configured by the user in SIMOTION SCOUT.

Configuration of Synchronous Operation

Either

- Select the **TO Axis „MaterialAxis“** as master (leading) axis
- Select **setpoint** as coupling type.

Figure 7-2: Configuration of the synchronous interconnections between the master axis (material web) and flying saw as setpoint coupling



or

- Select the TO ExternalEncoder „MaterialEncoder“ as master (leading) axis
- Select actual value with extrapolation as coupling type.
  The time constant required for extrapolation is determined by using the calculation tool on the Utility&Applications CD.
  (Scout\CD14\SRC\4_TOOLS\ExtrapolationTimeCalculation).

Figure 7-3: Configuration of the synchronous interconnection between the master value (machining encoder) and the flying saw as actual value coupling

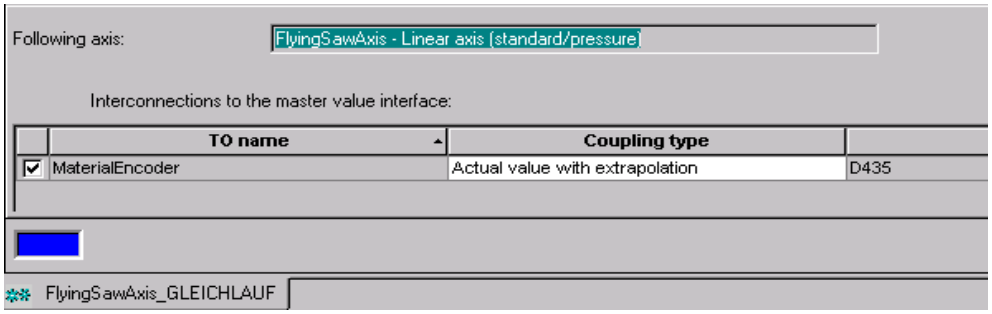


Table 7-7

Function / Variable	Value
Dynamic parameters	During synchonizing and desynchronizing the dynamic parameters of the time-related synchronization should be selected subject to the condition that the axis must not reach its dynamic limits.

7.3 Integrating the core functions of the application SIMOTION Flying Saw

The SIMOTION Flying Saw application comprises several functional units that are saved in the program unit *FS_Prog*.

7.3.1 Parameterizing in the startup task

It is useful to combine configuration data assignments – which won't be changed during operation - in one program (*startupFlyingSaw*). When starting the control in the *startupTask* these data need to be executed once and the values will be assigned to the variables.

- Assignment of the actual TO instances
- Assignment of machine-specific parameters
- Setting the restart bits.

The program is already prepared in the execution system as a program instance and is included in the UNIT *FS_Progs*.
The user only has to adapt the parameters to the real machine.

If several instances of the flying saw are used in the user program then the parameter assignments must be correspondingly supplemented.

7.3.2 Calling *FBFlyingSaw* in the user program

The function block of the flying saw core functions can be called after the integration in the user program.

The call is already prepared in the program unit *FS_Progs* in the *mainFlyingSaw* program. The user only has to integrate this program into one of the cycle tasks (preferably into the background task).

7.3.3 Global Variables

The *FBFlyingSaw* function block is influenced by several parameters and interfaces which are subdivided into the following areas:

- Block interfaces
- Block interface in the global data area

Changing tasks and modes are communicated to the function block via the block interface. The function block also communicates the actual status - and errors that have possibly occurred - back to the user program via this interface.

The parameters of the flying saw to be connected are communicated to the function block via the block interface in the global data area. In normal flying saw operation, these values will not change. Further, it includes data areas for communication with the user and for communication among the individual sub-functions in the overall.

Program Description

Content

The section “Program Description” is only then of interest if you wish to expand/adapt your application by using the technology templates..

Objective

This part of the documentation should

- Explain to readers the special features/issues when generating a program
- Describe in detail the program structure of the flying saw FB
- Illustrate and describe important program elements
- Provide information and instructions on how this template can be adapted.

Prerequisite

Before possibly expanding this template it is helpful to read the chapters in Section 1 to 3 and 7 in order to learn the basic functions and applications of the template.

Note

Before you make changes to the template, please observe the information and warnings listed in the following Chapter as well as the associated restrictions regarding support!

8 Program Description

8.1 Information and warnings

Before making changes

Before you start modifying components included in the core functions, please inform yourself about the component features in the ST/MMC documentation.

Uncontrolled, incorrect changes and modifications to core functions may result in death and severe bodily injury!

Restrictions regarding support when changing components of the core functions

The Application Center can only provide support for core functions that have not been changed

If changes have been made to the code, then support can no longer be provided for core functions.

This also applies for the revision and adaptation recommendations listed in this Chapter.

8.2 Type of Data

8.2.1 Overview

Enumeration types

Enumeration type declarations are provided for a part of the input and output parameters of the function blocks. Various modes and properties can be pre-set using these parameters.

Data structures

The function blocks of the SIMOTION Flying Saw application are in some cases parameterized via data structures that should be set-up for the particular block. Structures with the corresponding parameters are provided for each block.

8.2.2 Enumeration types

Name of the enumeration type	Contents
eOperationModesFlyingSawType	This enumerator type contains enumerators to select and display the operating state of the *FBFlyingSaw* function block. Declared in LFSLib.FS_Template:
eCutPositionGenerationModesType	Defines the method to determine the cutting position. Declared in: LFSLib.FS_Template
eCutPositionMeasureModesType	Defines the method to detect the measured values Declared in: LFSLib.FS_Template
eTypeOfSynchronizationType	Defines the synchronizing behavior of the knife slide on the motion of the material web. Declared in: LFSLib.FS_Template

eOperationModesFlyingSawType

This enumeration type contains enumerators to select and display the operating state of the *FBFlyingSaw* function block.

Element	Description
OMFS_ERROR	*FBFlyingSaw* is standing in state ERROR
OMFS_DISABLE	*FBFlyingSaw* is standing in state DISABLE
OMFS_START_POSITION	*FBFlyingSaw* is standing in state STARTPOSITION
OMFS_AUTOMATIC	*FBFlyingSaw* is standing in state AUTOMATIC
OMFS_MANUAL	*FBFlyingSaw* is standing in state MANUAL
OMFS_NOTHING_SELECTED	*FBFlyingSaw* has no change of state detected
OMFS_CHANGING	*FBFlyingSaw* is changing state

eCutPositionGenerationModesType

This enumerator type defines how the cutting position is determined:

Element	Description
CP_MEASURED	The cutting position is determined using sensor and print mark or is calculated and stored into the CutPositionBuffer.
CP_CALCULATED	The cutting position keeps constant, the material position is reduced by cutlength after each cut. This method doesn't need any stored position data.

eCutPositionMeasureModesType

This enumerator type defines how the measured value is detected:

Element	Description
CPM_DETECTED	The cutting position is detected using a sensor and a print mark.
CPM_SIMULATED	The cutting position is calculated from the given cut length.

eTypeOfSynchronizationType

This enumerator type defines the synchronizing characteristics of the cutting equipment to the material web motion:

Element	Description
BY_TIME	Synchronization by dynamic values
BY_LEADING_AXIS_VALUE	Synchronization through distance. The distance used will be stored in the variable “usersInterface.r64SyncLength”.
SYMMETRIC	The distance is dimensioned so that symmetrical synchronization is achieved without any overshoot.

8.2.3 Data Structures

Name of Structure	Content
sFlyingSawType	This structure contains the complete global data of the SIMOTION Flying Saw application. Declared in: LFSLib.FS_Template
sTOsType	This structure contains the definition of all technology objects used (axes, machine encoders and measuring probes). Declared in: LFSLib.FS_Template
sUsersInterfaceType	This structure contains the interface for the user to enter geometry and motion data of the shears mechanical system, as well as to enter operating versions. Declared in: LFSLib.FS_Template
sCutPositionBufferType	This structure contains the memory in which the cutting positions are saved with associated position deviations. Declared in: LFSLib.FS_Template
sCutPositionBufferManagement Type	This structure contains the internal data area to precisely enter the cutting position – and cutting position deviation for print mark correction. Declared in: LFSLib.FS_Template
sFlyingSawInfoType	This structure contains data about the actual leading value, the actual position of the leading value as well as its position from the perspective of the flying saw. Declared in: LFSLib.FS_Template
sMeasuringValueType	This structure contains the internal data area to determine the cutting position. Declared in: LFSLib.FS_Template

Generally applicable for labeling structural elements:

[IN]: Values that should be provided by the user

[OUT]: Results or feedback signals

[IO]: Values, depending on the block connection, must be supplied by the user or will be supplied by the function

sFlyingSawType

This structure contains the complete global data of the application SIMOTION Flying Saw:

I/O	Element	Type of Data	Description
IN	sAxisTOs	sTOsType	Object references of all technology objects (axes, external encoders and measuring probes).
IN	sUsersInterface	sUsersInterface Type	Interface for the user to enter geometry and motion data of the shears mechanical system – as well as to enter sequence options such as a restart in ongoing operation.
I/O	sCutPositionBuffer Management	sCutPosition BufferManagement Type	Internal data area to enter the cut position as a precise position and cut position deviation for the print mark correction.
I/O	sFlyingSawInfo	sFlyingSawInfo Type	Data area to display the actual leading value interconnection as well as the actual leading value position.
I/O	sMeasuringValue	sMeasuringValue Type	Internal data area to detect the cutting position.
IN	r64FormatLength	LREAL	[mm] *r64formatLength* is the length of the cut material element.
IN	boExecute	BOOL	Activates the mode changeover with the operating mode specified in *eMode*.
IN	eMode	eOperationModes FlyingSawType	Using *eMode* the operating mode is preselected that is then activated with the next positive edge at input *boExecute*. OMFS_ERROR: Preselects the ERROR mode OMFS_DISABLE: Preselects the DISABLE mode OMFS_START_POSITION: Preselects the STARTPOSITION mode OMFS_AUTOMATIC: Preselects the AUTOMATIC mode OMFS_MANUAL: Preselects the MANUAL mode
IN	boStopCutting	BOOL	Feedback signal from the cutting device: TRUE: Cutting has been completed
IN	boKnifeOut	BOOL	Feedback signal from the cutting device: TRUE: Cutting device withdrawn from the material web
IN	boGap	BOOL	TRUE: Create gap has been selected
IN	boOnTheSpotCut	BOOL	TRUE: Start “Immediate cut“
OUT	boDone	BOOL	TRUE: Selected mode is activated..
OUT	boBusy	BOOL	TRUE: Mode changeover is active.
OUT	boError	BOOL	Indicates errors in the *FBFlyingSaw* and at the FlyingSawAxis.
OUT	boCut	BOOL	Signal to the cutting device TRUE: Flying saw is in synchronism to the material web, the cutting device can cut
OUT	boFlyingSawInStartPos	BOOL	TRUE: Flying saw is in initial position (Startposition)
OUT	boEndSyncPosReached	BOOL	TRUE: The end of the synchronous range has been reached or exceeded
OUT	boOnTheSpotCutPossible	BOOL	TRUE: An “Immediate cut” is possible
OUT	boOnTheSpotCutDone	BOOL	TRUE: An “Immediate cut” was executed
OUT	boSyncNotPossible	BOOL	TRUE: The last synchronization operation was not successful

I/O	Element	Type of Data	Description
OUT	boForSyncToLate	BOOL	TRUE: The flying saw has not reached the start position in time for the next cut. The saw will cut automatical the next reachable integer multiple of cutlength.
OUT	b32ErrorID	DWORD	Error code
OUT	eState	eOperationModes FlyingSawType	Actual mode OMFS_ERROR: ERROR mode OMFS_DISABLE: DISABLE mode OMFS_START_POSITIONC: STARTPOSITION mode OMFS_AUTOMATIC: AUTOMATIC mode OMFS_MANUAL: MANUAL mode OMFS_NOTHING_SELECTED: No mode change OMFS_CHANGING: Mode change active
OUT	boFlyingSawAxisActive	BOOL	TRUE: flying saw axis enabled

sTOsType

This structure contains the definition of all of the axes, external encoders and measuring probes used (the user must parameterize these):

I/O	Element	Type of Data	Description
IN	toLeadingValue	_AXIS_REF	Reference to the technology object of the leading value. Possible technology objects: - posAxis (real or virtual) - externalEncoderType
IN	toFlyingSawAxis	followingAxis	Refer to the technology object of the Flying Saw axis. Only one real axis is permissible
IN	toPrintmark	measuringInput Type	Reference to the technology object of the measuring probe used for print mark detection. The measuring probe must be assigned to the leading value.

sUsersInterfaceType

This structure includes the interface for the user to enter physical boundary conditions, operating mode versions as well as to enter a restart during operation (this function must be parameterized by the user):

I/O	Element	Type of Data	Description
IN	r64DistanceToSensor	LREAL	[mm] Distance between the sensor for the material and print mark detection and the axis zero point of the distance-reference system of the flying saw (distance to the sensor).
IN	r64StartPos	LREAL	[mm] Start position within the distance reference system of the flying saw.
IN	r64SyncPos	LREAL	[mm] Position within the distance-reference system of the flying saw where synchronization is completed (synchronous position).
IN	r64EndSyncPos	LREAL	[mm] Maximum position within the distance-reference system of the flying saw where synchronous motion must have been completed (end of the synchronous range)
IN	r64EndPos	LREAL	[mm] Reversal point in the distance-reference position of the flying saw.
IN	r64ToStartPosVel	LREAL	[mm/s] Velocity when positioning to the start position.

I/O	Element	Type of Data	Description
IN	r64ToBackPosVel	LREAL	[mm/s] Velocity when positioning back to the start position after a cut.
IN	r64GapVelocity	LREAL	[mm/s] Velocity after executing the higher-level gap positioning (“creating a gap!).
IN	r64CutAcceleration	LREAL	[mm/s²] Acceleration when synchronizing for the cut (only if eTypeOfSynchronisation = BY_TIME!) .
IN	r64BackAcceleration	LREAL	[mm/s²] Acceleration when positioning back to the start position after a cut.
IN	r64GapAcceleration	LREAL	[mm/s] Acceleration when executing the higher-level gap positioning (“creating a gap!).
IN	r64GearingRatio	LREAL	Ratio between the measured and real material motion.
IN	r64CPDeltaTolerance Range	LREAL	[mm] Half the tolerance bandwidth for the activation range of the cut position detection
I/O	boRestart	BOOL	Flag restart, the user must set this, it is reset by the application
I/O	boInterruptProduction	BOOL	interrupt production of FlyingSaw without leaving automatic mode (e.g. to activate STO!)
I/O	boContinueProduction	BOOL	continue production of FlyingSaw after interrupt production in automatic mode (e.g. after STO!)
IN	eCutPosition GenerationMode	eCutPosition GenerationMode Type	Method to determine the cut position: CP_CALCULATED: Cut position is calculated by cutting to length of material position CP_MEASUERED: Cut position is detected using the measuring probe
IN	eCutPosition MeasureMode	eCutPosition MeasureMode Type	Method to measure cut position, only valid when CP_MEASURED active: CPM_DETECTED: Cut position is detected with sensor and print mark and then entered in CutPositionBuffer. CPM_SIMULATED: Cut position is calculated by cutlength and entered in CutPositionBuffer
IN	eTypeOf Synchronisation	eTypeOf Synchronisation Type	Synchronizing method to synchronize to the material axis: BY_TIME: Referred to time BY_LEADING_VALUE: Referred to distance SYMMETRIC: Referred to distance, without overshoot in velocity
IN	r64SyncLength	LREAL	[mm] length of the synchronization-distance at distance-related synchronization
IN	boRealMachine	BOOL	TRUE: drives connected with real machine FALSE: drives connected with test arrangement (e.g. demo case)

sCutPositionBufferType

This structure contains the cut position memory (this is not relevant for the user):

I/O	Element	Type of Data	Description
I/O	ar64CutPosition Measured	ARRAY [0..CPB_MAX-1] OF LREAL	[mm] Buffer of the measured cut position
I/O	ar64CutPosition MeasuredNumber Modulo	ARRAY [0..CPB_MAX-1]	[-] Buffer of the measured cut position number of modulo cycles

I/O	Element	Type of Data	Description
		OF DINT	
I/O	aboCutPositionRead	ARRAY [0..CPB_MAX-1] OF BOOL	Buffer of the flag “cut position read”
I/O	aboCutPositionEntered	ARRAY [0..CPB_MAX-1] OF BOOL	Puffer of the flag “cut position entered”
I/O	aboOnTheSpotCut PositionEntered	ARRAY [0..CPB_MAX-1] OF BOOL	Buffer of the flag “OnTheSpotCut Position entered”

sCutPositionBufferManagementType

This structure contains the internal data area to enter the cutting position and to control the cutting position correction as precise position (this is not relevant for the user):

I/O	Element	Type of Data	Description
I/O	sCutPositionBuffer	sCutPosition BufferType	data area of cut position buffer
I/O	i16BufferIn	INT	Pointer to next input position of buffer
I/O	i16BufferOut	INT	Pointer to next output position of buffer
I/O	i16CutCorrectionWait	INT	Number of cuts without cut position correction
I/O	boBufferOutTrigger	BOOL	Start trigger to delete the read buffer entry
I/O	boBufferReadTrigger	BOOL	Start trigger to read the next buffer entry
I/O	boBufferResetActive	BOOL	Reset of buffer active
I/O	boBufferInActive	BOOL	Input of value in buffer active
I/O	boBufferOutActive	BOOL	Deletion of buffer value active
I/O	boBufferReadActive	BOOL	Reading of buffer value active

sFlyingSawInfoType

This structure contains calculation results and display information – relevant for the user – that provides information about the state of the flying saw.

I/O	Element	Type of Data	Description
OUT	boAxisAsLeadingValue	BOOL	TRUE Leading value is a real or virtual posAxis FALSE No posAxis can be entered as leading value
OUT	toLeadingValueAxis	posAxis	If the leading value is a real or virtual posAxis, then the name of the TO is the axis - otherwise TO#NIL
OUT	boExternalEncoderAs LeadingValue	BOOL	TRUE Leading value is an external encoder FALSE No external encoder entered as leading value
OUT	toLeadingValue ExternalEncoder	externalEncoder Type	If the leading value is an external encoder, then the name of the TO is the external encoder, otherwise TO#NIL
OUT	boTOLeadingValue Active	BOOL	TRUE TO master value active
OUT	boLeadingValue Standstill	BOOL	TRUE ,Master value in standstill
OUT	toActiveFollowing	followingObject	Active following object of the FlyingSaw axis

I/O	Element	Type of Data	Description
	Object	Type	
OUT	r64MaterialPosActual Start	LREAL	[mm] Actual material position at start of FlyingSaw activity
OUT	i32MaterialPosActual StartNumberModulo	DINT	[-] Number of modulo cycles at actual material position at start of FlyingSaw activity
OUT	r64MaterialPosActual	LREAL	[mm] Actual material position
OUT	i32MaterialPosActual NumberModulo	DINT	[-] Number of modulo cycles at actual material position
OUT	boOnTheSpotCut Selected	BOOL	TRUE: On the spot cut selected
OUT	r64MASyncPos	LREAL	[mm] Actual synchronous position referred to the material axis
OUT	i32MASyncPos NumberModulo	DINT	[-] Number of modulo cycles at actual synchronous position referred to the material axis
OUT	r64MASyncOffset	LREAL	[mm] Actual shift between the reference systems – material-axis and axis of the flying saw
OUT	r64MAPosFor Technologie	LREAL	[mm] Actual position of the material axis in the reference system of the flying saw axis
OUT	r64CutLength Activated	LREAL	[mm] aktivated cutlength in mode „CP_CALCULATED“
OUT	r64MaterialPosition Maximum	LREAL	[mm] maximum material position before „Redefine Position“
OUT	r64Delta	LREAL	[mm] Knife positions error (caused by following error) at start cut (accuracy +/- $V_{mat\ akt}$ * Backgroundcycle)
OUT	boInterruptProduction Activated	BOOL	TRUE: interrupt active
OUT	boContinueProduction Activated	BOOL	TRUE: continue active
OUT	boProduction Interrupted	BOOL	TRUE: FlyingSaw is deactivated in automatic mode and ready for STO.
OUT	boProduction Continued	BOOL	TRUE: FlyingSaw is reactivated and ready for start operation again.
OUT	r64UpperLimitToLate	LREAL	[mm] Maximum material position to start FlyingSaw
OUT	i32UpperLimitToLate NumberModulo	DINT	[-] Number of modulo cycles at maximum material position to start FlyingSaw
OUT	r64UpperLimitNot Possible	LREAL	[mm] Material position for abort synchronization
OUT	i32UpperLimitNot PossibleNumber Modulo	DINT	[-] A Number of modulo cycles at material position to abort synchronization
IN	eCutPosition GenerationMode	eCutPosition GenerationMode Type	Method to determine the cut position: CP_CALCULATED: Cut position is calculated by cutting to length of material position CP_MEASUERED: Cut position is detected using the measuring probe
IN	eCutPosition MeasureMode	eCutPosition MeasureMode Type	Method to measure cut position, only valid when CP_MEASURED activ: CPM_DETECTED: Cut position is detected using the sensor and the print mark and entered in CutPositionBuffer. CPM_SIMULATED: Cut position is culculated by cutlength and entered in CutPositionBuffer

sMeasuringValueType

This structure contains the internal data area of the measured value detection (this is not relevant for the user):

I/O	Element	Type of Data	Description
I/O	boStartMeasuring	BOOL	Flag “Start measured value generation”
OUT	r64CutlengthActivated	LREAL	[mm] Actual cut length
OUT	r64MeasuredValueNew	LREAL	[mm] Measured value, actual
OUT	i32MeasuredValueNew NumberModulo	DINT	[-] Number of modulo cycles at actual measured value
OUT	r64MeasuredValueOld	LREAL	[mm] Measured value, old
OUT	i32MeasuredValueOld NumberModulo	DINT	[-] Number of modulo cycles at old measured value
OUT	r64MVDelta	LREAL	[mm] Difference between the new and old measured value
IN	eMeasuringEdge	Enum Measured Edge	Selects the measurement-initiating edge at the sensor: RISING_EDGE: Rising edge at the sensor FALLING_EDGE: Falling edge at the sensor
I/O	boWaitForFirstCut Position	BOOL	Flag “Wait for the first measured value”
I/O	boWaitForCyclCut Position	BOOL	Flag “Wait for the next measured value”
I/O	boCutPosition MeasuringActive	BOOL	Flag “Wait for measured value”
I/O	boStartPosAtSensor Detected	BOOL	Flag “Start position measured”
IN	boApplicationIn Automatic	BOOL	Flag “Application in the AUTOMATIC mode”
OUT	boNewCutPosition Entered	BOOL	Flag “New value entered in the CutPositionBuffer”
OUT	r64LeadingValueModulo Length	LREAL	[mm] Modulo length of the leading value
OUT	r64SetPosNextCut PosAtSensor	LREAL	[mm] Interpolated next cutting position at the sensor
OUT	i32SetPosNextCutPosAt SensorNumberModulo	DINT	[-] Number modulo cycles at interpolated next cutting position at the sensor
OUT	r64SetPosTolerance RangeStart	LREAL	[mm] Tolerance bandwidth of the next cutting position, start
OUT	r64SetPosTolerance RangeEnd	LREAL	[mm] Tolerance bandwidth of the next cutting position, end
IN	r64DeltaTolerance Range	LREAL	[mm] Half the tolerance bandwidth
OUT	i16CutPositionError Counter	INT	Cut position fault counter
OUT	r64CutLengthMeasured	LREAL	[mm] Actually measured cut length
OUT	r64CutPosAtSensor	LREAL	[mm] Actual measured material position if the cutting position is at the measuring point
OUT	r64CutPosAtSensor NumberModulo	DINT	[-] Number modulo cycles at actual measured material position if the cutting position is at the measuring point

I/O	Element	Type of Data	Description
OUT	r64CutPosAtSyncPos	LREAL	[mm] Interpolated material position if the cutting position coincides with the synchronous point.
OUT	r64CutPosAtSyncPos NumberModulo	DINT	[-] Number modulo cycles at interpolated material position if the cutting position coincides with the synchronous point.

8.3 FBFlyingSaw

Control block of the SIMOTION Flying Saw application.

8.3.1 Block name

FBFlyingSaw

8.3.2 Task

Control block of the SIMOTION Flying Saw application. The block includes the following functions:

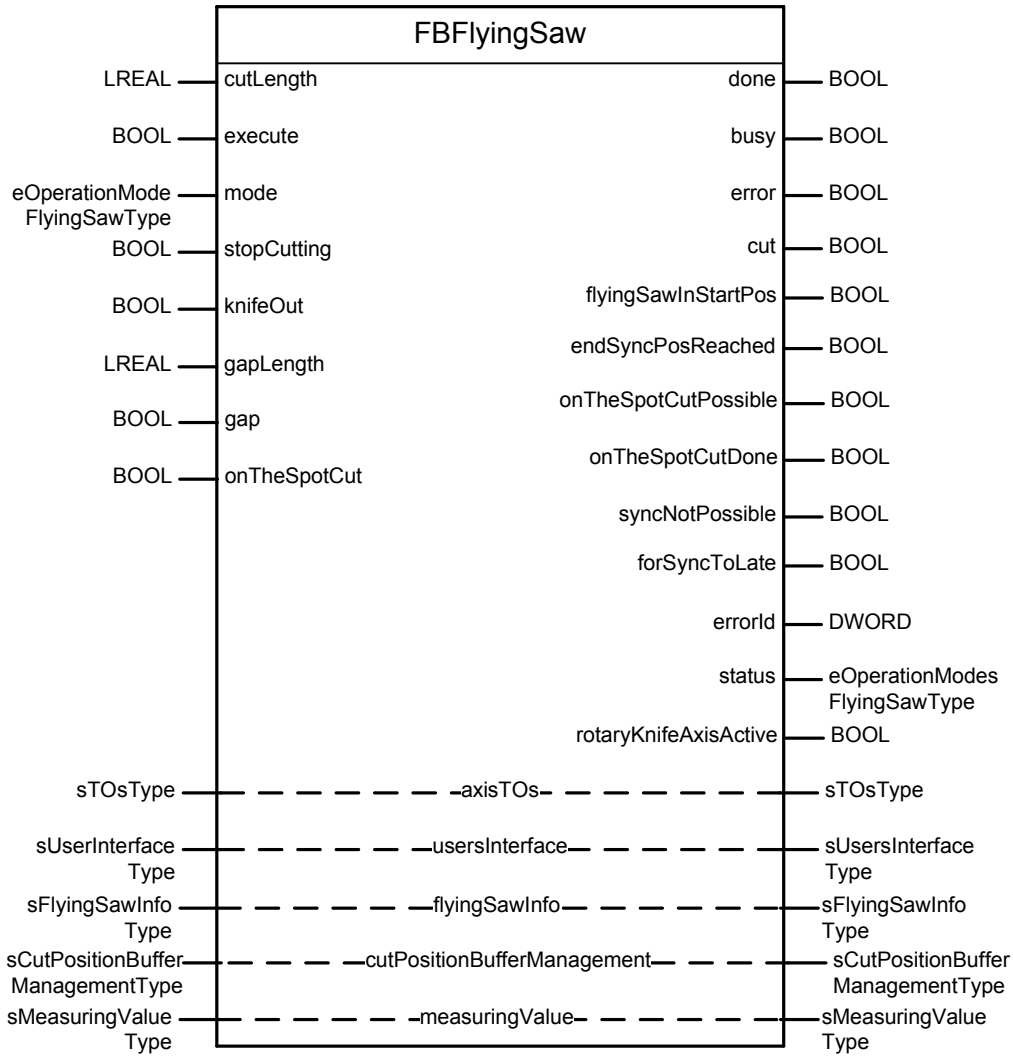
- Mode manager of the application, including the following modes ERROR, DISABLE, MANUAL, STARTPOSITION, AUTOMATIC
- Determining the cutting position either using a print mark and measuring sensor or a parameterizable cut length
- Synchronizing to the material web
- Controlling the cutting device during synchronous motion
- Possibility of “creating a gap”

Referencing the axes and setting-up operation (jog) are not included in the application.

8.3.3 Integration in the run-time system

The function block can be called in each cyclic task. The user must link the block.

8.3.4 Graphic representation of the block



8.3.5 Block parameters

Input parameters

Name	Data type	Initial value	Description
cutLength	LREAL	800.0	Cut material element length in mm
execute	BOOL	FALSE	A pending mode changeover is activated via this input.
mode	eOperation ModesFlying SawType	OMFS_ ERROR	With *mode*, the mode is pre-selected, which is activated with the next positive signal edge at the *execute* input. OMFS_ERROR: Pre-selection, ERROR mode OMFS_DISABLE: Pre-selection, DISABLE mode OMFS_AUTOMATIC: Pre-selection, STARTPOSITION mode OMFS_AUTOMATIC: Pre-selection, AUTOMATIC mode OMFS_MANUAL: Pre-selection, MANUAL mode
stopCutting	BOOL	FALSE	By setting this signal, the block is signaled that the cutting operation has been completed and a gap can be created - or the cutting equipment (knife) can be withdrawn from the material.
knifeOut	BOOL	FALSE	By setting this bit, the block is signaled that the cutting equipment is no longer connected to the material and the cutting device (knife) can be withdrawn to the start position
gapLength	LREAL	10.0	[mm] Length of the gap to be created
Gap	BOOL	FALSE	TRUE: A gap with length *gapLength* is created FALSE: No gap is created
onTheSpotCut	BOOL	FALSE	By setting this signal, a synchronous position is immediately generated on the material web at the “sensor position” for the next cutting operation.

Input / output parameters

Name	Data type	Initial value	Description
axisTOs	sTOsType		Defines the axes, machine encoder, and measuring probe being used
usersInterface	sUsersInter faceType		Interface to enter dynamic and geometrical parameters
flyingSawInfo	sFlyingSawInfo Type		Internal data area for display
cutPositionBuffer Management	sCutPosition Buffer Management Type		Internal data area to enter precise data for the cutting position and cutting position deviation for the cut position determination.
measuringValue	sMeasuring ValueType		Internal data area to determine the cutting position.

Output parameters

Name	Data type	Initial value	Description
done	BOOL	FALSE	Selected mode is activated.

Name	Data type	Initial value	Description
busy	BOOL	FALSE	Mode changeover is active.
Error	BOOL	FALSE	FALSE: No error TRUE: There is an error, refer to error ID Processing and calculating the process values is not interrupted in the case of an error.
cut	BOOL	FALSE	FALSE: The flying saw is stationary, is synchronizing or is moving back to the start position. TRUE: The flying saw is moving in synchronism with the material web, cutting can start.
flyingSawInStartPos	BOOL	FALSE	TRUE: FlyingSaw is in initial position (startposition) FALSE: FlyingSaw is out of initial position.
endSyncPosReached	BOOL	FALSE	Warning, if the flying saw passes the upper limit value of the synchronous range in the automatic mode
onTheSpotCutPossible	BOOL	FALSE	FALSE: No “On the spot cut” possible TRUE: The flying saw is in its start position, an “On the spot cut” is possible
onTheSpotCutDone	BOOL	FALSE	The “On the spot cut” was executed
syncNotPossible	BOOL	FALSE	FALSE: The last synchronizing operation was successful TRUE: The last synchronizing operation was not able to be executed under the specified conditions
forSyncToLate	BOOL	FALSE	FALSE: The last synchronizing operation was successful TRUE: The flying saw did not reach the start position in time for the next cut.
errorID	WORD	0	Error identification, refer to error messages errorID <= 16#000_8000: warning errorID >= 16#0000_8000: error
state	eOperation ModesFlying SawType	OMFS_ ERROR	Actual mode OMFS_ERROR: ERROR mode OMFS_DISABLE: DISABLE mode OMFS_AUTOMATIC: STARTPOSITION mode OMFS_AUTOMATIC: AUTOMATIC mode OMFS_MANUAL: MANUAL mode OMFS_NOTHING_SELECTED: No mode change OMFS_CHANGING: Mode change active
flyingSawAxisActive	BOOL	FALSE	The FlyingSaw axis is active

8.3.6 Functionality

The *FBFlyingSaw* function block is the most important element of the application. The connected flying saw is parameterized and controlled using this block.

The block and the operating mode changeover must be parameterized in the user program in order that the *FBFlyingSaw* executes the required functions.

The *FBFlyingSaw* is parameterized as explained in **Chapter 8 Program Description**. When parameterizing, basic values of the application are defined, which generally do not change, such as e.g. specifying the name of the axis or all geometrical data, such as *r64DistanceToSensor, r64StartPos, r64StartSyncPos*, etc. It is only permissible to change these values at standstill (zero speed) - and they require a re-initialization (restart) of the *FBFlyingSaw*.

The individual operating modes of the *FBFlyingSaw* are described in **Chapter 3.2 Operating modes of the application SIMOTION Flying Saw**. The procedure to change over the operating mode is described in more detail here in the following sections.

Basic state of the FBFlyingSaw

After being initialized, *FBFlyingSaw* is in the DISABLE operating mode.

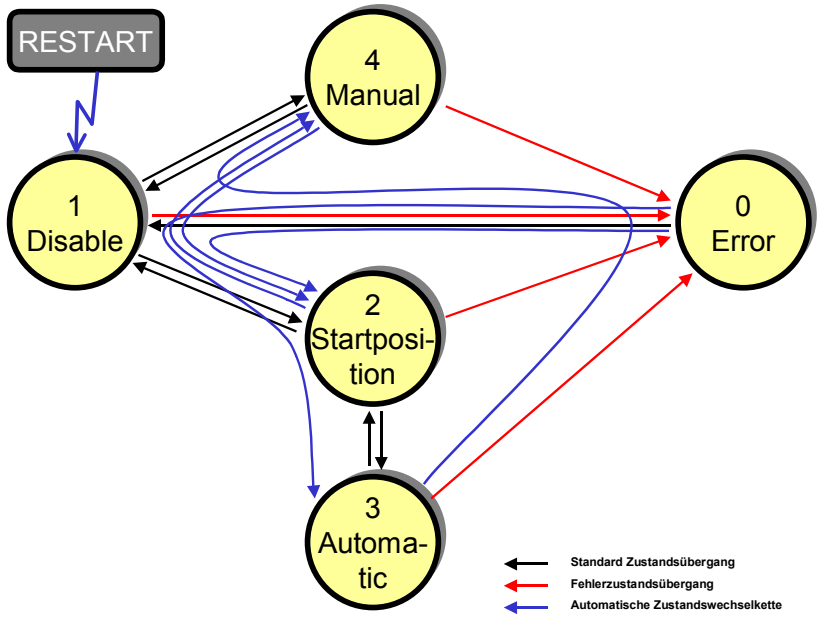
Initialization must be executed each time after the control has booted by setting the restart bit (*gasFlyingSaw[0].sUsersInterface.boRestart*) in the startup task. (This is realized automatically when using the *startupFlyingSaw()* program in the UNIT *FS_Progs*)

As a consequence, the block is brought into its initial state, important parameters are checked for plausibility, if possible adapted, and if not possible, parameterizing errors are displayed for user information at the error output.

Mode changeover sequence

The mode manager of the function block is shown in the following diagram. It includes the possible operating states that can be activated and clearly shows the possible transitions, that the user may initiate:

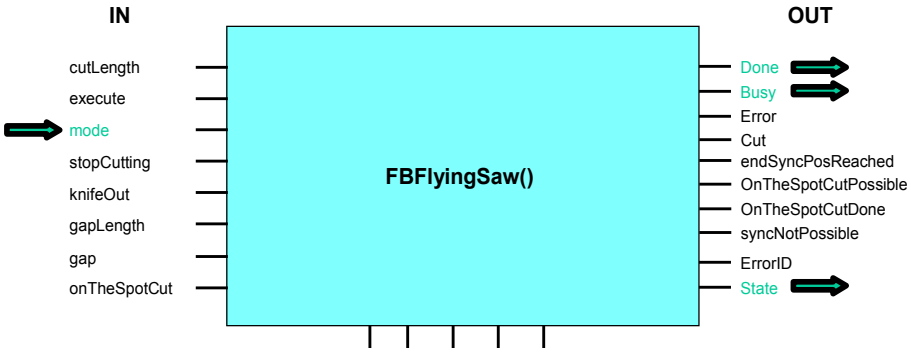
Figure 8-1: Possible state transitions at *FBFlyingSaw*



The actual block state is displayed at the ***state*** output

.

Figure 8-2: Inputs and outputs of the mode changeover



In order to transition the block into a new state, the number of the required target state must be specified at the *mode* input. The changeover is activated by a rising edge on the *execute* input.

The active changeover is signaled in the new block state with a high signal at the *busy* output. If all of the actions associated with the changeover have been successfully carried-out and the new state has been reached, this is flagged by a high signal at the *done* block output. The number of the new state can be read-off

at the *state* output.

Checking the input parameters and parameterization of the technology settings that are absolutely required (mandatory)

This program function is only executed after a “Restart” or if it was activated by the user using the restart bit of the *UsersInterface*.

The following are monitored:

- The existance and the type of specified TOs.
- Deviations with respect to what has been specified are displayed as error messages.
- Parameter assignments of the technology objects that are absolutely required are executed in an additional program section.

Axis monitoring

The function block is switched into the error state (ERROR operating mode) if there is an error at the technology object axis of the flying saw.

However, the monitoring function is only active if the parameterization check of the technology objects was successfully completed.

Using the restart bit (*sUsersInterface. boRestart*)

The restart bit must be set at each new start of the SIMOTION or when changing the parameterization. This is the reason that an initialization routine is run in the *FBFlyingSaw* and a parameter check is made and the block is brought into a defined state.

Note

We recommend that the program *startupFlyingSaw* of the unit *FS_Progs* is used in the startup task in order to initialize the block *FBFlyingSaw*.

The restart bit should also be set there in order to bring the block into a defined state!

Note

At restart, the block checks essential technology parameters as well as the existence of saved technology objects.

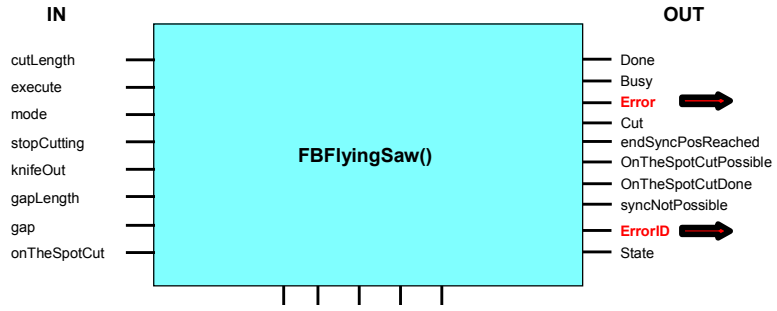
After the block has been successfully initialized, the restart bit is automatically reset.

Error handling

If faults occur in the block itself or in technology objects and technology functions involved while *FB FlyingSaw* is operational, then these are signaled using the block

outputs ***error*** and ***errorID***.

Figure 8-3: Block outputs for fault messages



The ***error*** bit signals an error while an error code is output at output ***errorID***; this can be analyzed using the error analysis function.

Note

The ***error*** output is only set if the ***execute*** input is set. The ***errorID*** and **state = 0** (ERROR) are still output even after ***execute*** has been reset!

8.3.7 Error messages

Table 8-1: List of the possible error priorities

ErrorID	Priority	Description
16#xxxx_0xxxx	Lowest priority	Warning
16#xxxx_4xxx	Low priority	Warning, execution possible with error
16#xxxx_8xxx	High priority	Error, no execution possible
16#xxxx_Cxxx	Highest priority	Critical error

Warnings

Warnings are indicated by the status of the outputs *error* and *errorID*: *error* = FALSE and *errorID* <> 16#0000_0000.

Table 8-2: Warnings

ErrorID	Description
16#0000_0000	No warning
16#060C_4002	FlyingSawAxis changed over to modulo axis
16#060C_4003	Modulo length of the FlyingSawAxis is adapted to (*r64EndPos + 500.0*) Only valid when using test environment (e.g. demo case!)
16#060C_4004	StartValue of the FlyingSawAxis is adapted to - (r64DistanceToSensor + 50.0) Only valid when using test environment (e.g. demo case!)
16#060C_4005	The axis as master value was switched to **linear without modulo**
16#060C_4006	The axis as master value was switched to **linear with modulo**
16#060C_4007	The external encoder as master value was switched to **linear without modulo**

16#060C_4008	The external encoder as master value was switched to **linear with modulo**
16#060C_4009	Modulo length of axis as master value was set to 0.0
16#060C_400A	Modulo length of axis as master value was set to MAXIMUM_CUTLENGTH * 3000
16#060C_400B	Modulo length of external encoder as master value was set to 0.0
16#060C_400C	Modulo length of external encoder as master value was set to MAXIMUM_CUTLENGTH * 3000

Errors

Error messages are indicated by the status of the outputs *error* and *errorID*: *error* = TRUE and *errorID* <> 16#0000_0000.

If the error was caused by a TO-specific command called from the FB, then its return parameter (**RetDINT** or **FR**) is kept in the internal data area of the FB for further evaluation by the user.

Table 8-3: Errors

Error ID	Description
16#0000_0000	No error
16#060A_8001	Error at the FlyingSawAxis
16#060A_8002	Error at the FlyingSawAxis synchronous operation
16#060A_8003	Actual position of FlyingSawAxis > *r64EndPos*
16#060C_8001	The TO axis FlyingSawAxis is not set-up
16#060C_8002	There is no TO set-up as master value
16#060C_8003	There is no TO measuring input set-up
16#060C_8004	The TO measuring input is not connected to leading value
16#060C_8005	TO Flying Saw Axis - Error adaption drive data
16#060C_8006	TO Master Value Axis - Error adaption drive data
16#060C_8007	TO Master Value External Encoder – Error adaption drive data
16#060C_8008	Error modulo settings master value
16#060F_8001	Invalid mode selection from the ERROR mode
16#060F_8002	Invalid mode selection from the DISABLE mode
16#060F_8003	Invalid mode selection from the STARTPOSITION mode
16#060F_8004	Invalid mode selection from the AUTOMATIC mode
16#060F_8005	Invalid mode selection from the MANUAL mode
16#060F_8006	Wrong mode
16#0611_8001	DISABLE to DISABLE Error when stopping the FlyingSawAxis (_move v=0), Details in *i32RetDINTStopFSA*
16#0611_8002	DISABLE to DISABLE Error when stopping the FlyingSawAxis (_move v=0) Details in *i32FRStopFSA*
16#0612_8001	DISABLE to STARTPOSITION Error when positioning the FlyingSawAxis (_pos) Details in *i32RetDINTPosFSA*
16#0612_8002	DISABLE to STARTPOSITION Error when positioning the FlyingSawAxis (_pos) Details in *i32FRPosFSA*

Error ID	Description
16#0621_8001	STARTPOSITION to DISABLE Error when stopping the FlyingSawAxis (_move v=0) Details in *i32RetDINTStopFSA*
16#0621_8002	STARTPOSITION to DISABLE Error when stopping the FlyingSawAxis (_move v=0) Details in *i32FRStopFSA*
16#0622_8001	STARTPOSITION to STARTPOSITION Error when positioning the FlyingSawAxis (_pos) Details in *i32RetDINTPosFSA*
16#0622_8002	STARTPOSITION to STARTPOSITION Error when positioning the FlyingSawAxis (_pos) Details in *i32FRPosFSA*
16#0603_8001	AUTOMATIC RUN Error when activating gearing of the FlyingSawAxis (_enablegearing) Details in *i32RetDINTEnableGearingFSA*
16#0603_8002	AUTOMATIC RUN Error when deactivating gearing of the FlyingSawAxis (_disablegearing) Details in *i32RetDINTDisableGearingFSA*
16#0603_8003	AUTOMATIC RUN Error when deactivating gearing of the FlyingSawAxis (_disablegearing) Details in *i32FRDisableGearingFSA*
16#0603_8004	AUTOMATIC RUN Error when superimposed positioning the FlyingSawAxis for Gap (_pos) Details in *i32RetDINTPosSuperimposedFSA*
16#0603_8005	AUTOMATIC RUN Error when superimposed positioning the FlyingSawAxis for Gap (_pos) Details in *i32FRPosSuperimposedFSA*
16#0603_8006	AUTOMATIC RUN Error when positioning the FlyingSawAxis back to start position (_pos) Details in *i32RetDINTPosFSA*
16#0603_8007	AUTOMATIC RUN Error when positioning the FlyingSawAxis back to start position (_pos) Details in *i32FRPosFSA*
16#0631_8001	AUTOMATIC to DISABLE Error when stopping the FlyingSawAxis (_move v=0) Details in *i32RetDINTStopFSA*
16#0631_8002	AUTOMATIC to DISABLE Error when stopping the FlyingSawAxis (_move v=0) Details in *i32FRStopFSA*
16#0632_8001	AUTOMATIC to STARTPOSITION Error when positioning the FlyingSawAxis (_pos) Details in *i32RetDINTPosFSA*
16#0632_8002	AUTOMATIC to STARTPOSITION Error when positioning the FlyingSawAxis (_pos) Details in *i32FRPosFSA*
16#0641_8001	MANUAL to DISABLE Error when stopping the FlyingSawAxis (_move v=0) Details in *i32RetDINTStopFSA*
16#0641_8002	MANUAL to DISABLE Error when stopping the FlyingSawAxis (_move v=0) Details in *i32FRStopFSA*

Error ID	Description
16#0642_8001	MANUAL to STARTPOSITION Error when positioning the FlyingSawAxis (_pos) Details in *i32RetDINTPosFSA*
16#0642_8002	MANUAL to STARTPOSITION Error when positioning the FlyingSawAxis (_pos) Details in *i32FRPosFSA*

8.4 FBCutPositionDetection

Block to determine the cutting position of the flying saw.

8.4.1 Block name

FBCutPositionDetection

8.4.2 Task

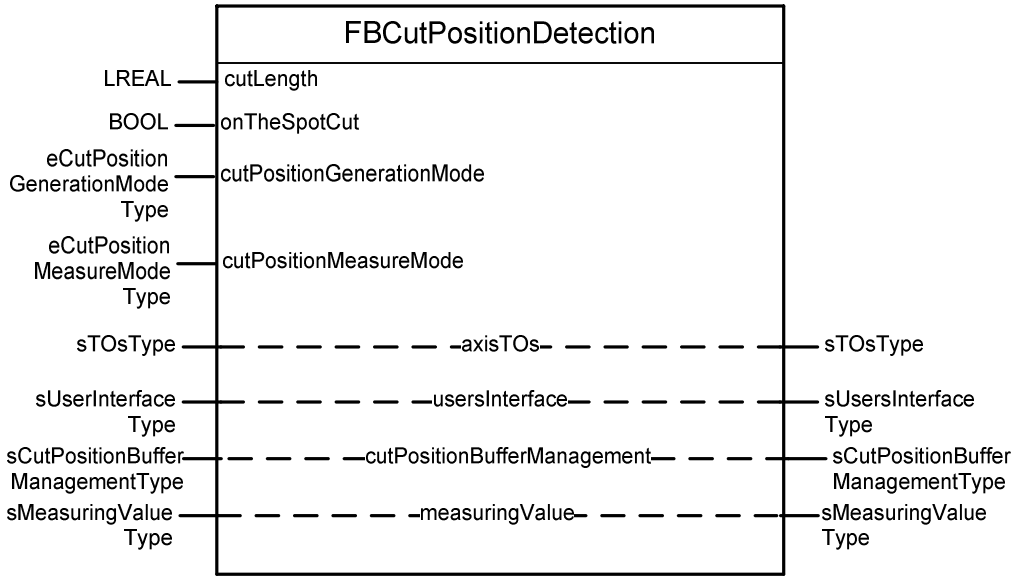
Determining the cutting positions

8.4.3 Integration in the run-time system

The function block can be called in each cyclic task.

The block is used by the SIMOTION Flying Saw application and does not have to be linked-in by the user.

8.4.4 Graphic representation of the block



8.4.5 Block parameters

Input Parameters

Name	Data type	Initial value	Description
cutLength	LREAL	800.0	[mm] Element length of the cut material.
onTheSpotCut	BOOL	FALSE	By setting this signal, a synchronous position is immediately generated for the next cut on the material web at the “sensor position”.
CutPositionGeneration Mode	eCutPositionG enerationMode Type	CP_ CALCULATED	There are two techniques to generate cutting positions: CP_CALCULATED: Cut position is calculated by breaking material web down by cutlength CP_MEASURED: Cut position is detected using the measuring probe and stored in the CutPositionBuffer
CutPositionMeasure Mode	eCutPosition MeasureMode Type	CPM_ SIMULATED	Method to measure cut position, only valid when CP_MEASURED activ: CPM_DETECTED: Cut position is detected using sensor and prink mark and entered in CutPositionBuffer. CPM_SIMULATED: Cut position is culculated by cutlength and entered in CutPositionBuffer

Input / output Parameters

Name	Data type	Initial value	Description
axisTOs	sTOsType		Definition of the axes, external encoders and measuring probes
usersInterface	sUsers InterfaceType		Interface to input dynamic and geometrical parameters
cutPositionBuffer Management	sCutPosition Buffer Management Type		Internal data area to enter precise cutting position and cutting position deviation for the cutting position determination.
measuringValue	sMeasuring ValueType		Internal data area to determine the cutting position.

8.4.6 Functionality

To determine the cutting position, the block has two different techniques:

- CP-MEASURED: The presetting of cut position setpoints is done via CutPositionBuffer
- CP-CALCULATED: The cut position of material web remains constant. The presetting of cut position setpoint is done by breaking material web position down by actual cutlength.

CP-MEASURED

To determine the cutting position put in the CutPositionBuffer, the block has two different techniques:

- CPM_DETECTED: The cutting position is detected using the measuring function with an activated sensitivity range at the distance of the cut length from the last measured position.
  After being activated, the cutting position determination function initially waits for the first sensor signal. Its position (*measuringValue.r64CutPosAtSensor*) is interpolated using the geometrical data *usersInterface.r64DistanceToSensor* and *usersInterface.r64StartSyncPos* to the cutting position (*measuringValue.r64CutPosAtSyncPos*) and is entered in the cutting position buffer as the first cutting position. For all additional cutting positions, a sensitivity range is now placed around the possible cutting position. To do this, the next cutting position expected is interpolated using the actual cut length to *measuringValue.r64NextCutPosAtSensor* and a tolerance bandwidth of *usersInterface. r64CPDeltaToleranceRange* is applied. If a measured value is determined, then this is interpolated as new cutting position and entered in the cutting position buffer. If a cutting position was not able to be measured within the range that had been selected, then the interpolated cutting position *measuringValue.r64NextCutPosAtSensor* is interpolated as new cutting position and entered in the cutting position buffer. At the same time, *measuringValue i16CutPositionError* is incremented by 1. Users can utilize this

variable for diagnostics/error monitoring. This counter is reset to 0 if the measurement was successful.

- CPM_SIMULATED: With the activation, the cutting position determination function immediately keeps one of the “measured values” referred to a fictitious sensor. Activating an “immediate cut” (OnTheSpotCut) initiates the same procedure as when activating the cutting position determination function – a “measured value” referred to the fictitious sensor is immediately kept.
  Its position (*measuringValue.r64CutPosAtSensor*) is, using the geometrical data *usersInterface.r64DistanceToSensor* and *usersInterface.r64StartSyncPos* interpolated to the cutting position (*measuringValue.r64CutPosAtSyncPos*) and is entered as the first cutting position into the cutting position buffer.
  For all additional cutting positions, the position value that was last saved is subtracted from the actual position value of the material web. If this difference is greater than the actual cut length, the new cutting position is determined from the sum of the position value saved and the actual cut length - and saved as position value.

CP-CALCULATED

In this operation mode there is no activity necessary.

8.4.7 Error messages

No errors are generated.

8.5 Measured value memory system blocks used in the flying saw application

The flying saw system uses these blocks. As standard, the user does not have to access them!

8.5.1 Deleting and resetting the measured value memory FBCutPositionBufferReset

Deleting and resetting the measured value memory.

Block name

FBCutPositionBufferReset

Task

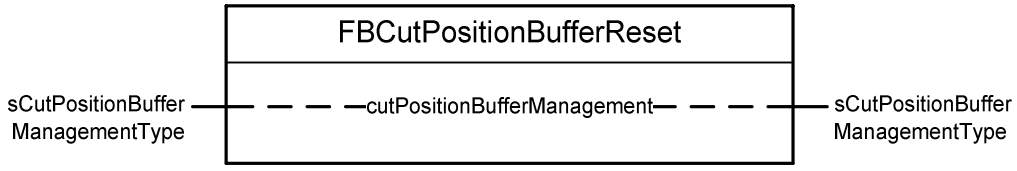
Deleting and resetting the measured value memory.

Integration in run-time system

The block can be called in any task.

The block is used by the SIMOTION Rotary Knife or SIMOTION Flying Saw application and does not have to be linked-in by the user.

Graphic representation of the block



Block parameters

Input / output parameters

Name	Data type	Initial value	Description
cutPositionBuffer Management	sCutPositionBuffer ManagementType		Structure of the measured value memory with all of the required control signals and pointers (refer to the chapter, data types)

Functionality

- Deletes all of the memory contents by overwriting memory elements with 0.0 or FALSE
- Resets all pointers to 0
- Resets all status signals to FALSE

Error Messages

No errors are generated.

8.5.2 Entries in the measured value memory FBCutPositionBufferIn

Enters a measured value (position measured value and/or position deviation) into the measured value memory.

Block name

FBCutPositionBufferIn

Task

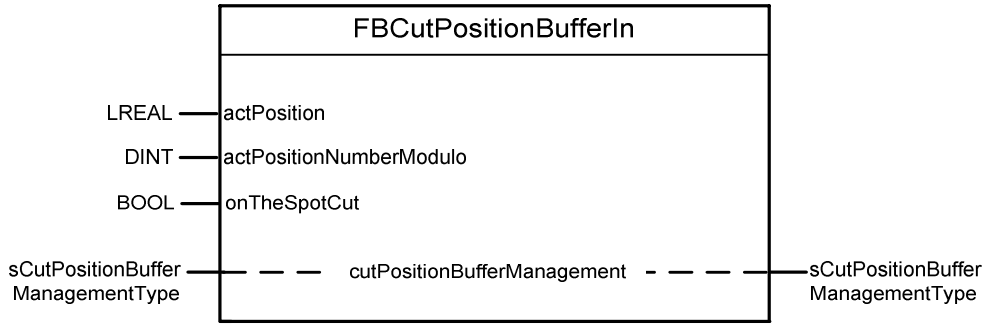
With each call, the values present at the block (position measured value and position deviation) are entered into a free location of the measured value memory.

Integration in run-time system

The block can be called in any task.

The block is used by the SIMOTION Rotary Knife or SIMOTION Flying Saw application and does not have to be linked-in by the user.

Graphic representation of the block



Block parameters

Input parameters

Name	Data type	Initial value	Description
cutPosition	LREAL	0.0	[mm] Position measured value
cutPositionNumber Modulo	DINT	0	[-] Number modulo cycles of position measured value
onTheSpotCut	BOOL	FALSE	"Immediate cut" active (on the spot cut)

Input / output parameters

Name	Data type	Initial value	Description
cutPositionBuffer Management	sCutPositionBuffer ManagementType		Structure of the measured value memory with all of the required control signals and pointers (refer to the chapter, data types)

Functionality

- The position measured value is entered into the measured value memory.

- The number of modulo cycles of the actual postion is entered into the measured value memory.

Error messages

No errors are generated.

8.5.3 Reading out the measured value memory FBCutPositionBufferRead

Reading measured values from the measured value memory.

Block name

FBCutPositionBufferRead

Task

If the block is being used in the SIMOTION Rotary Knife application, then it detects whether a new position deviation was entered and initiates a corresponding position correction.

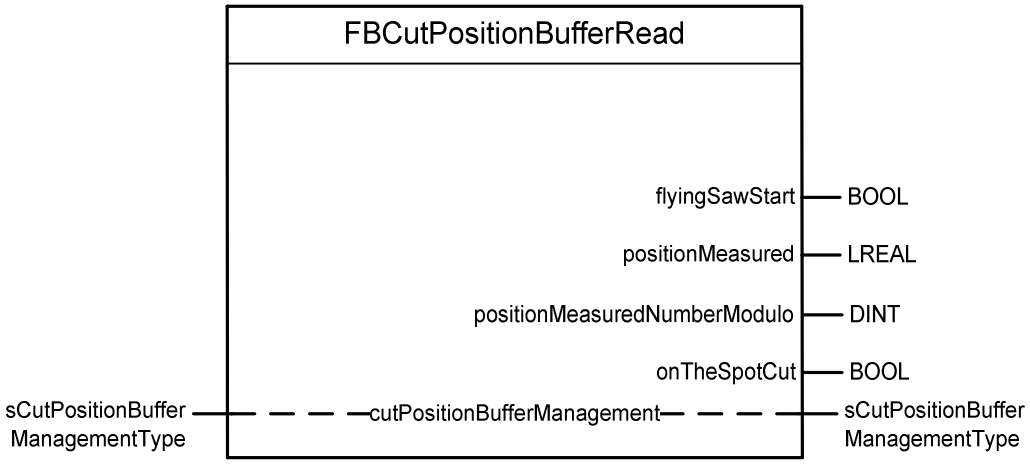
If the block is being used in the SIMOTION Flying Saw application, then it detects as to whether a new position measured value was entered and initiates an additional processing cycle.

Integration in run-time system

The block can be called in any task.

The block is used by the SIMOTION Rotary Knife or SIMOTION Flying Saw application and does not have to be linked-in by the user.

Graphic representation of the block



Block parameters

Output parameters

Name	Data type	Initial value	Description
flyingSawStart	BOOL	FALSE	Start condition of the flying saw
positionMeasured	LREAL	0.0	[mm] next sync position
positionMeasuredNumberModulo	DINT	0	[-] Number modulo cycle of next sync position
onTheSpotCut	BOOL	FALSE	TRUE: On the spot cut active

Input / output Parameters

Name	Data type	Initial value	Description
cutPositionBuffer Management	sCutPosition Buffer Management Type		Structure of the measured value memory with all of the required control signals and pointers (refer to the chapter, data types).

Functionality

Depending on the calling application, the block executes the following activity:

The block checks as to whether a new position measured value (*cutPosition*) was entered. If yes, the flag "*flyingSawStart*" is set and the measured position is transferred as synchronous position for the next processing cycle.

Error messages

No errors are generated.

8.5.4 Exporting from the measured value memory FBCutPositionBufferOut

A measured value (position measured value and position deviation) is exported from the measured value memory.

Block name

FBCutPositionBufferOut

Task

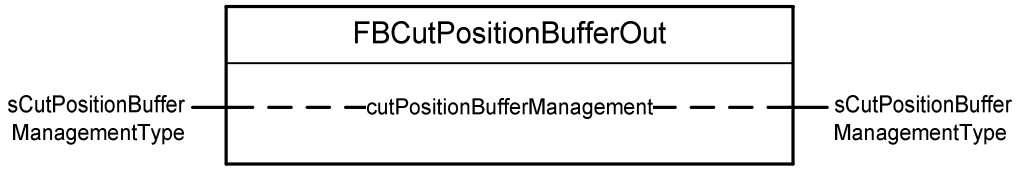
After the processing operation, the function block deletes the last measured value (position measured value and position deviation) from the measured value memory and releases the memory for memory management.

Integration in run-time system

The block can be called in any task.

The block is used by the SIMOTION Rotary Knife or SIMOTION Flying Saw application and does not have to be linked-in by the user.

Graphic representation of the block



Block parameters

Input / output parameter

Name	Data type	Initial value	Description
cutPositionBuffer Management	sCutPositionBuffer ManagementType		Structure of the CutPositionBuffer with all of the required control signals and pointers (refer to the chapter, data types)

Functionality

When the block is called, the actual measured value (position measured value and position deviation) are deleted from the measured value memory.

Error messages

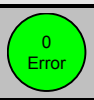
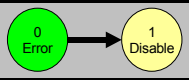
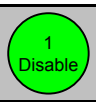
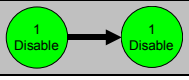
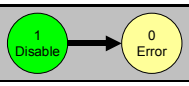
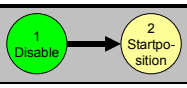
No messages are generated.

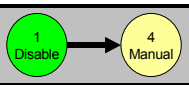
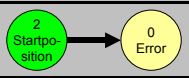
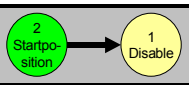
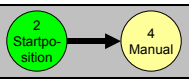
Attachment

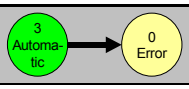
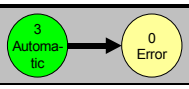
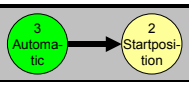
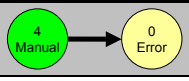
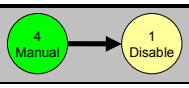
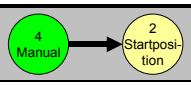
9 Description of the State Changes

The following table describes the function sequences that are hidden behind the modes and the mode changes.

Table 9-1 Description of the sequences at the state change (state transition) in the *FBFlyingSaw*

Step	Function	Comment
Mode ERROR (0)		
ERROR → DISABLE		
0	All ErrorID memories are reset	
1	Calls the **_resetAxisError()** for the flying saw axis Calls the **_resetFollowingObject()** for the flying saw axis If the axis and the synchronous object have no error condition, then the DISABLE mode is activated.	
Mode DISABLE (1)		
DISABLE → DISABLE		
0	The stop command is prepared	
1	**_move()** with **velocity = 0.0** is called for the flying saw axis	
2	**_disableAxis()** is called for the flying saw axis If the axis is disabled, then the DISABLE mode is activated.	
DISABLE → ERROR		
0	Preparation	
1	**_disableAxis()** is called for the flying saw axis If the axis is disabled, then the ERROR mode is activated	
DISABLE → STARTPOSITION		
0	The enable command is prepared	
1	**_enableAxis()** is called for the flying saw axis If the axis is enabled, then advance to step 2	
2	**_resetAxisError()** is called for the flying saw axis If the axis is OK, advance to step 3	
3	**_pos()** is called for the flying saw axis with the direction of rotation for the shortest path If the starting position is reached, then the “starting position” mode is activated.	

DISABLE → MANUAL		
0	The enable command is prepared.	
1	**_enableAxis()** is called for the flying saw axis If the axis is enabled, then the MANUAL mode is activated	
Mode, STARTPOSITION (2)		
STARTPOSITION → ERROR		
0	Preparation	
1	**_disableAxis()** is called for the flying saw axis If the axes are disabled, then the ERROR mode is activated	
STARTPOSITION → DISABLE		
0	The stop command is prepared.	
1	**_move()** with **velocity = 0.0** is called for the flying saw axis	
2	**_disableAxis()** is called for the flying saw axis If the axis is disabled, then the DISABLE mode is activated	
STARTPOSITION → AUTOMATIC		
0	Resetting the *CutPosBuffer* Activation of the measuring system is required. Activation the AUTOMATIC operating mode	
STARTPOSITION → MANUAL		
0	This state change is achieved using an automatic state change sequence that reaches the manual state through the disable state.	
Mode, AUTOMATIC (3)		
Automatic cycle		
0	Wait until a valid synchronous position is available If yes, then advance to step 1	
1	**_enableGearing()** is called to synchronize the flying saw axis to the material web at the master and slave positions If synchronization was successful, then continue with step 3, if not, then continue with step 2	
2	**_disableGearing()** is called to delete (clear) the command from the command queue. If this is successful, then continue with step 0	
3	The *cut* output bit is set The actual synchronous position is deleted (cleared) from the *CutPosBuffer*	

4	Wait until *stopCutting* signal appears If the *gap* signal is present, then continue with step 5 If not, then continue with step 6	
5	**_pos()** is called for the flying saw axis using the merge mode SUPERIMPOSED_MOTION_MERGE in order to create a gap with the superimposed motion. When completed, then continue with step 6.	
6	If the *knifeOut* signal appears, then **_pos()** is called for the flying saw axis to return to the starting position. When completed, then continue with step 1.	
AUTOMATIC → ERROR		
	_disableAxis() is called for the flying saw axis If the axis is disabled, then the ERROR mode is activated	
AUTOMATIC → DISABLE		
0	The stop command is prepared	
1	**_move()** with **velocity = 0.0** is called for the flying saw axis If the axis is disabled, then advance to step 2	
2	**_disableAxis()** is called for the flying saw axis If the axis is disabled, then the DISABLE mode is activated	
AUTOMATIC → STARTPOSITION		
0	Resets the *cut* output bit If the *knifeOut* signal is present, then prepare the position command and advance to step 1	
1	**_pos()** is called to return the flying saw axis back to its starting position.	
Mode, MANUAL (4)		
MANUAL → ERROR		
	_disableAxis() is called for the auxiliary axis and the flying saw axis If the axes are disabled, then the ERROR mode is activated.	
MANUAL → DISABLE		
0	The stop command is prepared	
1	**_move()** with **velocity = 0.0** is called for the flying saw axis	
2	**_disableAxis()** is called for the flying saw axis If the axes are disabled, then the DISABLE mode is activated	
MANUAL → STARTPOSITION		

0	**_pos()** is called for the flying saw axis with the direction of rotation for the shortest path If the starting position is reached, then the “starting position” mode is activated.	

10 General information on the application

10.1 Scope of supply

The "flying saw" package comprises:

- Program
  S7/Scout project as ZIP file:
  SIMOTION_FlyingSaw _V2.1.zip
- TestFB-Program
  S7/Scout-Projekt as ZIP-File:
  SIMOTION_FlyingSaw _TestFB_V2.1.zip
- Library
  HTML-Export as ZIP-File:
  SIMOTION_FlyingSaw _LFSLib_V2.1.zip
- Documentation
  Implementation/Operating Instructions as PDF file
  Manual_SIMOTION_FlyingSaw_V2.1.pdf

10.2 Revisions/Author

Table 10-1: Revisions/Author

Version	Date/Revision	Author
V 1.0	Generation	H.-E. Böhm
V 1.1	Change-over Ethernet	H.-E. Böhm
V 1.2	Protool templates trilingual	Ch. Pabst H.-E. Böhm
V 1.2	Changes General Notes	A. Hagelauer
V 1.3	13.04.06 Scout V4.0	H.-E. Böhm
V 1.3 (V 2.0!)	04.04.07 Functional Changes: - synchronization according to leading axis values. (asymptotic synchronization) - Ratio between measured and effective velocity of material web	H.-E. Böhm
V 1.4	06.06.08 Replacing ProTool by WinCflex / SCOUT V4.1.1.6	P. Tabori H.-E. Böhm
V 2.0	March 2010 changeover to the StyleGuide Adapt to ongoing projects Essential simplifications	H.-E. Böhm
V2.1	2011 up to September 2014 Extension by break down process Error correction Improvements error detection Adaption to CutToLength-Line requirements	H.-E. Böhm

11 Contact partner

Application Center

SIEMENS

Siemens AG
Industry Sector
Drives Technologies Division
Motion Control Systems
DF FA PMA APC
Frauenauracher Str. 80
91056 Erlangen
Fax: +49 (0) 9131-98–1297
mailto: tech.team.motioncontrol@siemens.com

12 Please help us to become even better

DF FA PMA APC
Application Center

D – 91056 Erlangen
Fax: +49 (0) 9131/98–1297
E-Mail: tech.team.motioncontrol@siemens.com

Sender:
Name:
Department:
City:
Telephone:
E-Mail:

Evaluation of the document

Is this subject helpful/ beneficial for you? ❑ Yes ❑ No

How high are the benefits of this application for you in the following working phases?

	Very high					Very low
– own training/ information	❑	❑	❑	❑	❑	❑
– conceptual phase	❑	❑	❑	❑	❑	❑
– engineering/writing code	❑	❑	❑	❑	❑	❑
– commissioning	❑	❑	❑	❑	❑	❑

How is this application from the didactic perspective?

	Very high					Very poor
– scope	❑	❑	❑	❑	❑	❑
– layout/design	❑	❑	❑	❑	❑	❑
– clarity	❑	❑	❑	❑	❑	❑

Can the application be transferred to your own application?

Very easily					Not easily
❑	❑	❑	❑	❑	❑

Do you require support for this application? ❑ Yes ❑ No

Other comments .....

.....