Print Standard Add-On TRC1000

SIMOTION & SINAMICS

Application description • August 2015

Applikationen & Tools

Answers for industry.

SIEMENS

Siemens Industry Online Support

This article is taken from the Siemens Industry Online Support. The following link takes you directly to the download page of this document:

http://support.industry.siemens.com/cs/de/de/view/59753224

Caution

The functions and solutions described in this article confine themselves to the realization of the automation task predominantly. Please take into account furthermore that corresponding protective measures have to be taken up in the context of Industrial Security when connecting your equipment to other parts of the plant, the enterprise network or the Internet. Further information can be found under the Item-ID 50203404.

http://support.industry.siemens.com/cs/de/de/view/50203404

You can also actively use our Technical Forum from the Siemens Industry Online Support regarding this subject. Add your questions, suggestions and problems and discuss them together in our strong forum community:

http://www.siemens.com/forum-applications

SIEMENS

SIMOTION
Print Standard Add-On TRC1000

Warranty and liability

Note

The Application Examples are not binding and do not claim to be complete regarding the circuits shown, equipping and any eventuality. The Application Examples do not represent customer-specific solutions. They are only intended to provide support for typical applications. You are responsible for ensuring that the described products are used correctly. These application examples do not relieve you of the responsibility to use safe practices in application, installation, operation and maintenance. When using these Application Examples, 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 Application Examples at any time without prior notice.
If there are any deviations between the recommendations provided in these application examples and other Siemens publications − e.g. Catalogs − the contents of the other documents have priority.

We do not accept any liability for the information contained in this document.

Any claims against us – based on whatever legal reason – resulting from the use of the examples, information, programs, engineering and performance data etc., described in this Application Example shall be excluded. Such an exclusion shall not apply in the case of mandatory liability, e.g. under the German Product Liability Act (“Produkthaftungsgesetz”), in case of intent, gross negligence, or injury of life, body or health, guarantee for the quality of a product, fraudulent concealment of a deficiency or breach of a condition which goes to the root of the contract (“wesentliche Vertragspflichten”). The damages for a breach of a substantial contractual obligation are, however, limited to the foreseeable damage, typical for the type of contract, except in the event of intent or gross negligence or injury to life, body or health. The above provisions do not imply a change of the burden of proof to your detriment.

Security information

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, solutions, machines, equipment and/or networks. They are important components in a holistic industrial security concept. With this in mind, Siemens’ products and solutions undergo continuous development. Siemens recommends strongly that you regularly check for product updates.

For the secure operation of Siemens products and solutions, it is necessary to take suitable preventive action (e.g. cell protection concept) and integrate each component into a holistic, state-of-the-art industrial security concept. Third-party products that may be in use should also be considered. For more information about industrial security, visit http://www.siemens.com/industrialsecurity.

To stay informed about product updates as they occur, sign up for a product-specific newsletter. For more information, visit http://support.automation.siemens.com.

Preface

Target

This document describes the application/programming concepts and functions, which are required for a register control solution with SIMOTION and TRC1000.

The objective of this document is to provide the user with detailed information about the structure of the SIMOTION Print Standard Add-On TRC1000 project.

Some reasons for creating such a standard application are:

- reduce necessary engineering time
- provide the user with tested and proven concept
- maintain and further develop know-how on printing industry specific functionality
- provide an example software that can run on a “SIMOTION register control demo unit” for demonstration and training purposes
- allow Siemens to better support its customers

Target Readers

This document applies to programmers and application engineers in the printing and converting industries.

Main Contents

The main purpose of this application is the description of the integration of TRC1000 hardware to a machine project based on Print Standard. Also the software handling of the TRC device and the evaluation of the return values for using them with the standard register controller.

Limitations

This document does not include a description of SIMOTION and SINAMICS in general. General knowledge of …

- SCOUT
- SIMOTION
- SINAMICS
- WinCC flexible
- SIMOTION Print Standard master application

… is necessary.

Scope of supply

Print Standard Add-On TRC1000 is available for download on the customer support website at applications. The scope of supply contains:

- Scout Project: Print Standard Add-On TRC1000 Vx.x.x.zip
- Documentation Print Standard Add-On TRC1000 Vx.x.x.pdf (this document)

Table of contents

1 Application description

The basic functionality of the application Print Standard Add-On TRC1000 is the handling of the print mark detector IDS-PN from manufacturer "*Wiedeg"*. Additionally the application shows how to integrate the functionality of register control and how visualization on a panel or PC can be realized.

Handling of TRC1000 is based on a cyclic and acyclic standard interface between sensor and SIMOTION. Via these interface all necessary functionality is being managed. E.g. parameterization, reading of the oscilloscope data and much more.

Moreover to the basic communication functionalities, it provides a lot of additional functions to make the handling of the device as easy as possible.

Print Standard Add-On TRC1000 functionalities:

- Collection of parameterization data and handling of transfer the printing job data to the TRC hardware
- Evaluation of the markfiled and mark error
- Calculation of register errors depending on measuring mode and selected marks
- Preparation of analogue oscilloscope curves for graphic display on the HMI screens and gate settings
- Evaluation of cyclic communication, monitoring of TRC sign of life, fault handling
- TRC mode handling

1.1 Basic information and data

1.1.1 System requirements

This application was developed and tested using:

Software:

- SIMOTION SCOUT 4.4 HF 2
- WinCC flexible 2008 SP3 Upd5
- SIMATIC STEP 7 V5.5 + SP4 + HF5
- Wiedeg sensor firmware V2.2.0
- GSD file date 2014-11-14

Hardware:

- SIMOTION D445-2 on training case TK-SIM-D435 built to be connected to 1 AC 230 V or 1AC 115 V with transformer
- SLM 5kW
- Double Motor Module 3A
- 2 (additional) Synchronous Motors (1FK7060-3BF71-1BA0) (Encoder AS24DQI P03)

- Wiedeg print mark detector IDS-PN

1.1.2 Supported Panels (HMI)

- SIMATIC IPC 277D
- PC Runtime

1.2 Integrated register control with SIMOTION Print Standard

In modern printing presses each print unit is individually driven by servo motors and the synchronism is achieved by an electronic line shaft (ELS) with virtual master. The register is adjusted by the offset value (angle) of the synchronism.

Thus the synchronized drive is the actuator of the register adjustment. The register controller needs to manipulate the synchronism between the virtual master and the real axis of the print unit drive.

In order to control the register in a closed loop, a register measured value is required. Therefore register marks are printed additionally to the print view onto the substrate. Depending on the measurement method and printing process different types of marks are used.

In comparison to an external register control system an integrated system provides different advantages.

- faster register control and less startup waste by integration into the electronic line shaft function
- less HW components
- smaller cabinet size
- same HW for motion and register
- integration into machine operation, automation and startup sequence
- open system, extendable by OEM
- same engineering tools for automation, motion control and register control
- clear responsibility: OEM with the support of Siemens
- Sensor HW is the only cost addition to integrate a register control
- End-user support by OEM possible

Siemens offers different solutions for register control in printing machine concepts with SIMOTION/SINAMICS in combination with the Print Standard application. Separate application examples for each solution are available depending on the sensor type.

- Print Standard Add-On TRC5000 (camera solution)
- Print Standard Add-On TRC3000 (sensor solution)
- Print Standard Add-On TRC1000 (sensor solution)

Table 1-1: Solutions for integrated register control

TRC1000	TRC3000	TRC5000
Different types of wedge marks	Different types of wedge marks	Point marks
Fiber optic monochrome sensor	Fiber optic RGB sensor	Camera with flash
Round spot light	Bar sport light	Integrated flashlight
PROFINET IRT	PROFINET IRT	Ethernet, TM41
Webspeed 1000m/min	Webspeed 1000m/min	Webspeed 1000m/min
2 marks	3/20 marks (by configuration)	16+5 (for reference)

1.3 TRC1000 Hardware

NOTE

This chapter provides a short summary of the TRC1000 device.

For more detailed information please see the Wiedeg user manual IDS-PN User Manual en V2.0.pdf.

For the TRC1000 there are two hardware components available:

- Wiedeg IDS-PN: Main device with integrated single head print mark sensor for standard mark field detection
- Wiedeg DS: Extension device for a second (double head) channel (for detection of marks on both sides of the web or a mark-arrangement next to each other)

Figure 1-1 TRC1000 hardware “IDS-PN” (“singlehead”-version)

Figure 1-2 TRC1000 hardware “DS” (extension for “doublehead” measurement)

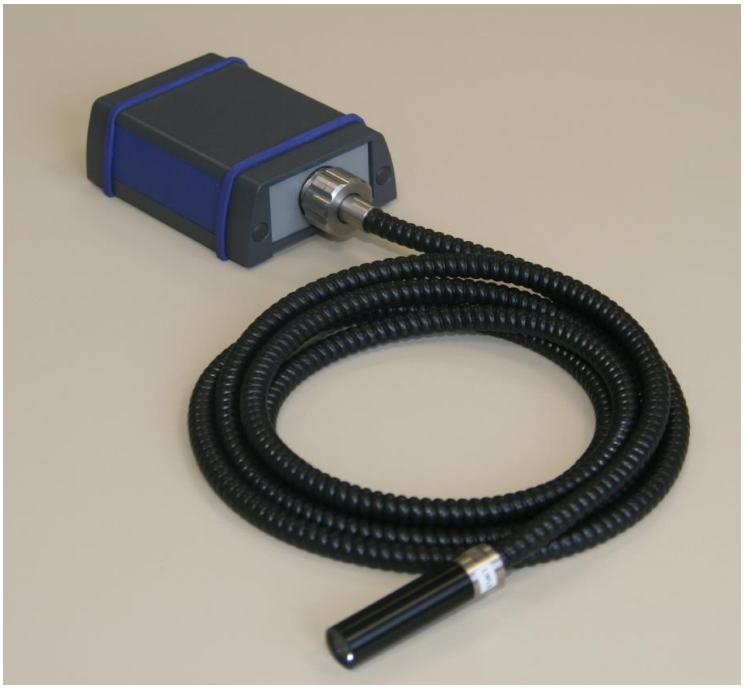
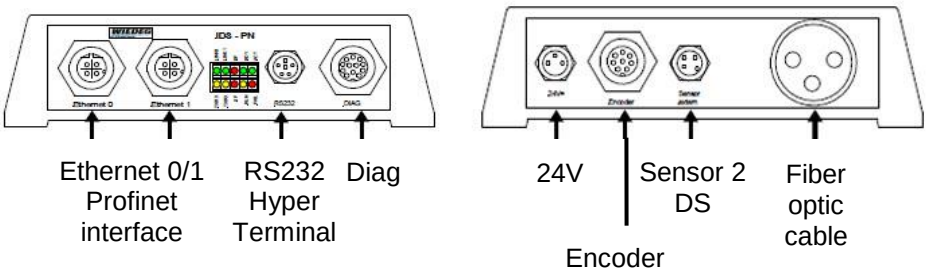


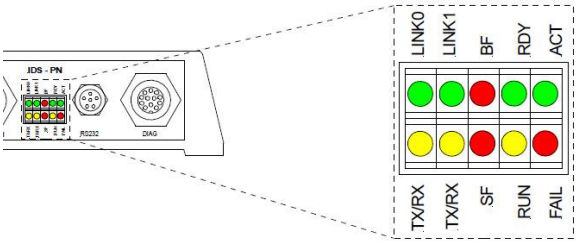
Table 1-2 Characteristics of TRC1000

Features	TRC1000
Sensor Type	Fiber optic sensor (cable length 2.5m or 5.0m)
Evaluation method	monochrome sensor
Light source	round spotlight
ATEX	II 3 G [Ex op is] IIC (Zone 2 for FOC)
Communication	PROFINET IO with IRT
Web speed	1000 m/min
Make	Wiedeg (Siemens trade good)
Backside printing	Extension device
Mark (incl. reference)	2
Mark type	Wedge, double wedge, block, double block
Automatic mark recognition	Bar code (3 bars)
Mark arrangement Space requirement	In-line in print free area across with 2nd FOC
Register measurement	Integrated
Register resolution	< 5µm
Measurement method	Mark to cylinder Mark to mark Mark to mark (2 sensor heads)
Register control strategy	Two marks only (selection by parameter)
Register recognition	Automatic and manual
Recognition of lacquer or reflecting material	By deflecting sensor head (restricted use)

Features	TRC1000
Recognition diagnostic	1 trace / scanning head
Trace resolution	0.5ms max. 798 values
End of press mode	no

1.3.1 Interfaces





Link 0/1	PN Diagnostic
TX/RX	Data exchange
BF	Bus fault
SF	System fault
RDY	FW and netX ready
RUN	netX information
ACT	Device state
FAIL	

1.4 Integration of TRC1000 into a machine concept

The cyclic communication is based on PROFINET IRT. Therefore the integration into an existing machine concept is easy. Every TRC-hardware is equipped with two PN-Interfaces (Ethernet) for series connection of several devices.

The figure 1-3 shows a TRC1000 standard configuration. Every print unit is equipped with one TRC. The TRC for printing unit #1 is the “doublehead”- version for front and backside detection. All other print units are equipped with the TRCs in “singlehead”-version.

Figure 1-3 Machine concept

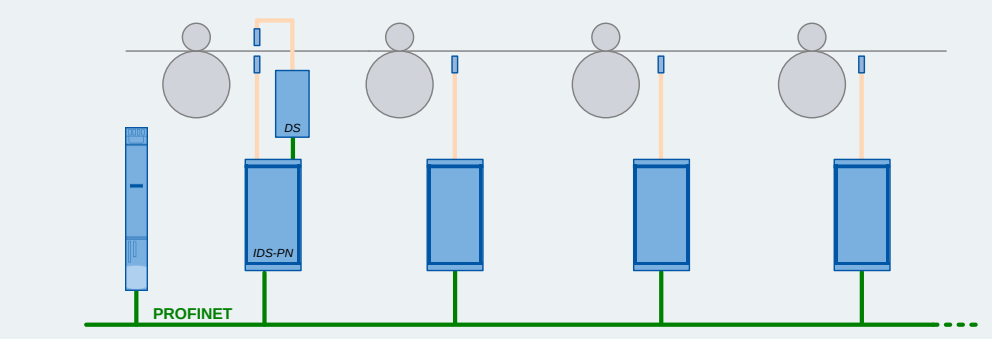


Figure 1-4 shows the detailed connection between the different system components. The SIMOTION as master device in the PROFINET IO system controls the printing cylinder.

Via PROFINET IRT the cyclic data will be transferred and received from SIMOTION.

The parameterization and the analog and digital oscilloscope data will be transferred via acyclic communication.

The communication methods are summarized in table 1-3.

Figure 1-4 System overview

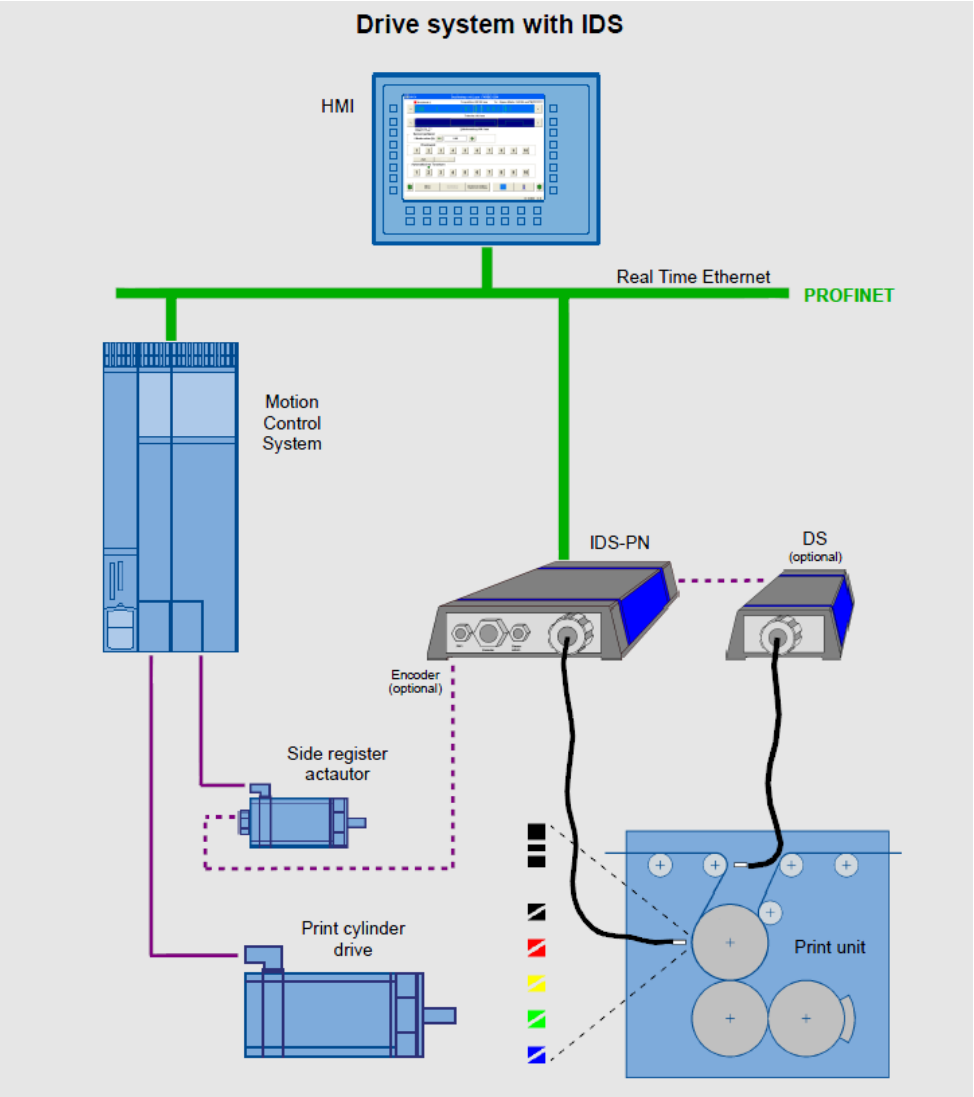


Table 1-3 TRC1000 communication

Communication	e.g.
cyclic data (PROFINET IRT)	Status words, control words, cylinder position, velocity, actual mark error
acyclic data (LDPV1 read/write parameter command)	Parameterization of TRC (mark field, mark definition, control mode…) Actual TRC data (Error numbers, actual gate positions, FW version…)

The TRC1000 is working with all known types of wedge and block marks. Low contrast ink or reflecting materials are reliable to detect.

Typical applications for the TRC1000 solution are rotogravure printing and converting applications.

The example project contains the preconfigured HMI example screens for TRC parameterization, fault handling and evaluation of the oscilloscope data.

2 Application structure

The application for the TRC1000 print mark detector is totally integrated into the Print Standard application concept. Print Standard provides easy software standardization for different kinds of printing machines and several solutions for technological tasks.

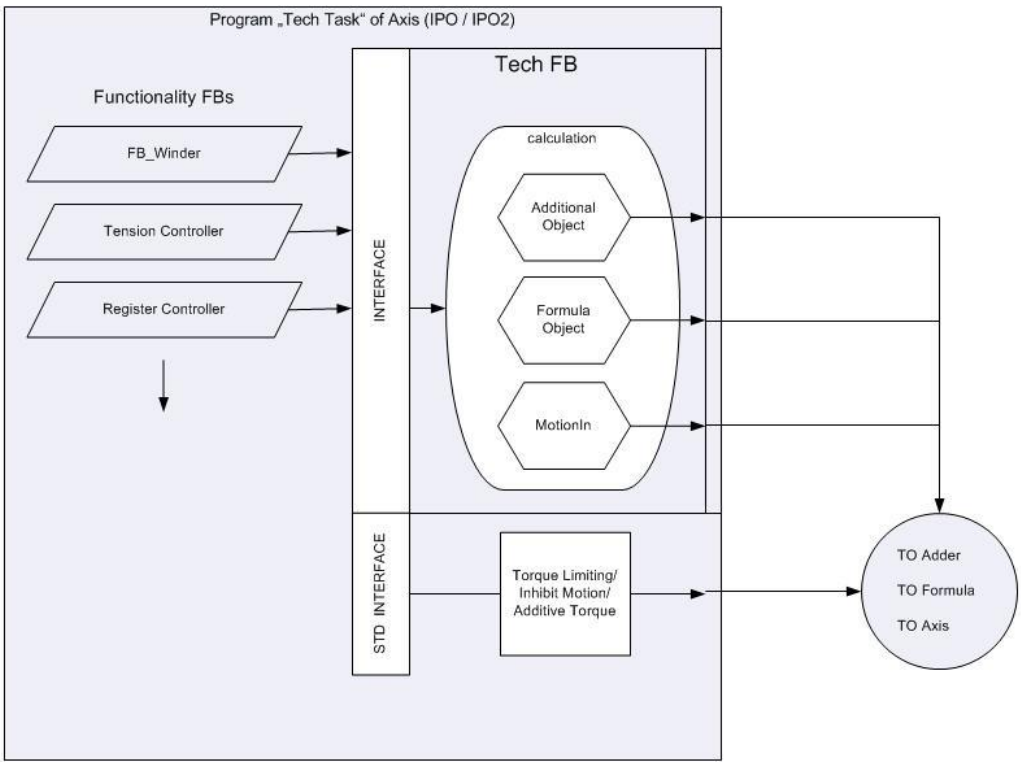
A user who is familiar with this software concept should be able to understand and integrate the TRC1000 software parts very fast into an existent project.

2.1 Print Standard and TRC1000

The function block “FBTech” of Print Standard offers a possibility to influence the axis basic motion setpoint channel by external or internal created additional values. The figure 2-1 shows this concept basically.

A technology function block e.g. winder, tension controller or register controller is calculating an additional velocity setpoint value. The value will be switched to the Print Standard “FBTech” for calculation of the entire motion vector. It will be directly connected to the axis additional object which is acting as interface to the axis setpoint channel (as long as the SIMOTION additional object is integrated as interface between user application and SIMOTION axis TO).

Figure 2-1 “FBTech” principle



2.2 TRC1000 software parts

The TRC1000 application works as a flexible module which could be integrated very easy into an existing Print Standard project. Generally it is also possible to run the application stand alone.

Basically the application consists of a library “LTRC1000” which contains the necessary function blocks and type definitions. The main function of these blocks is the handling of cyclic and acyclic communication to the hardware device and the print mark result evaluation.

The program parts S_Var_GI, S_H_Var_GI and pTRC1000 show exemplary the structure variable definitions, the function blocks calls and the collaboration of the TRC1000 in combination with the register controller function block “FBLRegCtrlController” (separate library “LRegCtrl”) and “FBTech” (library “LPrint”).

Additional to the SIMOTION project, the application contains example screens for the visualization with WinCC flexible. Depending on the project philosophy the HMI masks can be integrated and adapted to an existing HMI project or used stand alone as they are in the example project.

NOTE The HMI masks are only examples they are freely adaptable by the user depending on the project requirements.

NOTE Chapter 4 of this manual contains a more detailed description of the TRC1000 software parts, constants and variables.

2.3 WinCC flexible part

The standard application example is prepared for a maximum of 20 print units. The data array which are used for HMI communication are predefined to this length.

NOTICE **It’s not allowed to reduce the length also in machine concepts with less print units!**

In machines with less than 20 print units the user has the possibility to adapt the array length of the SIMOTION internal variables to the real number of printing units which are active in the machine. This helps to safe internal storage space and to get a clear and small structure for observations.

NOTE The adaptable variables and constants are declared in the global variable source (S_Var_GI) in the SIMOTION program part.

2.4 Additionally applications used together with TRC1000 application

Commonly the two following functionalities are used together with the register control functionality:

2.4.1 Insetting

Sometimes it is necessary to handle with preprinted materials. For example there is an extra printing unit at the end of the line or further improvement steps for the product are necessary before starting with the essential job.

Unfortunately the format of preprinted web is not exactly the same like specified, because of drying effects and other influences. In this case an adaption of the format is required.

In the Add-On application “Insetting” this problem is solved by changing the electronic gearing of the respective printing cylinder. The growing difference of the register deviation will be measured. Afterwards the adjustment of the gearing factor will synchronize the printing cylinder with the preprinted material.

The insetting functionality is provided in the library “LRegCtrl” which is already part of the TRC1000 example project. Also the program unit “pInsetting” is integrated in the project and the programs (pInsettingStartUp, pInsettingIpo, pInsettingBackround) are assigned to the SIMOTION execution system.

NOTE For the complete integration of the insetting functionality and further information, the insetting documentation “Print Standard AddOn Insetting.pdf” can be used.

2.4.2 DRD (Dynamic Register Decoupling)

A special behavior of a rotogravure printing machine is that the impression cylinder (presseur) and the printing cylinder have a high nip pressure. With this not only the ink transfer but also the web transport is realized. In rotogravure electronic line shaft (ELS) machines the length register movement is also performed by the printing cylinder. With this connection a register movement is not possible without influencing the web transport leading to a web stretch and web tension change. So every register movement is influencing the register stability of the following printing units in a negative way.

To avoid this effect, the Dynamic Register Decoupling (DRD) function block can be used.

The DRD function block forwards an active register adjustment to all following printing units. In this way these printing units will adjust themselves to the register adjustment before the effects – transferred via the web – are visible at the following printing units.

With the use of the DRD function block a more stable register control system is achieved, leading to a higher print quality.

The DRD functionality is provided in the library “LRegCtrl” which is already part of the TRC1000 example project.

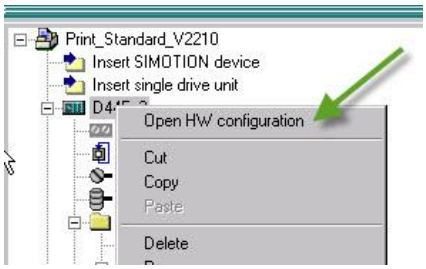
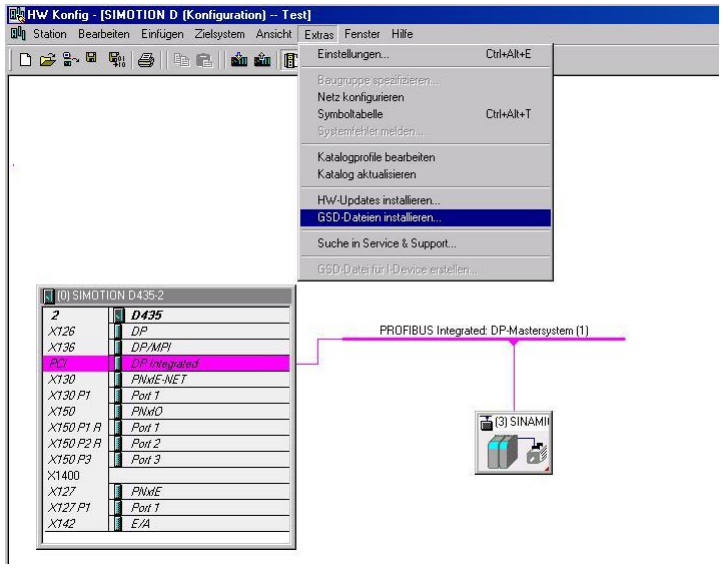
NOTE For the integration of the DRD functionality and further information, the DRD documentation “Print Standard AddOn DRD.pdf” can be used.

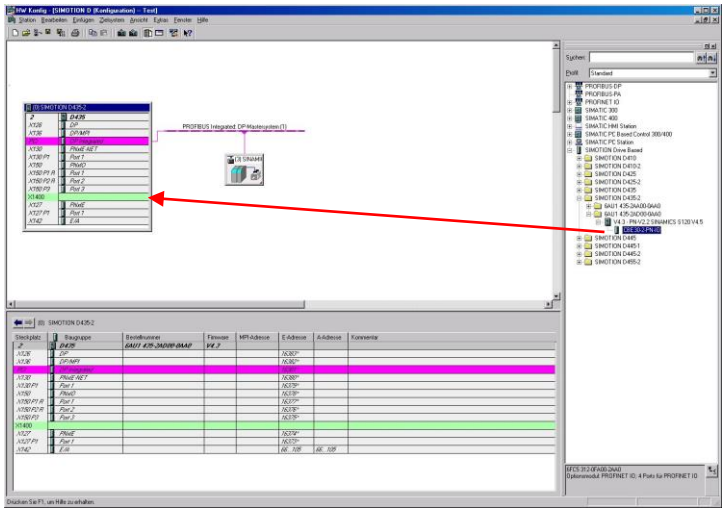
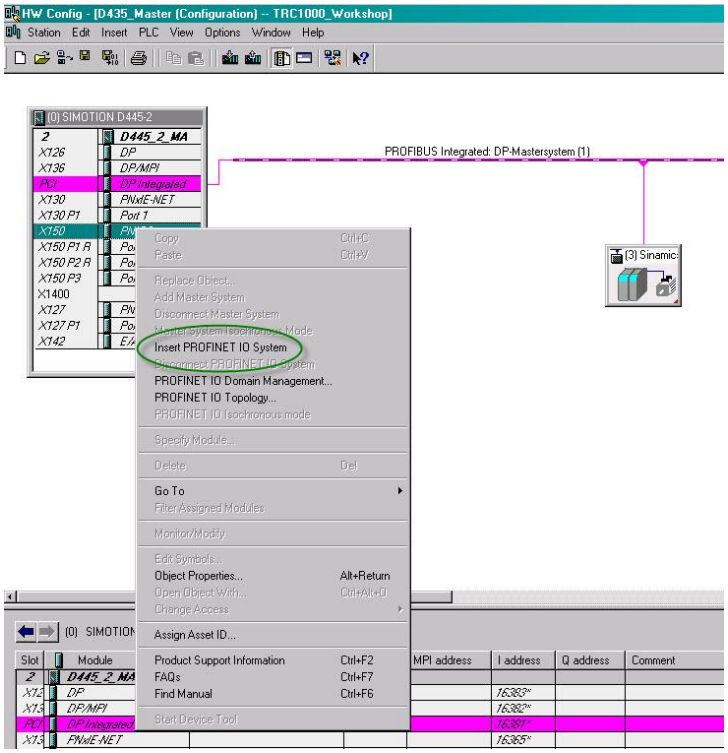
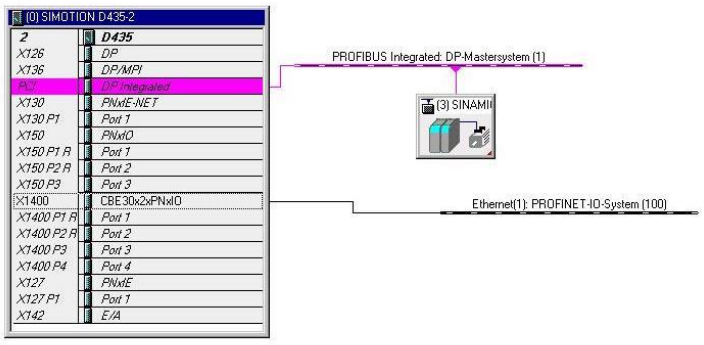
3 Integration

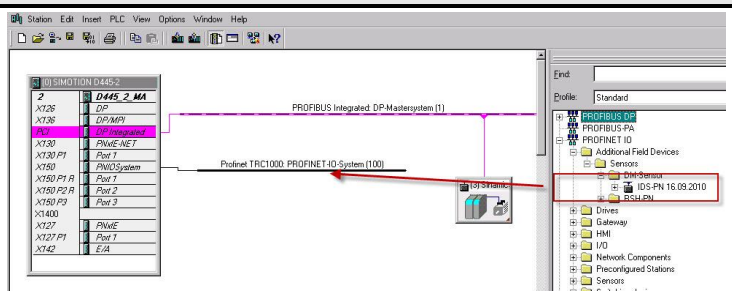
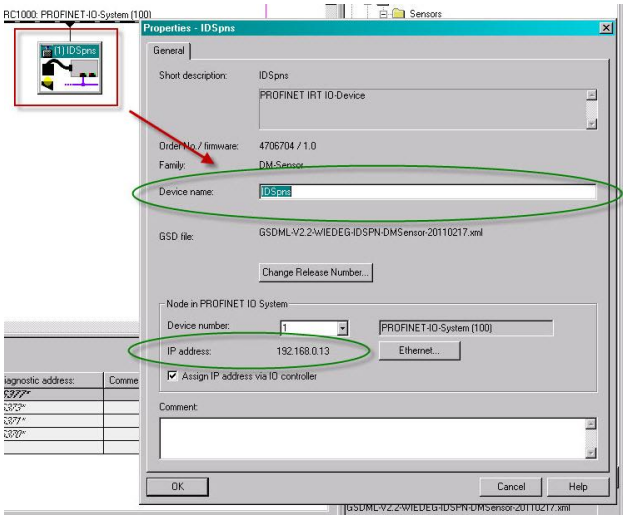
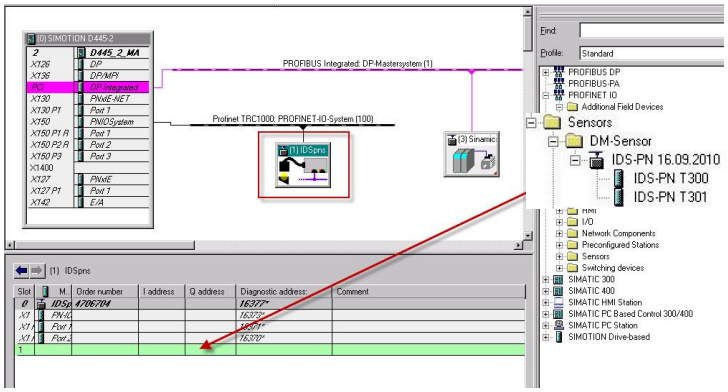
The following tables show how to integrate a TRC1000 device into a new or existing project. Moreover the integration of all used software parts and the HMI.

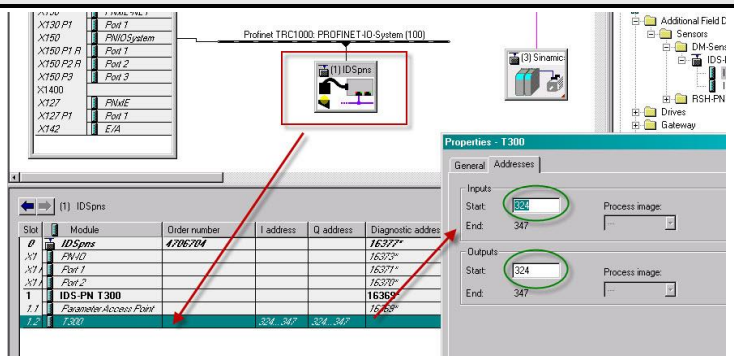
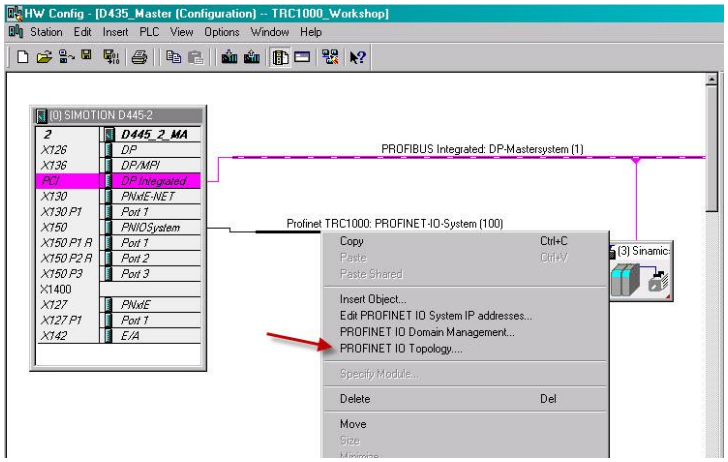
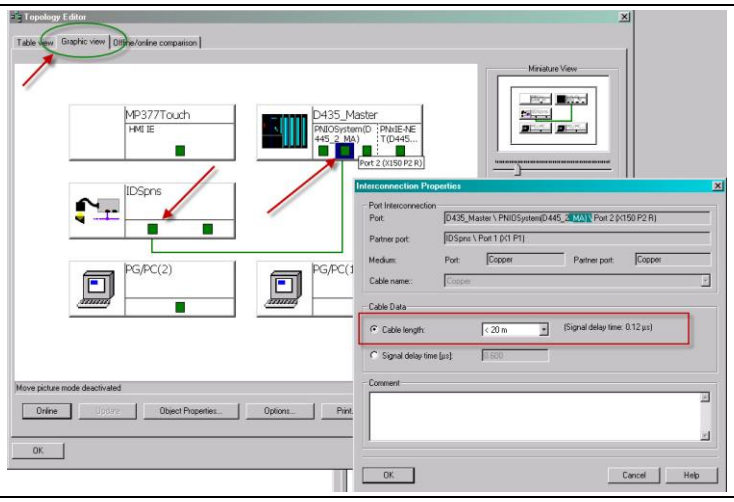
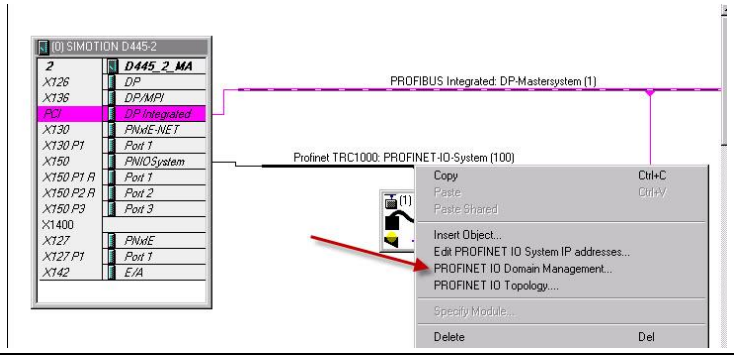
3.1 Integration of the TRC1000 hardware device

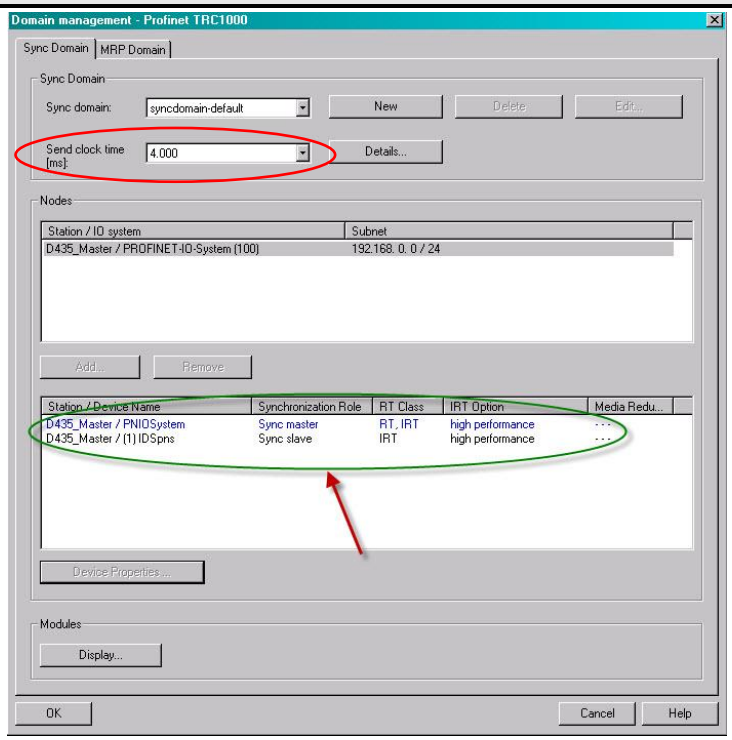
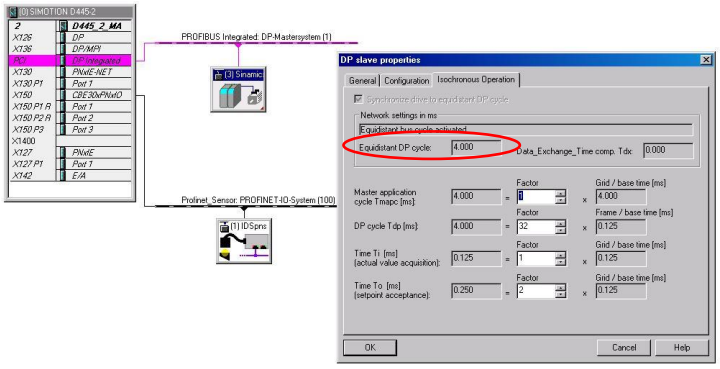
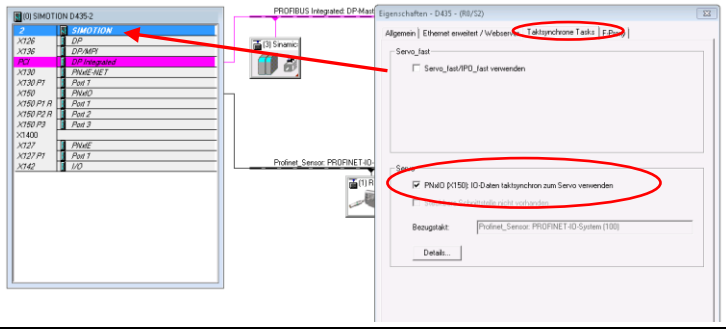
Table 3-1 Integration of TRC1000 hardware device

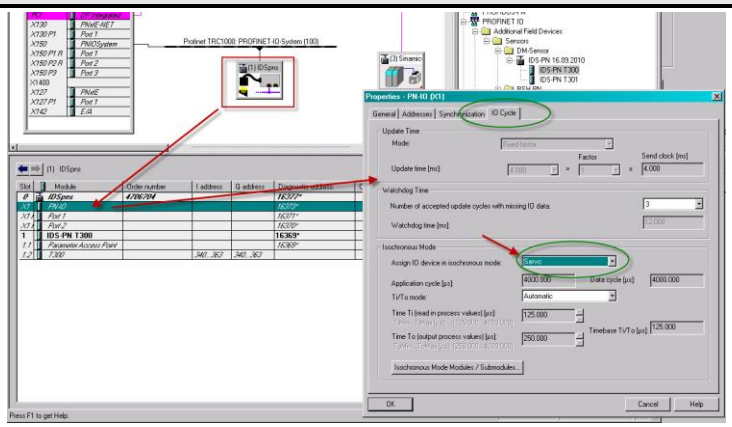
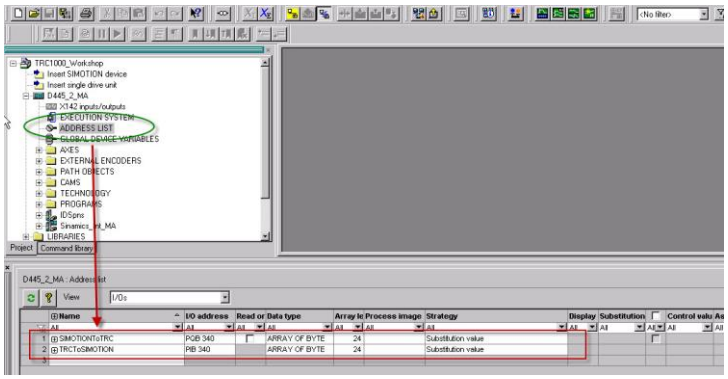
		Description
1.		Open the hardware configuration of the SIMOTION device in the project
2.		**Installing GSD-File:** To integrate the TRC1000 into the hardware configuration, the GSD-File of the TRC1000 has to be installed in the engineering-tool SIMOTION Scout. Go to “Options / install GSD-file”. Browse to the GSD-file and press “install”.

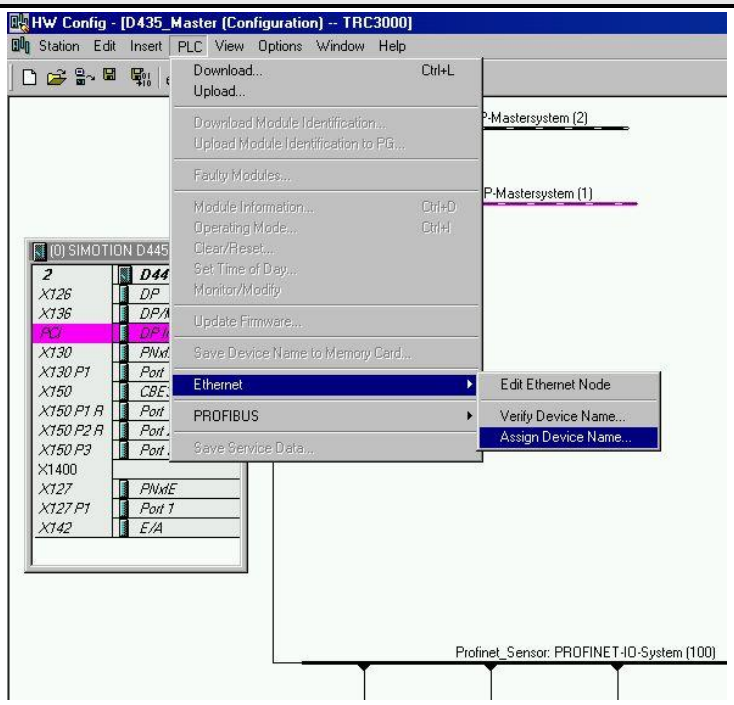
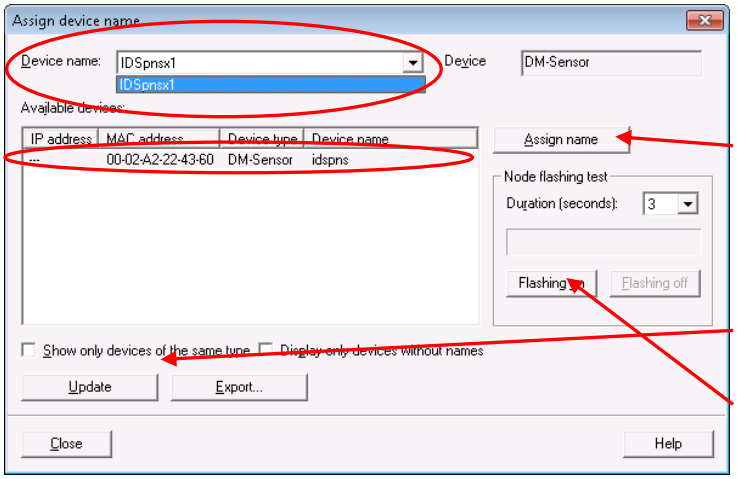
		Description
3.		**Insert Profinet I/O Controller:** (If there already exists an I/O Controller, step 2 can be skipped.) Select the suitable I/O Controller from the hardware catalogue on the right side and assign it to a free slot in the SIMOTION hardware rack.
4.		**Insert I/O-System:** (If there is already an I/O-System existing, step 3 can be skipped.) Right mouse click on the PROFINET I/O Controller “Insert PROFINET IO System”
5.		Now I/O devices (e.g. TRC1000) can be connected to the PROFINET-I/O-System.

		Description
6.		**Insert TRC1000 device :** (Premise: GSD file has been installed (step 2!) Select the TRC1000 device from the hardware catalogue (“PROFINET IO / additional field devices / sensors / DM-Sensor / IDS-PN) and shift it with drag & drop to the PROFINET-IO-System.
7.		Define the TRC name and the IP-address.
8.		**Select telegram for projected sensor unit:** Select the telegram in the hardware catalogue and shift it to Slot 1 of the sensor device. <u>Two telegrams are available:</u> - **300**: contains all available cyclic data of the sensor device (24 Bytes input and output data) - **301**: contains only 20 Bytes input and 24 Bytes output data. **Telegram 301 can be used if the external encoder isn’t used!**

		Description
9.		**Define addresses:** Define the TRC addresses by clicking on the telegram in slot 1.2. **NOTE:** **Start-Address of input and output data should be the same!**
10.		**Adjust the PROFINET IO topology:** Right mouse click on the PROFINET IO system and go to PROFINET IO Topology
11.		Select graphical view and connect the TRC PROFINET interface to a free port of the SIMOTION controller. Further TRC1000 devices has to be connected from port 2 of the previous device to port 1 of the following device. **NOTE:** **This wiring needs to match with the physical wiring!**
12.		Right mouse click on the PROFINET IO system and go to PROFINET IO Domain Management.

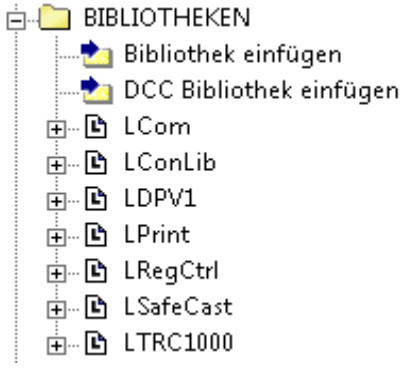
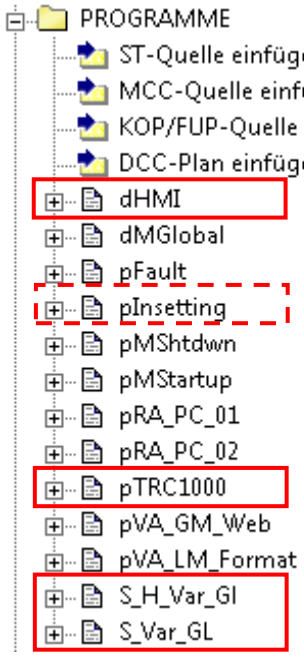
		Description
13.		**Define the synchronization role for every PN device:** SIMOTION: Sync master IDS-PN: Sync slave (IRT high performance) Select the send clock time of the I/O controller to the I/O devices. The TRC1000 sensor device supports the following send clock times: 1.000 ms 1.500 ms 2.000 ms 2.500 ms 3.000 ms 3.500 ms 4.000 ms **(This setting has to match with the DP slave properties of the SINAMICS_Integrated (step 14))**
14.		**DP slave properties:** Double-click on the SINAMICS_Integrated and check whether the master application cycle and the DP cycle matches with the send clock time of the I/O controller (step 13).
15.		**SIMOTION properties:** Double click on the SIMOTION device. In the tab “isosychronous tasks” the interface need to be operated synchronous to servo (set checkbox).

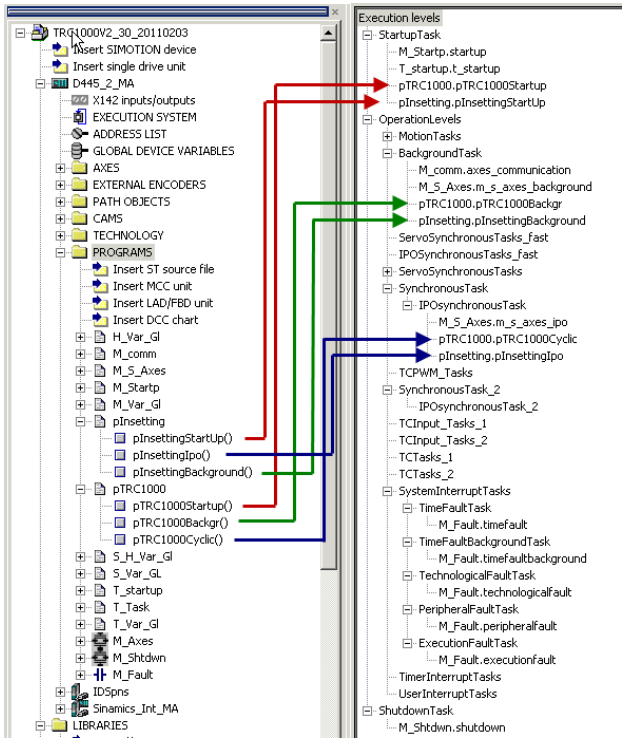
		Description
16.		**Select IO cycle:** Mark the device, double click on Port X1 “PN-IO” and go to the tab “IO Cycle”. Select “**Servo**” at “Assign IO device in isochronous mode”
17.		**Safe and compile** the hardware configuration. Close the hardware configuration and change back to the SCOUT project
18.		**Define the necessary IO-variables on the address-list of the controller:** Depending on the selected telegram: **300**: 24 Byte output data, 24 Byte input data **301**: 24 Byte output data, 20 Byte input data **Make sure the start addresses matches with the settings of step 9.** **There is no process image necessary! (No multiple IO access per task-cycle)**
19.		Safe and compile the project

		Description
20.		**Assign device names:** Open HW-Config and select the PROFINET-IO-System. Go to PLC → Ethernet → Assign Device Name
21.		Select the device by MACAddress, choose the device name from the drop down list and assign the name to the device by clicking the button “**Assign name**”. If your device is missing, click the “**Update**” button. You can also use the “**Flashing on**” button to detect the corresponding device.

3.2 Integration of SIMOTION libraries and programs

Table 3-2 Integration of software parts to the scout project

		Description
1.		Copy the necessary libraries from the example project or via XML-import: • LCom • LConLib • LDPV1 • LRegCtrl • LSafeCast • LTRC1000 • LPrint (resp. SPrint) Possibly the used hardware and the version needs to be selected at all library's first. **Accept and compile each library after integration!**
2.		Integrate necessary program sources: Units with global variables: • S_H_Val_GI • S_Var_GI **No changes of these sources are necessary!** Program sources: • pTRC1000 (need to be adapted to your specific project) • pInsetting (if required)

		Description
3.		**Assign programs to the respective tasks:** Startup-Task: • pTRC1000Startup • pInsettingStartUp Background-Task: • pTRC1000Backgr • pInsettingBackground IPO-Task: • pTRC1000Cyclic • pInsettingSynchronous

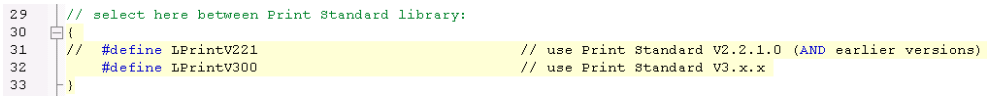
3.2.1 Selection of the Print Standard library version used in the project

The Print Standard version upgrade from V2.2.x.x to V3.x.x requires several changes inside libraries and program sources. To simplify these adaption for the user with TRC1000 Add-On version V3.2.2 some pre-processor commands has been implemented to define the used Print Standard version and perform the necessary adaption in the program automatically.

In the following libraries and program sources the pre-processor command has to be defined in the Interface section of the respective unit:

- LRegCtrl/cPublic
- S_Var_Gl
- pTRC1000
- pInsetting

Figure 3-1 Selection of Print Standard library version



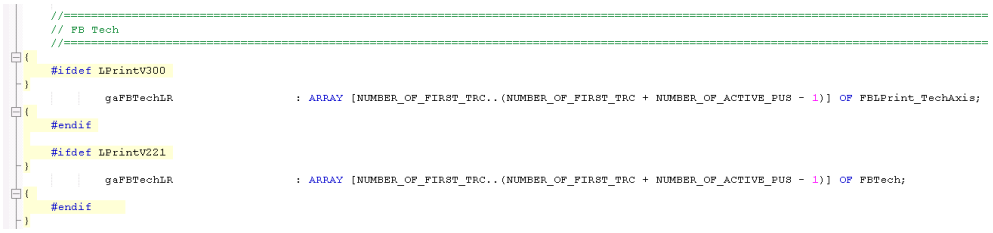
#define LPrintV221: Print Standard <= V2.2.1.0

#define LPrintV300: Print Standard >= V3.0.0

The version which is being used in the project need to be commented in, the one which is not used need to be commented out.

During compilation of the program code it's being decided between to "options" depending of the selected version. The following screenshot is showing one example:

Figure 3-2 Usage of pre-processor command in the program code

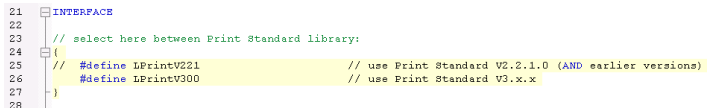
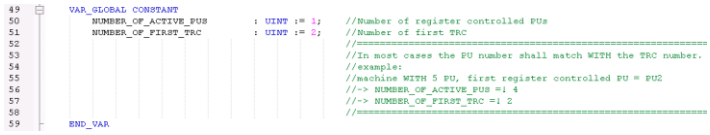


3.3 Adaption/Extension of the SIMOTION program for used TRC's/printing units

To adapt the example project to an own machine project or extend the project by further TRC's the following steps need to be done:

S_Var_GI

Table 3-3 Adaptions in S_Var_GI

	Adaption	Description
1.		**Pre-processor commands:** **#define LPrintV221/V300:** With this pre-processor command the version of the Print Standard library used in the project will be defined. #define LPrintV221: Print Standard <= V2.2.1.0 #define LPrintV300: Print Standard >= V3.0.0
2.		**Number of Sensors:** NUMBER_OF_ACTIVE_PUS defines the number of register controlled printing units. NUMBER_OF_FIRST_TRC defines the index of the first register controlled TRC. The number of structures and function blocks will be adapted via this global constants.

pTRC1000

NOTE All parts which need to be adapted by the user are labeled with the comment “User adaption necessary”.

pTRC1000Startup

Table 3-4 Adaptions in the startup program

	Adaption	Description
1.	`//=== // Register controller presettings //=== IF TRUE THEN // general settings HMI_Command_TRC1000.in.eTechnology := GRAVURE; HMI_Command_TRC1000.in.eControlAlg := symmetricalOptimumSync; HMI_Command_TRC1000.in.boEnableIntActionLR := TRUE; //HMI_Command_TRC1000.in.boEnableIntActionSR := TRUE; HMI_Command_TRC1000.in.r32LRRegCtrlKPWebWeb := 1.0; HMI_Command_TRC1000.in.r32LRRegCtrlKPWebCyl := 1.0; HMI_Command_TRC1000.in.r32LRRegCtrlIWebWeb := 10.0; HMI_Command_TRC1000.in.r32LRRegCtrlIWebCyl := 10.0; HMI_Command_TRC1000.in.r32SRRegCtrlKPWebWeb := 1.0; HMI_Command_TRC1000.in.r32SRRegCtrlKPWebCyl := 1.0; HMI_Command_TRC1000.in.r32SRRegCtrlIWebWeb := 10.0; HMI_Command_TRC1000.in.r32SRRegCtrlIWebCyl := 10.0;`	**Register Controller Pre-setting’s** Pre-setting’s of the register controller can be done in the startup task. In the example project the controller settings (kp, Ti) can be changed from HMI.
2.	`//-- // copy retain data (last parameterization to trc) // user adapt //-- IF TRUE THEN IF (gsTRCRetainDataSet.boRetainDatasetFilled = FALSE) THEN // no retain data available FOR u16SetParameterCounter := NUMBER_OF_FIRST_TRC TO (NUMBER_OF_FIRST_TRC + NUMBER_OF_ACTIVE_ HMI_Command_TRC1000.in.aboSelectPUactive[u16SetParameterCounter] := TRUE; END_FOR; // inch factors HMI_Command_TRC1000.in.r32RegInchLRFactorSlow := 0.1; //[mm] HMI_Command_TRC1000.in.r32RegInchSRFactorSlow := 0.1; //[mm] HMI_Command_TRC1000.in.r32RegInchLRFactorFast := 5; //[mm] HMI_Command_TRC1000.in.r32RegInchSRFactorFast := 5; //[mm] // HMI_Command_TRC1000.in.r32GateJerkFactorFast := 5; //[mm] // HMI_Command_TRC1000.in.r32GateJerkFactorSlow := 1; //[mm] // resolution factor grafic register error HMI_Command_TRC1000.in.r32Graph_Factor := 10;`	**Define retain data** In the very first start up program cycle the retain data can be defined here. Premise is, the bit boRetainDatasetFille = FALSE. In case this variable is TRUE, the retain data are copied to the main data structure in this section of the start up program.
3.	`//=== // Bits for control via HMI Control Screen //=== IF TRUE THEN HMI_axis_1_stdcIOActivateTestIO := TRUE; HMI_axis_2_stdcIOActivateTestIO := TRUE; HMI_axis_3_stdcIOActivateTestIO := TRUE; HMI_axis_4_stdcIOActivateTestIO := TRUE; END_IF;`	**Machine control via the “Machine control” HMI screen** If the “Machine control” HMI screens is used (e.g. example project) these bits need to be set to TRUE. In customer machine project the machine is controlled by PLC normally. In this case these bits need to be set to FALSE or commented out!

pTRC1000Backgr

Table 3-5 Adaptions in the background program

	Adaption	Description

1.	
	//----------
	// relative register inching
	//----------
	IF TRUE THEN
	{
	#ifdef LPrintV300
	}
	afTrigRegistrationDone[2](gsRA_PC_02AxisSTDcIO.OUT.boRegistrationDone);
	IF (gsRA_PC_02AxisSTDcIO.OUT.u16ActualModeNumber = 60)
	THEN
	HMI_axis_4_stdcIO.IN.boActivateRelRegister := TRUE; //RA_PC_02_STDcI
	IF NOT (boRegAktive)
	AND (gasTRCData[2].sStdcIO.in.r32InchDistMMLR <> 0.0)
	THEN
	HMI_axis_4_stdcIO.IN.boNegativeRelRegister := gasTRCData[2].sSt
	HMI_axis_4_stdcIO.IN.r64RegisterSetpoint := (360.0 * gasTRCDa
	boRegAktive := TRUE;
	END_IF;
	IF (afTrigRegistrationDone[2].q)
	THEN
	HMI_axis_4_stdcIO.IN.r64RegisterSetpoint := 0.0;
	boRegAktive := FALSE;
	gasTRCData[2].sStdcIO.out.boSetRefActive := FALSE;
	gasTRCData[2].sStdcIO.out.boSetRegActive := FALSE;
	END_IF;
	END_IF;
	//add further TRCs here
	{
	#endif
		Register Inching
		
		Depending on the register inching solution of the project the commands need to be connected.
		In the example project the register inching will be done by using Print Standard relative inching. The respective signals from HMI are connected to the Print Standard-STDcIO variables.
		
		Copy the code and adapt the indices.

2.	//----------
	// call FBTRC1000Backgr
	//----------
	IF TRUE THEN
	ai32SensorLogAddress[2] := 324;
	//ai32SensorLogAddress[3] := xxx;
	//add further TRCs here ...
	FOR u16FBBackgroundCounter := NUMBER_OF_FIRST_TRC TO (NUMBER_OF_FIRST_TRC
	gaFBTRC1000Backgr[u16FBBackgroundCounter](PrintUnitNr := UI
	,SensorLogAdress := ai
	,TRCConfig := gs
	,TRCData := ga
	);
	END_FOR;
	END_IF;
		FBTRC1000Backgr
		
		In this section the function block FBTRC1000Backgr is called.
		The function block is called in a FOR-loop for all print units.
		
		One FB input parameter need to be defined outside the FOR-loop for each print unit.
		
		Copy the code and adapt the indices and the sensor HW address.

3.	// local error
	HMI_Command.out.b32GlobalError.1 := gasTRCData[1].sStdcIO.out.boApplicationError
	OR gasTRCData[1].sStdcIO.out.boSensorError
	OR gasTRCData[1].sStdcIO.out.boRegisterFBErrorLR
	OR gasTRCData[1].sStdcIO.out.boRegisterFBErrorSR
	OR gasTRCData[1].sStdcIO.out.boLifesignError;
	// global error
	HMI_Command.out.b32GlobalError.0 := HMI_Command.out.b32GlobalError.1
	OR HMI_Command.out.b32GlobalError.2
	OR HMI_Command.out.b32GlobalError.3
	OR HMI_Command.out.b32GlobalError.4
	OR HMI_Command.out.b32GlobalError.5
	OR HMI_Command.out.b32GlobalError.6
	OR HMI_Command.out.b32GlobalError.7
	OR HMI_Command.out.b32GlobalError.8
	OR HMI_Command.out.b32GlobalError.9
	OR HMI_Command.out.b32GlobalError.10
	OR HMI_Command.out.b32GlobalError.11
	OR HMI_Command.out.b32GlobalError.12
	OR HMI_Command.out.b32GlobalError.13
	OR HMI_Command.out.b32GlobalError.14
	OR HMI_Command.out.b32GlobalError.15
	OR HMI_Command.out.b32GlobalError.16
	OR HMI_Command.out.b32GlobalError.17
	OR HMI_Command.out.b32GlobalError.18
	OR HMI_Command.out.b32GlobalError.19
	OR HMI_Command.out.b32GlobalError.20;
		Error handling
		
		There is one global error variable (Word) for the HMI.
		Bit 0 is the general error (TRUE if there is a fault at any of the printing units).
		Bit 1-x are the printing unit specific error bits.
		
		Copy the code and adapt the indices.

pTRC1000Cyclic

Table 3-6 Adaptions in the IPO program

	Adaption	Description

1.	`//----------------------------------------------------------------------------`
	`//FBTRC1000Cyclic`
	`//----------------------------------------------------------------------------`
	`IF TRUE THEN`
	`gaFBTRC1000Cyclic[2]( Axis := RA_PC_02`
	`,CyclicDataToSimotion := TRC2_to_simotion_24bytes`
	`,TRCDAta := gasTRCData[2]`
	`,CyclicDataToSensor => simotion_to_TRC2_24bytes`
	`);`
	
	`//add further TRCs here ...`
		FBTRC1000Cyclic
		
		In this section the function block FBTRC1000Cyclic called.
		
		Copy the code and adapt the indices, the axis name and the I/O variables.

2.	`//----------------------------------------------------------------------------`
	`// TRC internal variable copy used in FOR loop`
	`//----------------------------------------------------------------------------`
	`IF TRUE THEN`
	
	`axPrintCylinderInternal[2] := RA_PC_02;`
	`//axPrintCylinderInternal[3] := RA_PC_03;`
	
	`//add further TRCs here ...`
	
	`{`
	`#ifdef LPrintV300`
	`}`
	`asAxisSTDcIOInternal[2] := gsRA_PC_02AxisSTDcIO;`
	`//asAxisSTDcIOInternal[3] := gsRA_PC_03AxisSTDcIO;`
	
	`asAxisDataInternal[2] := gsRA_PC_02AxisData;`
	`//asAxisDataInternal[3] := gsRA_PC_03AxisData;`
	
	`asAxisTOConfigDataInternal[2] := gsRA_PC_02AxisTOConfigData;`
	`//asAxisTOConfigDataInternal[3] := gsRA_PC_03AxisTOConfigData;`
	
	`// send format to Print Standard STDc`
	`gsRA_PC_02AxisSTDcIO.IN.r64ActualCylinderFormatLength := gasTRCData[2`
	`//gsRA_PC_03AxisSTDcIO.IN.r64ActualCylinderFormatLength := gasTRCData[3`
	
	`//add further TRCs here ...`
	
	`{`
	`#endif`
		Preperations for function block calls FBCharacteristic, FBController, FBDRD and FBTech
		
		The above mentioned function blocks are called in a FOR-loop for all print units.
		
		Some of the FB input parameter need to be defined outside the FOR-loop for each print unit.

3.4 HMI integration

NOTE	Before starting the integration of the HMI all necessary variable sources in SIMOTION should be integrated. Otherwise the reconnection of the variables will not match.

NOTICE	**It is recommended to use the TRC HMI example project as basis!** If there is already a user HMI project existing, copy these user HMI screens into the TRC example HMI project. Because of scripts and recipes which are used inside the TRC HMI project it's easier to keep this project as basis.

In the first step the PrintMarkControl need to be installed. How to do this is described in chapter 3.4.1

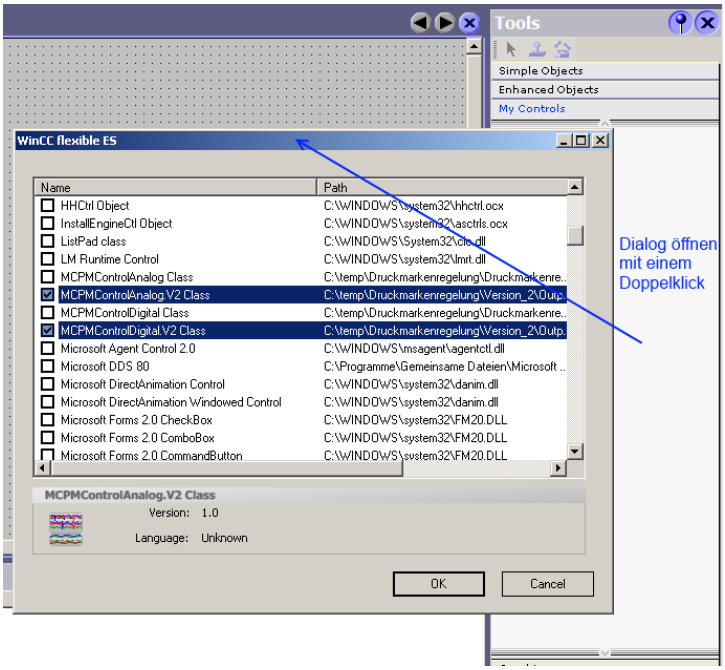
Chapter 3.4.2 describes how to copy the TRC HMI example project into the STEP7 user project.

In chapter 3.4.3 it is described how to set up the OCX PrintMarkControl (oscilloscope) in WinCC flexible.

3.4.1 Install and register PrintMarkControl

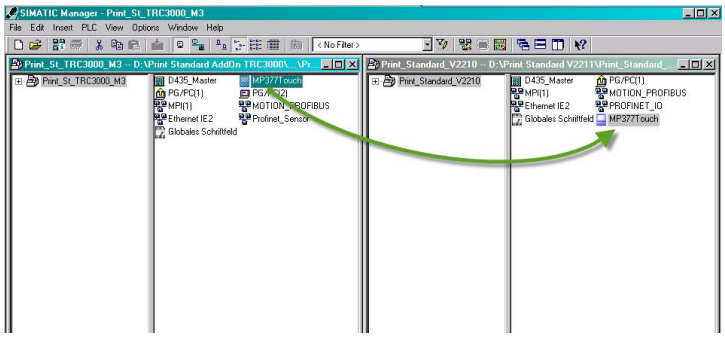
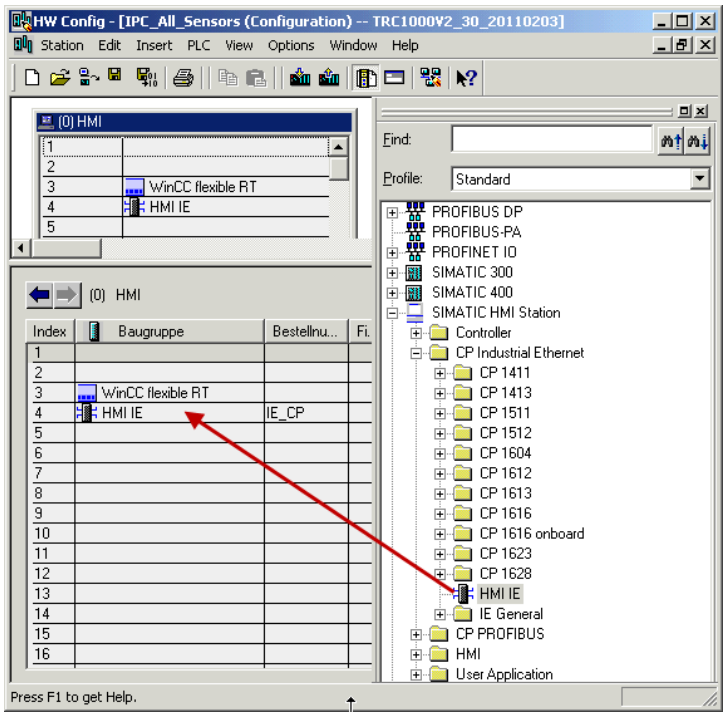
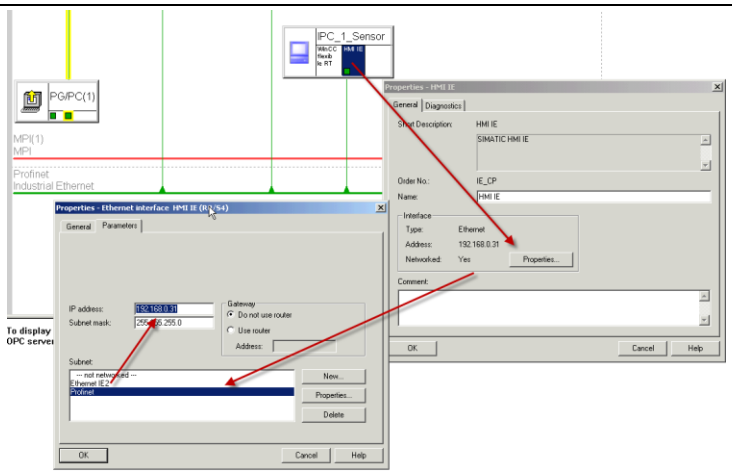
NOTE To update the PrintMarkerControl, install the latest "SetupPrintMarkerControls.exe" file. No further steps are necessary!

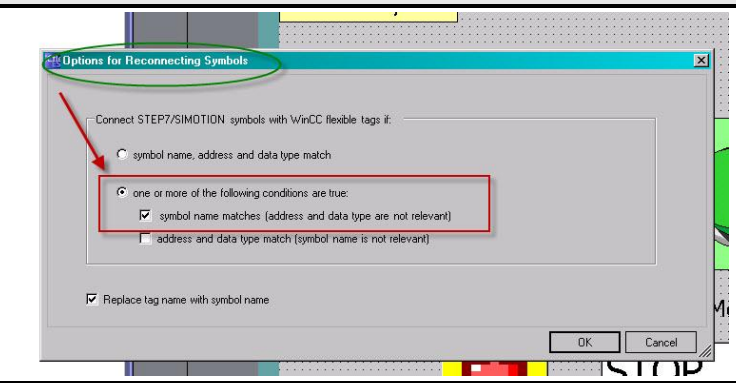
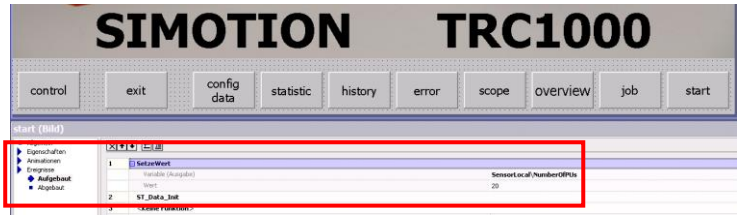
Table 3-7 Installation of PrintMarkerControls

	Description
1.	unzip "SetupPrintMarkerControls.zip" e.g C:\Program Files\Siemens\PrintMarkerControls.
2.	run "SetupPrintMarkerControls.exe"
3.	Open WinCC flexible Open "My Controls" in "Tools"; Select Controls --> right mouse click in white area opens context menu Select "MCPMControlAnalog.V2 Class" and "MCPMControlDigital.V2 Class". 
4.	Now analog and digital PrintMarkControl can be used.

3.4.2 Integration of the HMI project to a STEP7 project

Table 3-8 Integration of HMI project as new HMI project

		Description
1.		Open Step7 with both projects and copy the HMI project from the example project to the user project.
2.		IF there is no “HMI IE” Interface in the HMI-Station available, doubleclick on the HMI-Station (in NetPro) to open the HW-Config. Select the “HMI IE” Interface in the library (SIMATIC HMI Station\CP-Industrial Ethernet\CP 1628) and put it to the rack via drag and drop.
3.		Open net pro of user project. Set IP-address of HMI and connect to the network which is also connected to the SIMOTION.

		Description
4.		Open the HMI project in WinCC flexible and reconnect all symbols by name. If all variables are existent in SIMOTION the reconnection should be successful.
5.		In the “Start” screen at the event “loaded” the variable “NumberOfPUs” need to be set to the maximum number of print units which can be used in this machine.
6.		Safe and compile the project

NOTE If there is already a HMI project existing, these screens can be copied now to the TRC HMI project.

3.4.3 Setting up OCX PrintMarkControl

At the moment the WinCC flexible screen which contains the OCX PrintMarkControl element (oscilloscope field) is getting opened, the OCX PrintMarkControl (TCP/IP client) connects itself to SIMOTION (TCP/IP server).

For this, OCX needs the IP and port address of TCP/IP server (SIMOTION).

It is configured in the WinCC flexible scripts “OCX_InitPictureAnalogControlTRC” and “OCX_InitPictureDigitalControlTRC”.

- The IP address has to match with the SIMOTION IP address
- The default port is 1023 which is also set up in the pTRC1000Backgr program in the section pComServer

The output filed “TCP/IP connection” on the DOAO screens shows the connection state of the OCX PrintMarkControl (Green = connected, Red = not connected).

4 Function description

4.1 Overview

The Print Standard Add-On TRC1000 Application consists of the following parts:

- **LTRC1000** library
- **LRegCtrl** library
- **pTRC1000** program unit
- **S_Var_Gl** global variable source
- **S_H_Var_Gl** global variable source

Content of the libraries:

Table 4-1 Content of the LTRC1000 library

Unit	Content / Description
aVersion	Library changelog
xTypeDef	Global type definitions (constants, enumerators, structures)
fTRC1000	Cyclic and acyclic communication between SIMOTION and TRC - FBTRC1000Backgr (acyclic communication) - FBTRC1000Cyclic (cyclic communication) - Functions used in both FB's
fTRC1000HMI	Data copy sTRCData ←→ HMI_Command Interface sStdcIO structure ←→ HMI - FBHMIDataTransfer - Functions used in FBHMIDataTransfer
fTRC1000TcpIpHMI	TCP/IP communication between SIMOTION and HMI PrintmarkControl - FBTRC1000TcpIpHMI - Functions used in FBTRC1000TcpIpHMI

Table 4-2 Content of the LRegCtrl library

aVersion	Library change log
cProtected	protected constants
cPublic	public onstants
dPublic	global type definitions and variables
fCommunication	UDP Communication function blocks used in TRC5000

fDRD	DRD function blocks
fInsetting	Insetting function blocks
fRegCtrl	Register controller function blocks
fRegExtern	Control external drives
fStatistic	Statistic function blocks

Content of the program unit pTRC1000:

Table 4-3 Content of the program unit pTRC1000

Program	Task	Content / Description
pTRC1000Startup	Startup	Configuration
pTRC1000Cyclic	IPO	- FB calls: - FBTRC1000Cyclic - FBLRegCtrlController (Register controller) - FBTech
pTRC1000Backgr	Background	- FB calls: - FBTRC1000Backgr - FBHMIDataTransfer - TRC config download - Register inching - Error handling

Global variable declarations:

S_Var_GL

Table 4-4 S_Var_GL

Name	Type	Description
NUMBER_OF_ACTIVE_PUS	UINT	Number of register controlled printing units
NUMBER_OF_FIRST_TRC	UINT	Index of first register controlled PU/TRC
gaFBTRC1000Backgr	ARRAY OF FBTRC1000Backgr	FB instance
gaFBTRC1000Cyclic	ARRAY OF FBTRC1000Cyclic	FB instance
gaFBHMIDataTransfer	ARRAY OF FBHMIDataTransfer	FB instance
gaFBTRC1000RegCtrlLR/SR	ARRAY OF FBLRegCtrlController	FB instance
gaFBTRC1000CharacteristicLR	ARRAY OF FBLRegCtrlCharacteristic	optional register characteristic instance
gaFBRegDRD	ARRAY OF FBLRegCtrlDRD	optional DRD function block instance
gaFBTechLR/SR	ARRAY OF FBTech	FB definition
gsTRCConfig	sTypeTRCConfig	Data structure instance

Name	Type	Description
gbopComEnable	BOOL	enable bit FBCom
gasTrcHmiOcxCom	ARRAY OF sTrcHmiOcxCom	transfer data structure to WinCC flexible PrintMarkControl; communication parameters
gsTRCRetainDataSet	sTRCDataRetain	VAR_GLOBAL_RETAIN

S_H_Var_GL

Table 4-5 S_H_Var_GL

Name	Type	Description
HMI_SetOCX	sTRCSetCurveOCX	Data structure for OCX control settings
HMI_Command	sHMI_Command	Data structure for HMI data exchange

Function blocks:

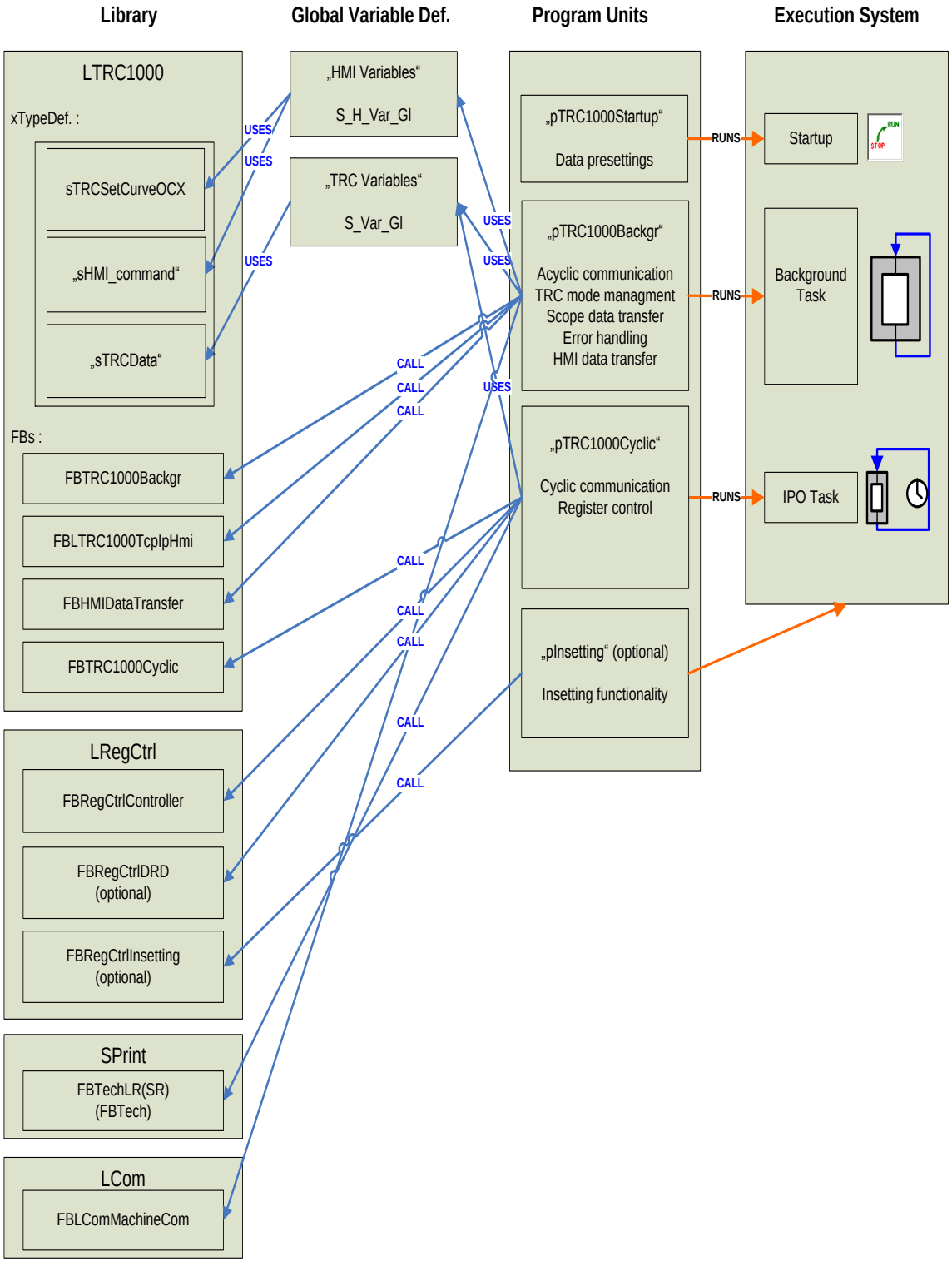
Table 4-6 Function Blocks of LTRC1000 (and LRegCtrl)

Function Block	Unit	Description
FBTRC1000Backgr	fTRC1000	TRC1000 basic functionality/handling - Acyclic communication (TRC parameterization) - Conversion of job setting to TRC structures - Error handling
FBTRC1000Cyclic	fTRC1000	TRC1000 handling of cyclic data. (servo, ipo…) - Transfer of control and status word to cyclic interface - Sign of life monitoring - Register error evaluation
FBTRC1000HMIDataTransfer	fTRC1000HMI	TRC1000 data handling to HMI - Conversion of HMI input data to TRC input structure
FBLTRC1000TcpIpHMI	fTRC1000TcpIpHMI	Function block to send oscilloscope curves to WinCC flexible PrintMarkControl
FBLRegCtrlController	Library LRegCtrl	Register controller function block

NOTE For a more detailed function block description see chapter 4.4!

Figure 4-1 shows a graphical overview about the interaction between the library (FB's and structures), the global variable sources, the program parts and the execution system.

Figure 4-1 Graphical overview about application part interaction



4.2 Constants

Generally the basic application is able to catch the requirements of a complete machine without big changes. However a few adaptions for optimization are useful by adapting the application to the needs of the respective printing machine project.

The basic application is prepared for a range of 20 printing units. It is not recommendable to change the length of the arrays for the HMI connection. Otherwise the whole HMI application needs to be changed!

An adaption of the SIMOTION internal variables to the real machine range is possible. The variables are preset for the example project. To adapt the application to a real machine project only the variable NUMBER_OF_ACTIVE_PUS (S_Var_GL) need to be changed.

Table 4-7 Constants in the library LTRC1000, xTypeDef

Name	Type	Description	Preset
KONST_INT16	LREAL	Value INT16 **Do not change!**	32767.0
KONST_USINT8	REAL	Value USINT8 **Do not change!**	256.0
SCOPE_DATA_LENGTH_DO	INT	Array length of digital scope edges from TRC **Do not change!**	399
SCOPE_DATA_LENGTH_AO	INT	Array length of analog scope from TRC **Do not change!**	799
HMI_CURVE_LENGTH_DO	INT	Array length of digital scope for HMI **Do not change!**	719
HMI_CURVE_LENGTH_AO	INT	Array length of analog scope for HMI **Do not change!**	799
NUMBER_OF_PUS	UINT	Number of print units for HMI application. **Do not change!**	20
ERROR_HISTORY_LENGTH	UINT	Number of errors which can be stored in the error history. If limit is reached, circular buffering.	7

Table 4-8 Constants S_Var_Gl

Name	Type	Description	Preset
NUMBER_OF_ACTIVE_PUS	UINT	Number of register controlled printing units	1
NUMBER_OF_FIRST_TRC	UINT	Index of first register controlled PU/TRC	2

4.3 Data structures

The TRC1000 data can be divided in the following main structures:

- sTRCData
- sTRCRetainDataSet
- sHMI_Command
- sTRCHmiOcxCom
- sTRCConfig

Every FB of the TRC application uses the sTRCData structure as In/Output-structure and is able to read and write to the values of the structure.

Additionally to the basic structure of the single TRC handling a HMI structure for the data transfer to the HMI-screens is necessary. The structure collects all global data from the TRC application and works as an array based interface.

Some of the TRCData are saved in a retain data structure additionally.

The sTRCConfig data are used for TRC configuration. It will be preset by the function “FCTRC1000DefValPresetting” called in the SIMOTION StartUp task. It contains the parameter values and type descriptions of the TRC1000 and will be needed for the acyclic data transfer

4.3.1 sTRCData

The structure sTRCData contains all the data values of a single TRC. To provide a clear and useable structure the main structure is divided into the following parts:

- sStdcIO (TRC interface)
- sData_cyclic (cyclic communication data)
- sData_acyclic (acyclic communication data)
- sHMIControlData (analog/digital curve, gate curve, gate start/end positions)

sTRCData.sStdcIO.IN

Table 4-9 sTRCData.sStdcIO.IN (TRC Input data)

Name	Type	Description
u16PrintUnitNr	UINT	Number of assigned printing unit
boSensorActive	BOOL	Print unit / TRC active
boUseCommandPosition	BOOL	TRUE: The axis command position will be used as reference position FALSE: The axis actual position will be used as reference position
r32SensorDelayCompTime	REAL	[s] Dead time compensation
boReadSensor1DOAO	BOOL	Read analog and digital oscilloscope data sensor 1
boReadSensor2DOAO	BOOL	Read analog and digital oscilloscope data sensor 2

Name	Type	Description
boActivateLifesignMonitoring	BOOL	Activation of life sign monitoring Pre-assigned TRUE in xTypeDef
i16LifesignTolerence	INT	Tolerance of life sing monitoring Recommended value: 0
boReadTRCStatus	BOOL	Start read TRC Status (system info, actual parameter, etc.)
boReadTRCConfig	BOOL	Start read TRC Config (configuration parameter)
boWriteTRCParameter	BOOL	Start write TRC parameter (job download)
boReadTRCError	BOOL	Start read TRC errorID's
boSetSensor1TriggerValue	BOOL	Set new trigger values (sensor 1)
boSetSensor2TriggerValue	BOOL	Set new trigger values (sensor 2)
sSensorTriggerValues	sSensorTriggerValuesActType	Data structure: TRC trigger values
r32SetSensor1Offset_Ref	REAL	Offset reference mark (sensor 1)
r32SetSensor1ThresholdLevel_Ref	REAL	Threshold level (trigger level) reference mark (sensor 1)
r32SetSensor1ThresholdLevel2_Reg	REAL	Threshold level (trigger level) printing mark (sensor 1)
r32SetSensor2Offset_Ref	REAL	Offset reference mark (sensor 2)
r32SetSensor2ThresholdLevel_Ref	REAL	Threshold level (trigger level) reference mark (sensor 2)
r32SetSensor2ThresholdLevel2_Reg	REAL	Threshold level (trigger level) printing mark (sensor 2)
boStartTriggerValueTeaching	BOOL	Start trigger value teaching (travel measurement)
boChangeCommandDistance_WebWeb	BOOL	Register fine adjustment: Change distance between reference and printing mark gate (WebWeb, WebWeb2)
boChangeGateWidth	BOOL	Change gate width to r32GateWidth
r32GateWidth	REAL	gate width set value
boAGSStart	BOOL	Start AGS
boChangeGatePosition	BOOL	Change gate position to r32GatePosition
r32GatePosition	REAL	gate position set value
boCenterGate	BOOL	Start center gate function
boSensorReset	BOOL	Start sensor reset
eResetMode	eSensorResetENUM	Sensor_reset: reboot TRC device Factory_reste: reset factory settings
boErrorReset	BOOL	Fault acknowledge
sTRCCalibration	sTRCCalibrationInData	Data structure: TRC calibration
boStartTRCCalibration	BOOL	Start sensor calibration
boNextStep	BOOL	Go to next calibration step
i16TRCNumber	INT	Select sensor number to calibrate

Name	Type	Description
i16TRCHead	INT	Select head1 / head2 to calibrate
sFiberOpticLength	sFiberOpticLengthInData	Data structure: fiber optic length
i16TRCNumber	INT	Select sensor number to change fiber optic length
i16SetLengthHead1	INT	Select fiber optic length head1 [25] 2.5 m [50] 0.0 m
i16SetLengthHead2	INT	Select fiber optic length head2 [25] 2.5 m [50] 0.0 m
boSetLength	BOOL	Set selected length
boReadLength	BOOL	Read actual fiber optic length
boRegCtrlEnableLR	BOOL	Activate register controller (length register)
boFBRegCtrlResetLR	BOOL	Fault acknowledge FB register controller (length register)
boRegCtrlSpdPreCtrlLR	BOOL	Enable velocity precontrol (length register) r32SpeedPreControl will added after the control loop
boRegCtrl_I_channel_WebWebLR	BOOL	Activate I-channel (WebWeb) (length register)
r32RegCtrlVPosMaxLR	REAL	Limit positive register correction velocity [mm/s]
r32RegCtrlVNegMaxLR	REAL	Limit negative register correction velocity [mm/s]
r32RegCtrl_Kp_WebWebLR	REAL	Controller gain Kp mode WebWeb (length register controller)
r32RegCtrl_Kp_WebCylLR	REAL	Controller gain Kp mode WebCylinder (length register controller)
r32RegCtrl_I_WebWebLR	REAL	Controller integral time mode WebWeb (length register controller)
r32RegCtrl_I_WebCylLR	REAL	Controller integral time mode WebCylinder (length register controller)
r32RegCtrlVInchingLR	REAL	Inching step width [mm]
r32RegCtrlSetValueLR	REAL	Register controller set value (length register)
boRegCtrlEnableSR	BOOL	Activate register controller (side register)
boFBRegCtrlResetSR	BOOL	Fault acknowledge FB register controller (side register)
boRegCtrlSpdPreCtrlSR	BOOL	Enable velocity precontrol (side register) r32SpeedPreControl will added after the control loop
boRegCtrl_I_channel_WebWebSR	BOOL	Activate I-channel (WebWeb) (side register)
r32RegCtrlVPosMaxSR	REAL	Limit positive register correction velocity [mm/s]
r32RegCtrlVNegMaxSR	REAL	Limit negative register correction velocity [mm/s]
r32RegCtrl_Kp_WebWebSR	REAL	Controller gain Kp mode WebWeb (side register controller)
r32RegCtrl_Kp_WebCylSR	REAL	Controller gain Kp mode WebCylinder (side register controller)
r32RegCtrl_I_WebWebSR	REAL	Controller integral time mode WebWeb (side

Name	Type	Description
		register controller)
r32RegCtrl_I_WebCylSR	REAL	Controller integral time mode WebCylinder (side register controller)
r32RegCtrlVInchingSR	REAL	Inching step width [mm]
r32RegCtrlSetValueSR	REAL	Register controller set value (length register)
sRegCtrlPU	structPrintingUnit	Data structure used for register controller FB Structure of the library LRegCtrl **(Only the used structure variables are described here!)**
technology	enumTechnology	Selection of print technolgy. Generally it will distinguished between printing technologies where register movements influencing the material (fix Nip) e.g. gravure and printing technologies where register movements have less influence on the web e.g. flexo.
controlMode	enumOperatingMode	WebCylinder WebWeb WebWeb2
printFormat	REAL	Format length
v_setpoint	REAL	Axis command velocity
boRegInchNegativeLR	BOOL	Register inching direction (length register) FALSE: positive direction TRUE: negative direction
r32InchDistMMLR	REAL	Register inching value (length register) Interconnected with print standard stdcIO interface
boRegInchNegativeSR	BOOL	Register inching direction (side register) FALSE: positive direction TRUE: negative direction
r32InchDistMMSR	REAL	Register inching value (side register) Interconnected with print standard stdcIO interface
sInsettingIn	sInsettingType	Insetting sub-structure (see Insetting Add-On documentation)
sDRDIn	sLRegCtrlDRDIn	DRD sub-structure (dynamic register decoupling) (see DRD Add-On documentation)
boEnableIntActionLR	BOOL	Enable/Disable integral part of register control modes with integral part (length register)
boEnableIntActionSR	BOOL	Enable/Disable integral part of register control modes with integral part (side register)
u8FilterDepth	USINT	Averaging over the parameterized number of register error values

sTRCData.sStdcIO.OUT

Table 4-10 sTRCData.sStdcIO.OUT (TRC Output data)

Name	Type	Description
boLifesignError	BOOL	Communication SIMOTION – TRC interrupted
boApplicationError	BOOL	Application internal error
u16ApplicationErrorID	UINT	Application error number
boSensorError	BOOL	TRC internal error
u16SensorErrorID	UINT	TRC error number (converted into own error number)
boRegisterFBErrorLR	BOOL	Register controller function block error (length register)
boRegisterFBErrorSR	BOOL	Register controller function block error (side register)
u16RegisterFBErrorIDSR	UINT	Register controller function block error number (length register)
u16RegisterFBErrorIDLR	UINT	Register controller function block error number (side register)
i32Parameter1	DINT	Additional parameter 1 application error number
i32Parameter2	DINT	Additional parameter 2 application error number
sErrorHistory	sErrorHistory Type	TRC error histoy
au16ErrorNumber_P947	ARRAY [0..7] OF UINT	TRC error number (converted into own error number)
adtTimeStamp	ARRAY [0..7] OF DT	SIMOTION time stamp
boDataReadStatusAO	BOOL	Read analog oscilloscope data from TRC active
boDataReadStatusDO	BOOL	Read digital oscilloscope data from TRC active
boLDPV1_Busy	BOOL	LDPV1 channel in use (data transfer active)
b8TRCUpdateActive	BYTE	Status TRC job download: 1: data download SIMOTION → TRC 3: data download panel → SIMOTION 4: download successfully 5: error
b8ReadTRCConfig	BYTE	Status read TRC config (configuration parameter)
b8ReadTRCStatus	BYTE	Status read TRC status (system info, actual parameter, etc.)
b8ReadTRCError	BYTE	Status read TRC error from TRC device *
b8TRCReset	BYTE	Status TRC reset *
sTRCCalibration	sTRCCalibrationOutData	Data structure: TRC calibration
boTRCCalibrationActive	BOOL	TRC calibration is being processed
b8TRCCalibrationStatus	BYTE	TRC calibration status: 0: idle 1: use calibration gauge input white 2: use calibration gauge input black

Name	Type	Description
		3: use calibration gauge input white (again) 4: calibration successfully 5: error
sFiberOpticLength	sFiberOpticLengthOutData	Data structure: fiber optic length
i16LengthHead1	INT	Actual fiber optic length sensor 1 25: 2.5m 50: 5.0m
i16LengthHead2	INT	Actual fiber optic length sensor 2 25: 2.5m 50: 5.0m
b8FiberOpticLengthStatus	BYTE	Status set fiber optic length *
b8ChangeGatePos	BYTE	Status gate position change *
b8ChangeGateWidth	BYTE	Status gate width change *
b8CentreGate	BYTE	Status gate center *
b8AGSStatus	BYTE	Status AGS (auto gate setting) *
b8MarkfieldUpdateActive	BYTE	Status markfield update (function to change TRC parameter) *
b8SetTriggerSensor1	BYTE	Status set new trigger values (sensor 1) *
b8SetTriggerSensor2	BYTE	Status set new trigger values (sensor 2) *
sSensorTriggerValues	sSensorTriggerValuesActType	Data structure: TRC trigger values
r32ActSensor1Offset_Ref	REAL	Offset reference mark (sensor 1)
r32ActSensor1Offset2_Reg	REAL	Offset printing mark (sensor 1)
r32ActSensor1ThresholdLevel_Ref	REAL	Threshold level (trigger level) reference mark (sensor 1)
r32ActSensor1ThresholdLevel2_Reg	REAL	Threshold level (trigger level) printing mark (sensor 1)
r32ActSensor2Offset_Ref	REAL	Offset reference mark (sensor 2)
r32ActSensor2Offset2_Reg	REAL	Offset printing mark (sensor 2)
r32ActSensor2ThresholdLevel_Ref	REAL	Threshold level (trigger level) reference mark (sensor 2)
r32ActSensor2ThresholdLevel2_Reg	REAL	Threshold level (trigger level) printing mark (sensor 2)
b8TRCTeaching	BYTE	Status trigger value teaching (travel measurement) *
r32RegCtrlRegErrorLR	REAL	Actual register error (length register)
r32RegCtrlRegErrorSR	REAL	Actual register error (side register)
boRegCtrlAlarmMarkPos	BOOL	**Bit "Alarm PM Position" (StatusWord2):** FALSE = GREEN: valid PM position and a valid number of active edges within the gate. TRUE = RED: no (valid) PM detected within the gate for the last three printing cylinder

Name	Type	Description
		revolutions (PM out of gate, no PM within the gate).
boRegCtrlAlarmMarkWidth	BOOL	**Bit "Alarm PM Width" (StatusWord2):** FALSE = GREEN: valid PM width within the gate and "Alarm PM Position" is FALSE. TRUE = RED: no valid PM width for three printing cylinder revolutions (width exceeds maximum/minimum width limit); "Alarm PM Position" = TRUE
boRegCtrlActiveLR	BOOL	Register controller active (length register)
boRegCtrlActiveSR	BOOL	Register controller active (side register)
r32IpoCycleTime	REAL	Task cycle time (used for register controller FB)
sInsettingOut	sInsettingOutType	Insetting sub-structure (see Insetting Add-On documentation)
sDRDOut	sLRegCtrlDRDOut	DRD sub-structure (dynamic register decoupling) (see DRD Add-On documentation)
boSetRegActive	BOOL	Indicates an active SetReg print cylinder adjustment. Signal can be used to open and close the gap in rotogravure printing machines before starting the cylinder adjustment.
boSetRefActive	BOOL	Indicates an active SetRef print cylinder adjustment. Signal can be used to open and close the gap in rotogravure printing machines before starting the cylinder adjustment.

* **Meaning of the status variables (byte):**

0: idle

1: functionality activated

4: functionality successfully completed

5: error

sData_cyclic

Table 4-11 sTRCData.sData_cyclic (cyclic communication data)

Name	Type	Description
Simotion_to_Sensor		
sControlWord_1		
boFaultAcknowledge	BOOL	Fault acknowledge
boControlbyPLC	BOOL	TRUE: connection is fully established and valid setpoint values are transmitted from the controller to the sensor.
boSensor1DOReading	BOOL	Handshake bit to achieve data consistency during oscilloscope data reading. Bit TRUE during reading dataset.
boSensor2DOReading	BOOL	Handshake bit to achieve data consistency during oscilloscope data reading. Bit TRUE during reading dataset.
boSensor1AOReading	BOOL	Handshake bit to achieve data consistency during oscilloscope data reading. Bit TRUE during reading dataset.
boSensor2AOReading	BOOL	Handshake bit to achieve data consistency during oscilloscope data reading. Bit TRUE during reading dataset.
sControlWord_2		
boSensorInit	BOOL	Initialize printing mark detection
boManualGate	BOOL	Activate manual gate setting
boGateActivate	BOOL	Enable printing mark detection (only if bit boManualGate = TRUE)
boAGSStart	BOOL	Start AGS (automatic gate setting)
boBacksideActivate	BOOL	Switch between integrated printing mark sensor (IDS, head 1) and the external printing mark sensor (DS, head 2) FALSE: head 1 TRUE: head 2
boLengthRegActive	BOOL	Register controller (length register) active Currently this bit isn't connected in the application and in the sensor device!
boSideRegActive	BOOL	Register controller (side register) active Currently this bit isn't connected in the application and in the sensor device!
boSideRegManualSetting	BOOL	TRUE: disable detection of printing mark width TRUE -> FALSE: initialize/start detection of printing mark width Can be used during manual correction of the side register by moving the printing cylinder in axial direction.
i16SignOfLifeCounter	INT	Life sign counter (I/O communication)
r32AxisPosition	REAL	Printing cylinder position [mm]
r32AxisSpeed	REAL	Printing cylinder speed [mm/s]

Name	Type	Description
r32AxisAcceleration	REAL	Printing cylinder acceleration [mm/s²]
r32OffsetLengthRegister	REAL	Correction length register [mm]
r32OffsetSideRegister	REAL	Correction side register [mm]
Sensor_to_Simotion		
sStatusWord_1		
boSwitchOnReady	BOOL	Initialized, ready to switch on
boOperateReady	BOOL	Ready for operation
boFault	BOOL	Fault present
boControlRequested	BOOL	Control requested / no control requested
boOscilDataReady	BOOL	New oscilloscope data available
sStatusWord_2		
boTriggerValueTeaching	BOOL	TRC teaching (travel measurement) finished
boAlarmMarkPosition	BOOL	FALSE: valid PM position and a valid number of active edges within the gate. TRUE: no (valid) PM detected within the gate for the last three printing cylinder revolutions (PM out of gate, no PM within the gate).
boAlarmMarkWidth	BOOL	FALSE: valid PM width within the gate and “Alarm PM Position” is FALSE. TRUE: no valid PM width for three printing cylinder revolutions (width exceeds maximum/minimum width limit); “Alarm PM Position” = TRUE
boGateActive	BOOL	FALSE: At least three valid PM positions are detected after a gate setting. TRUE: Shift Gate Position, Set Gate Position.
boAGSActive	BOOL	FALSE: AGS block mark detected, no AGS block mark detected after three print cylinder revolutions, maximum/minimum speed limit exceeded or stand still of the printing cylinder during AGS active TRUE: AGS active
i16MarkCounter	INT	Printing mark counter
i16SignOfLifeCounter	INT	Sign of life counter (I/O communication)
r32RegisterError_Length	REAL	Actual register error (length register) [mm]
r32Registererror_Side	REAL	Actual register error (side register) [mm]
r32ActualPosition_Refmark	REAL	Position reference mark [mm]
r32ActualWidth_Refmark	REAL	Width reference mark [mm]
i32EncoderPosition	DINT	Signal of the encoder input of the IDS (see interface)

NOTE Generally the cyclic data (to and from TRC) are only user information. All commands and necessary responses will be given from the application to the variables.

sData_acyclic

Table 4-12 sTRCData.sData_acyclic (acyclic communication data)

Name	Type	Description
Simotion_to_Sensor		
eSensor_P1_Command	eCommandType	TRC P1 command (for more information see Wiedeg user manual)
sSensor_Set_Parameter		
sSensorConfig_P2	sP2Config	Data structure: sensor config P2
boMeasuringEdge	BOOL	Measuring edge selection FALSE: rising (front) edge TRUE: falling (back) edge
boSensorChoice	BOOL	sensor selection FALSE: head 1 (front side) TRUE: head 2 (back side)
eMarkConfig	eMarkType	Block mark Wedge mark Double wedge mark
boAutomatic	BOOL	Sensor automatic mode FALSE: manual (trigger and offset values need to be set manually) TRUE: automatic (sensor teaching is used once a mark is inside the gate)
boLimitRegError	BOOL	Limited register error FALSE: inactive TRUE: active (register deviation is limited to the actual gate width)
boMackleDetection	BOOL	Mackle detection FALSE: inactive TRUE: active (alarm, if limit of permitted active edges in the gate is overrun)
eControlMode	eSensorMode	WebCylinder, WebWeb, WebWeb2
boSensor1Amplification	BOOL	TRUE: Sensor 1 signal gain (factor 2) active
boSensor2Amplification	BOOL	TRUE: Sensor 2 signal gain (factor 2) active
boDOAOWindowReference	BOOL	AO/DO reference value FALSE: Oscilloscope refers absolute to axis position (0…360°) TRUE: Oscilloscope refers to gate middle position (P2014)
i16SensorCycleTime_P3	INT	AO/DO detection refresh cycle
r32CylinderCircumference_P10	REAL	Printing cylinder circumference
r32SetValueWebWeb_P11	REAL	Printing mark distance set point (WebWeb, WebWeb2)
r32GateWidth_P12	REAL	Gate width
r32GateOffset_P13	REAL	Gate shift offset
r32GatePosition_P14	REAL	Gate position
r32OffsetLengthregister_P15	REAL	Correction offset length register
r32OffsetSideregister_P16	REAL	Correction offset side register

Name	Type	Description
sSensor1TriggerVaule	sTriggerValue	Data structure: sensor 1 trigger values
r32ThresholdLevel_RefMark	REAL	Threshold level (trigger level) reference mark (sensor 1)
r32ThresholdLevel_RegMark	REAL	Threshold level (trigger level) printing mark (sensor 1)
r32Offset	REAL	Offset reference mark (sensor 1)
r32SpeedCompensationValue	REAL	TRC dead time compensation (not used)
sSensor2TriggerVaule	sTriggerValue	Data structure: sensor 2 trigger values
r32ThresholdLevel_RefMark	REAL	Threshold level (trigger level) reference mark (sensor 2)
r32ThresholdLevel_RegMark	REAL	Threshold level (trigger level) printing mark (sensor 2)
r32Offset	REAL	Offset reference mark (sensor 2)
sMarkGeometry	sMarkDefine	Data structure: mark geometry
r32Width_Blockmark	REAL	[mm] Width block mark
r32Width_WedgeMark_Min	REAL	[mm] Width (short side) wedge mark (0 is possible)
r32Width_WedgeMark_Max	REAL	[mm] Width (long side) wedge mark
r32EdgeLength_WedgeMark	REAL	[mm] Edge length (height) wedge mark
r32Width_DoubleWedgeMark_Min	REAL	[mm] Width (short side) double wedge mark (0 is possible)
r32Width_DoubleWedgeMark_Max	REAL	[mm] Width (song side) double wedge mark
r32MiddleWidth_DoubleWedgeMark	REAL	[mm] Width (middle) double wedge mark
r32Width_DoubleWedgeMark	REAL	[mm] Width (total) double wedge mark
r32EdgeLength_DoubleWedgemark	REAL	[mm] Edge length (height) double wedge mark
r32Width_DoubleBlockMark_Min	REAL	[mm] Width (total, short side) double block mark
r32Width_DoubleBlockMark_Max	REAL	[mm] Width (total, long side) double block mark
r32WidthStraight_DoubleBlockmark	REAL	[mm] Width straight mark of double block mark
r32WidthBevel_DoubleBlockMark	REAL	[mm] Width oblique mark of double block mark
r32EdgeLength_DoubleBlockmark	REAL	[mm] Edge length (height) double block mark
r32Distance_AGSMarktoRefMark	REAL	[mm] Distance AGS block mark to reference mark
r32Tolerence_AGSMark	REAL	[mm] Tolernace range of AGS mark
r32Width_AGSMark_1	REAL	[mm] Width block mark (first of three blocks)
r32Width_AGSMark_2	REAL	[mm] Width block mark (second of three blocks)
r32Width_AGSMark_3	REAL	[mm] Width block mark (third of three blocks)
r32Chasm_AGSMark_1	REAL	[mm] Gap between first and second block mark block
r32Chasm_AGSMark_2	REAL	[mm] Gap between second and third block mark block
r32TriggerVauleMIN_P45	REAL	TRC teaching (travel measurement) min. trigger voltage

Name	Type	Description
r32DeviationValueMax_P46	REAL	TRC teaching (travel measurement) max.web voltage deviation
i16SignofLifeTolerence_P925	INT	Number of tolerable sign of life interruptions
End sSensor_Set_Parameter		
sSensor_Change_Parameter		
sSensor1TriggerValue	'sTriggerValue'	Data structure: sensor 1 trigger values
r32ThresholdLevel_RefMark	REAL	Threshold level (trigger level) reference mark (sensor 1)
r32ThresholdLevel_RegMark	REAL	Threshold level (trigger level) printing mark (sensor 1)
r32Offset	REAL	Offset reference mark (sensor 1)
sSensor2TriggerValue	'sTriggerValue'	Data structure: sensor 2 trigger values
r32ThresholdLevel_RefMark	REAL	Threshold level (trigger level) reference mark (sensor 2)
r32ThresholdLevel_RegMark	REAL	Threshold level (trigger level) printing mark (sensor 2)
r32Offset	REAL	Offset reference mark (sensor 2)
r32CommandDistance_WebWeb	REAL	[mm] Set distance print marks (web web, web web 2)
r32GateWidth	REAL	Gate width
r32GatePosition	REAL	Gate position
End sSensor_Change_Parameter		
r32Sensor1_POS_Offset	REAL	[mm] Web length between printing unit and sensor head
r32Sensor2_POS_Offset	REAL	[mm] Web length between printing unit and sensor head
Sensor_to_Simotion		
sActualSensorConfig		
sSensorConfig_P200	sP2Config	Data structure: sensor config P2
boMeasuringEdge	BOOL	Measuring edge selection FALSE: rising (front) edge TRUE: falling (back) edge
boSensorChoice	BOOL	sensor selection FALSE: head 1 (front side) TRUE: head 2 (back side)
eMarkConfig	eMarkType	Block mark Wedge mark Double wedge mark
boAutomatic	BOOL	Sensor automatic mode FALSE: manual (trigger and offset values need to be set manually) TRUE: automatic (sensor teaching is used once a mark is inside the gate)
boLimitRegError	BOOL	Limited register error

Name	Type	Description
		FALSE: inactive TRUE: active (register deviation is limited to the actual gate width)
boMackleDetection	BOOL	Mackle detection FALSE: inactive TRUE: active (alarm, if limit of permitted active edges in the gate is overrun)
eControlMode	eSensorMode	WebCylinder, WebWeb, WebWeb2
boSensor1Amplification	BOOL	TRUE: head 1 signal gain (factor 2) active
boSensor2Amplification	BOOL	TRUE: head 2 signal gain (factor 2) active
boDOAOWindowReference	BOOL	AO/DO reference value FALSE: Oscilloscope refers absolute to axis position (0…360°) TRUE: Oscilloscope refers to gate middle position (P2014)
i16SensorCycleTime_P406	INT	AO/DO detection refresh cycle
r32CylinderCircumference_P203	REAL	Printing cylinder circumference
r32SetValueWebWeb_P208	REAL	Printing mark distance set point (WebWeb, WebWeb2)
r32GateWidth_P222	REAL	Gate width
r32GateOffset_P295	REAL	Gate shift offset
r32GatePosition_P215	REAL	Gate position
r32OffsetLengthregister_P276	REAL	Correction offset length register
r32OffsetSideregister_P234	REAL	Correction offset side register
sSensor1TriggerVaule	sTriggerValue	Data structure: sensor 1 trigger values
r32ThresholdLevel_RefMark	REAL	Threshold level (trigger level) reference mark (sensor 1)
r32ThresholdLevel_RegMark	REAL	Threshold level (trigger level) printing mark (sensor 1)
r32Offset	REAL	Offset reference mark (sensor 1)
r32SpeedCompensation_P287	REAL	TRC dead time compensation (not used)
sSensor2TriggerVaule	sTriggerValue	Data structure: sensor 2 trigger values
r32ThresholdLevel_RefMark	REAL	Threshold level (trigger level) reference mark (sensor 2)
r32ThresholdLevel_RegMark	REAL	Threshold level (trigger level) printing mark (sensor 2)
r32Offset	REAL	Offset reference mark (sensor 2)
sMarkGeometry	sActualMarkDefine	Data structure: mark geometry
r32Width_Blockmark_P800	REAL	Width block mark
r32Width_Min_WedgeMark_P801	REAL	Width (short side) wedge mark (0 is possible)
r32Width_Max_WedgeMark_P802	REAL	Width (long side) wedge mark
r32EdgeLength_WedgeMark_P803	REAL	Edge length (height) wedge mark

Name	Type	Description
r32Width_Min_DoubleWedgeMark_P804	REAL	Width (short side) double wedge mark (0 is possible)
r32Width_Max_DoubleWedgeMark_P805	REAL	Width (song side) double wedge mark
r32MiddleWidth_DoubleWedgeMark_P806	REAL	Width (middle) double wedge mark
r32Width_DoubleWedgeMark_P807	REAL	Width (total) double wedge mark
r32EdgeLength_DoubleWedgemark_P808	REAL	Edge length (height) double wedge mark
r32Width_Min_DoubleBlockMark_P809	REAL	Width (total, short side) double block mark
r32Width_Max_DoubleBlockMark_P810	REAL	Width (total, long side) double block mark
r32WidthStraight_DoubleBlockmark_P811	REAL	Width straight mark of double block mark
r32WidthBevel_DoubleBlockMark_P812	REAL	Width oblique mark of double block mark
r32EdgeLength_DoubleBlockmark_P813	REAL	Edge length (height) double block mark
r32Distance_AGSMarktoRefMark_P288	REAL	Distance AGS block mark to reference mark
r32Tolerence_AGSMark_P289	REAL	Tolernace range of AGS mark
r32Width_AGSMark_1_P290	REAL	Width block mark (first of three blocks)
r32Width_AGSMark_2_P291	REAL	Width block mark (second of three blocks)
r32Width_AGSMark_3_P292	REAL	Width block mark (third of three blocks)
r32Chasm_AGSMark_1_P293	REAL	Gap between first and second block mark block
r32Chasm_AGSMark_2_P294	REAL	Gap between second and third block mark block
r32TriggerValueMIN_P814	REAL	TRC teaching (travel measurement) min. trigger voltage
r32DeviationValueMax_P815	REAL	TRC teaching (travel measurement) max.web voltage deviation
i16SignofLifeTolerence_P925	INT	Number of tolerable sign of life interruptions
End sActualSensorConfig		
sActualSensorStatus		
sSystemInfo	sSystemInfo	Data structure: TRC internal info
u32SystemStatus_P102	UDINT	TRC system state
i32SensorCyclic_P103	DINT	TRC cycle
i32ProfinetCyclic_P104	DINT	PROFINET data cycle
u32StatusFlowControl_P121	UDINT	Status flow control
sPowerONTimer_P122	P122_Struct	Operating hours counter
sgSystemID_P919	STRING	System ID
u16TelegramType_P922	UINT	Telegram type
u16TolerenceSignofLife_P925	UINT	Tolerable sign of life interruptions
u16ErrorCounter_P944	UINT	TRC error counter
au16ErrorNumber_P947	ARRAY [0..7] OF UINT	TRC internal error number
asErrorTimeStamp_P948	ARRAY [0..7] OF P122_Struct	Error time stamp
sgDriveIdentificationNr_P965	STRING	Profile identification number

Name	Type	Description
sDriveObjectIdentification_P975	P975_Struct	Data structure: drive object identification
ProducerCode	UINT	Manufacturer code
Typ_DriveUnit	UINT	Type drive object
FirmwareVersion	UINT	Firmware version
FirmwareDatumY	UINT	Firmware date (year)
FirmwareDatumMD	UINT	Firmware date (day, month)
DOTypeClass	UINT	Drive Object Type class
DOSubClass	UINT	Drive Object Sub class
sPDDiagClockSync_P2064	P2064_Struct	Data structure: diagnostics clock synchronous mode
ClockSynchModeActive	DINT	Clock synchronous mode activated
BuscyclicTime	DINT	Bus cycle time
MasterCyclicTime	DINT	Master cycle time
Ti_Time	DINT	Instant of value acquisition
To_Time	DINT	Instant of setpoint acquisition
Tdx_Time	DINT	Data exchange interval
PLL_WindowTime	DINT	PLL window
PLL_DelayTime	DINT	PLL delay time
sgSensorName_P61000	STRING[20]	TRC name
sSensorIP_P61001	P61001_Struct	TRC IP address
sActualParaValue	sRParaValue	Data structure: actual TRC parameter
u32MeasuringStatus_P201	UDINT	*Wiedeg internal parameter*
u32MeasuringStatus_P202	UDINT	*Wiedeg internal parameter*
r32MarkActualValue_P205	REAL	Printing mark actual position (WebCylinder)
r32RefMarkActualPos_P206	REAL	Reference mark actual position (WebWeb, WebWeb2)
r32RegMarkActualPos_P207	REAL	Printing mark actual position (WebWeb, WebWeb2)
r32ActualRefRegMarkDistance_p209	REAL	Actual distance reference mark – printing mark (WebWeb, WebWeb2)
r32RegisterErrorLength_P210	REAL	Printing mark Difference set value – act value
r32CommandPosition_P214	REAL	Display setpoint position
r32CommandPosition_P215	REAL	Gate positions at gate setting (WebCylinder: reference mark, WebWeb/WebWeb2: printing mark)
r32CommandPositionRefMark_P216	REAL	Gate position reference mark at gage setting (WebWeb(WebWeb2)
i16MarkCounter_P217	INT	Printing mark counter
r32MarkWidthMax_P223	REAL	PM width maximum value (depends on P2, PM type)
r32MarkWidthMin_p224	REAL	PM width minimum value (depends on P2, PM type)

Name	Type	Description
r32MarkEdgeLength_P225	REAL	PM edge length (depends on P2, PM type)
r32ConversionFactor_P226	REAL	Slope factor Δb <-> Δs
u32StatusSideRegister_P227	UDINT	State side regsiter
r32MarkCommandWidthWebcylidner_P228	REAL	PM width setpoint (WebCylinder)
r32MarkActualWidthWebCylinder_P229	REAL	PM width actual value (WebCylinder)
r32RefMarkActualWidthWebWeb_P230	REAL	Reference mark actual value (WebWeb, WebWeb2)
r32RegMarkActualWidthWebWeb_P231	REAL	Printing mark actual value (WebWeb, WebWeb2)
r32RegisterErrorSide_P232	REAL	PM width register deviation
r32RegisterErrorSideCompensation_P233	REAL	Correction offset side register axial
i32EncoderPosition_P235	DINT	Actual encoder value
i32GateStatus_P236	DINT	Gate state
r32GateOpenPositionWebcylinder_P237	REAL	Gate start position (WebCylinder)
r32GateStopPosRegMarkWebWeb_P238	REAL	Gate end position printing mark (WebWeb, WebWeb2)
r32GateStartPosRegMarkWebWeb_P239	REAL	Gate start position printing mark (WebWeb, WebWeb2)
r32GateStopPosRefMarkWebWeb_P240	REAL	Gate end position reference mark (WebWeb, WebWeb2)
r32GateStartPosRefMarkWebWeb_P241	REAL	Gate start position reference mark (WebWeb, WebWeb2)
i16MarkErrorCounter_p250	INT	PM error counter (WebWeb, WebWeb2)
u32AGSStatus_P252	UDINT	AGS state
r32AGSEndposition_P257	REAL	AGS end position of the block mark
r32DoubleMarkWidth_P261	REAL	PM width overall
r32Sensor1Offset_P279	REAL	Sensor 1 offset 1 (offset reference mark)
r32Sensor2Offset_P283	REAL	Sensor 2 offset 2 (offset printing mark)
r32Signal1ActualLevel_P408	REAL	Signal level head 1 (IDS)
r32Signal2ActualLevel_P409	REAL	Signal level head 2 (DS)
r32PositionIntervalANOC_P411	REAL	Position interval analog oscilloscope
End sActualSensorStatus		
sActualSensorError		
u16ErrorCounter_P944	UINT	TRC internal error counter
au16ErrorNumber_P947	ARRAY OF UINT	TRC error ID
asErrorTimeStamp_P948	ARRAY OF P122_Struct	Error time stamp

sHMIControlData

Table 4-13 sHMIControlData

ab8Sensor1AnalogCurve	ARRAY OF BYTE	analog curve values channel 1 (sensor output, copied in TRC1000Background)
ab8Sensor2AnalogCurve	ARRAY OF BYTE	analog curve values channel 2 (sensor output, copied in TRC1000Background)
ai16Sensor1DigitalCurve	ARRAY OF INT	digital curve values channel 1 (sensor output, copied in TRC1000Background)
ai16Sensor2DigitalCurve	ARRAY OF INT	digital curve values channel 2 (sensor output, copied in TRC1000Background)
ai16Sensor1GateCurve	ARRAY OF INT	gate curve channel 1
ai16Sensor2GateCurve	ARRAY OF INT	gate curve channel 2
r32Gate1StartPos	REAL	start position gate 1 (calculated in fTRC1000Background)
r32Gate1EndPos	REAL	end position gate 1 (calculated in fTRC1000Background)
r32Gate2StartPos	REAL	start position gate 2 (calculated in fTRC1000Background)
r32Gate2EndPos	REAL	end position gate 2 (calculated in fTRC1000Background)

4.3.2 sHMI_Command

The HMI screens will be connected via the sHMI_Command variables to the sTRC1000 interface. The function block “FBHMIDataTransfer” copies the HMI_Command variables to the StdcIO variables and other way round.

In case of using the Application without HMI it is also possible not to use this structure and the conversion function block “FBHMIDataTransfer”. In this way the application will be controlled via the StdcIO interface directly.

In the sHMI_Command structure all printing unit depending variables are collected in array form to realize the HMI connection as easy as possible. The sHMI_command structure is prepared to provide a standard solution with a maximum of 20 print units. If the number of print units is less than twenty the HMI part and the array length of the sHMI_command should be still at 20 (constant NUMBER_OF_PUS in xTypeDef). Otherwise the HMI masks needs to be adapted.

To safe the last active TRC dataset, all the necessary values will be saved in the retain data structure “sTRCRetainDataSet” after download. The retain dataset will be reloaded automatically to the HMI_Command.IN structure after run up of the controller and after a life sign interruption. For this the functions “FCSafeRetain” and “FCRetainRefresh” are used.

All variables of the sTRCRetainDataSet structure are marked in the table of the sHMI_Command.IN structure below in the column “R” with green color.

Table 4-14 sHMI_Command.IN

Name	Type	Description	R
i16TRCNumber	INT	Select TRC number	
sCommissioningDataToTRC			
Sensor_P1_Command	eCommandType		
eResetMode	eSensorResetENUM	Sensor_Reset: soft reset (TRC reboot) Factory_Reset: TRC factory reset	
boTRCReset	BOOL	Trigger TRC reset (depending on eResetMode)	
boWriteTRCParameter	BOOL	Trigger write TRC parameter in diag screen	
Sensor_Set_Parameter	sSensorSetConfig	Sensor set parameter structure	
End sCommissioningDataToTRC			
sTRCCalibration	'sTRCCalibrationInType'	Data structure: TRC calibration	
b8StartTRCCalibration	BYTE	Start sensor calibration	
b8NextStep	BYTE	Go to next calibration step	
i16TRCNumber	INT	Select sensor number to calibrate	
i16TRCHead	INT	Select head1 / head2 to calibrate	
sFiberOpticLength	sFiberOpticLengthInType	Data structure: fiber optic length	
i16TRCNumber	INT	Select sensor number to change fiber optic length	
i16SetLengthHead1	INT	Select fiber optic length head1	

Name	Type	Description	R
		[25] 2.5 m [50] 0.0 m	
i16SetLengthHead2	INT	Select fiber optic length head2 [25] 2.5 m [50] 0.0 m	
b8SetLength	BYTE	Set selected length	
b8ReadLength	BYTE	Read actual fiber optic length	
boTRCConfigDownload	BOOL	Start job download	
boTRCSettingsDownload	BOOL	falling edge: TRC settings (e.g. controller parameter, inching factors, web lengths) will be copied from HMI to SIMOTION	
boDRDSettingsDownload	BOOL	falling edge: DRD settings will be copied from HMI to SIMOTION	
boInsettingSettingsDownload	BOOL	falling edge: Insetting settings will be copied from HMI to SIMOTION	
SensorSetParameter			R
sSensorConfig_P2	sP2Config	Data structure: sensor config P2	R
boMeasuringEdge	BOOL	Measuring edge selection FALSE: rising (front) edge TRUE: falling (back) edge	R
boSensorChoice	BOOL	sensor selection FALSE: head 1 (front side) TRUE: head 2 (back side)	R
eMarkConfig	eMarkType	Block mark Wedge mark Double wedge mark	R
boAutomatic	BOOL	Sensor automatic mode FALSE: manual (trigger and offset values need to be set manually) TRUE: automatic (sensor teaching is used once a mark is inside the gate)	R
boLimitRegError	BOOL	Limited register error FALSE: inactive TRUE: active (register deviation is limited to the actual gate width)	R
boMackleDetection	BOOL	Mackle detection FALSE: inactive TRUE: active (alarm, if limit of permitted active edges in the gate is overrun)	R
eControlMode	eSensorMode	WebCylinder, WebWeb, WebWeb2	R
boSensor1Amplification	BOOL	TRUE: head 1 signal gain (factor 2) active	R
boSensor2Amplification	BOOL	TRUE: head 2 signal gain (factor 2) active	R
boDOAOWindowReference	BOOL	AO/DO reference value FALSE: Oscilloscope refers absolute to axis position (0…360°) TRUE: Oscilloscope refers to gate middle position (P2014)	R

Name	Type	Description	R
i16SensorCycleTime_P3	INT	AO/DO detection refresh cycle	R
r32CylinderCircumference_P10	REAL	Printing cylinder circumference	R
r32SetValueWebWeb_P11	REAL	Printing mark distance set point (WebWeb, WebWeb2)	R
r32GateWidth_P12	REAL	Gate width	R
r32GateOffset_P13	REAL	Gate shift offset	R
r32GatePosition_P14	REAL	Gate position	R
r32OffsetLengthregister_P15	REAL	Correction offset length register	R
r32OffsetSideregister_P16	REAL	Correction offset side register	R
sSensor1TriggerVaule	sTriggerValue	Data structure: sensor 1 trigger values	R
r32ThresholdLevel_RefMark	REAL	Threshold level (trigger level) reference mark (sensor 1)	R
r32ThresholdLevel_RegMark	REAL	Threshold level (trigger level) printing mark (sensor 1)	R
r32Offset	REAL	Offset reference mark (sensor 1)	R
r32SpeedCompensationValue	REAL	TRC dead time compensation (not used)	R
sSensor2TriggerVaule	sTriggerValue	Data structure: sensor 2 trigger values	R
r32ThresholdLevel_RefMark	REAL	Threshold level (trigger level) reference mark (sensor 2)	R
r32ThresholdLevel_RegMark	REAL	Threshold level (trigger level) printing mark (sensor 2)	R
r32Offset	REAL	Offset reference mark (sensor 2)	R
sMarkGeometry	sMarkDefine	Data structure: mark geometry	R
r32Width_Blockmark	REAL	[mm] Width block mark	R
r32Width_WedgeMark_Min	REAL	[mm] Width (short side) wedge mark (0 is possible)	R
r32Width_WedgeMark_Max	REAL	[mm] Width (long side) wedge mark	R
r32EdgeLength_WedgeMark	REAL	[mm] Edge length (height) wedge mark	R
r32Width_DoubleWedgeMark_Min	REAL	[mm] Width (short side) double wedge mark (0 is possible)	R
r32Width_DoubleWedgeMark_Max	REAL	[mm] Width (song side) double wedge mark	R
r32MiddleWidth_DoubleWedgeMark	REAL	[mm] Width (middle) double wedge mark	R
r32Width_DoubleWedgeMark	REAL	[mm] Width (total) double wedge mark	R
r32EdgeLength_DoubleWedgemark	REAL	[mm] Edge length (height) double wedge mark	R
r32Width_DoubleBlockMark_Min	REAL	[mm] Width (total, short side) double block mark	R
r32Width_DoubleBlockMark_Max	REAL	[mm] Width (total, long side) double block mark	R
r32WidthStraight_DoubleBlockmark	REAL	[mm] Width straight mark of double block mark	R
r32WidthBevel_DoubleBlockMark	REAL	[mm] Width oblique mark of double block mark	R

Name	Type	Description	R
r32EdgeLength_DoubleBlockmark	REAL	[mm] Edge length (height) double block mark	R
r32Distance_AGSMarktoRefMark	REAL	[mm] Distance AGS block mark to reference mark	R
r32Tolerence_AGSMark	REAL	[mm] Tolernace range of AGS mark	R
r32Width_AGSMark_1	REAL	[mm] Width block mark (first of three blocks)	R
r32Width_AGSMark_2	REAL	[mm] Width block mark (second of three blocks)	R
r32Width_AGSMark_3	REAL	[mm] Width block mark (third of three blocks)	R
r32Chasm_AGSMark_1	REAL	[mm] Gap between first and second block mark block	R
r32Chasm_AGSMark_2	REAL	[mm] Gap between second and third block mark block	R
r32TriggerVauleMIN_P45	REAL	TRC teaching (travel measurement) min. trigger voltage	R
r32DeviationValueMax_P46	REAL	TRC teaching (travel measurement) max.web voltage deviation	R
i16SignofLifeTolerence_P925	INT	Number of tolerable sign of life interruptions	R
End SensorSetParameter			
aboSelectPUactive	ARRAY [0..20] OF BOOL	Print unit active/inactive	R
r32Graph_Factor	REAL	Factor for graphical register error display	R
ab8AMRStart	'ARRAY [0..20] OF BYTE'	Start AGS (automatic gate setting)	
ab8TriggerValueTeaching	'ARRAY [0..20] OF BYTE'	Start trigger value teaching (travel measurement)	
ab8SetTriggerValues1	'ARRAY [0..20] OF BYTE'	Set trigger values for head 1	
ab8SetTriggerValues2	'ARRAY [0..20] OF BYTE'	Set trigger values for head 2	
aboTriggerHead2	'ARRAY [0..20] OF BOOL'	Display trigger settings for head 2 on HMI screen	
ar32GatePos	'ARRAY [0..20] OF REAL'	Gate position set value	
ab8ChangeGatePos	'ARRAY [0..20] OF BYTE'	Start change gate position	
ab8CenterGate	'ARRAY [0..20] OF BYTE'	Start center gate	
ab8ChangeGateWidth	'ARRAY [0..20] OF BYTE'	Start change gate width	
ar32GateWidth	'ARRAY [0..20] OF REAL'	Gate width set value	
u32ResetError	UDINT	Fault acknowledge (bit 1: printing unit 1, …)	

Name	Type	Description	R
ab8RegInchLRBackwardFast	'ARRAY [0..20] OF BYTE'	Trigger printing cylinder inching backward fast (length register) Inching distance: HMI_Command.IN. r32RegInchLRFactorFast	
ab8RegInchLRForwardFast	'ARRAY [0..20] OF BYTE'	Trigger printing cylinder inching forward fast (length register) Inching distance: HMI_Command.IN. r32RegInchLRFactorFast	
ab8RegInchLRBackwardSlow	'ARRAY [0..20] OF BYTE'	Trigger printing cylinder inching backward slow (length register) Inching distance: HMI_Command.IN. r32RegInchLRFactorSlow	
ab8RegInchLRForwardSlow	'ARRAY [0..20] OF BYTE'	Trigger printing cylinder inching forward slow (length register) Inching distance: HMI_Command.IN. r32RegInchLRFactorSlow	
ab8RegInchSRBackwardFast	'ARRAY [0..20] OF BYTE'	Trigger printing cylinder inching backward fast (side register) Inching distance: HMI_Command.IN. r32RegInchSRFactorFast	
ab8RegInchSRForwardFast	'ARRAY [0..20] OF BYTE'	Trigger printing cylinder inching forward fast (side register) Inching distance: HMI_Command.IN. r32RegInchSRFactorFast	
ab8RegInchSRBackwardSlow	'ARRAY [0..20] OF BYTE'	Trigger printing cylinder inching backward slow (side register) Inching distance: HMI_Command.IN. r32RegInchSRFactorSlow	
ab8RegInchSRForwardSlow	'ARRAY [0..20] OF BYTE'	Trigger printing cylinder inching forward slow (side register) Inching distance: HMI_Command.IN. r32RegInchSRFactorSlow	
r32RegInchLRFactorSlow	REAL	Inching distance printing cylinder jerk slow (length register)	R
r32RegInchSRFactorSlow	REAL	Inching distance printing cylinder jerk slow (side register)	R
r32RegInchLRFactorFast	REAL	Inching distance printing cylinder jerk fast (length register)	R
r32RegInchSRFactorFast	REAL	Inching distance printing cylinder jerk fast (side register)	R
aboLRRegCtrlEnable	'ARRAY [0..20] OF BOOL'	Activate register controller (length register)	
ab8RegCtrlSetFromActValueLR	'ARRAY [0..20] OF BYTE'	Set register controller set value to actual value (length register)	
ab8RegCtrlSetValueZeroLR	'ARRAY [0..20] OF BYTE'	Set register controller set value to zero (length register)	
r32LRRegCtrlKPWebWeb	REAL	Controller gain Kp mode WebWeb (length register controller)	R
r32LRRegCtrlKPWebCyl	REAL	Controller gain Kp mode WebCylinder (length register controller)	R
r32LRRegCtrlIWebWeb	REAL	Controller integral time mode WebWeb (length	R

Name	Type	Description	R
		register controller)	
r32LRRegCtrlIWebCyl	REAL	Controller integral time mode WebCylinder (length register controller)	R
aboSRRegCtrlEnable	'ARRAY [0..20] OF BOOL'	Activate register controller (side register)	
ab8RegCtrlSetFromActValueSR	'ARRAY [0..20] OF BYTE'	Set register controller set value to actual value (side register)	
ab8RegCtrlSetValueZeroSR	'ARRAY [0..20] OF BYTE'	Set register controller set value to zero (side register)	
r32SRRegCtrlKPWebWeb	REAL	Controller gain Kp mode WebWeb (side register controller)	R
r32SRRegCtrlKPWebCyl	REAL	Controller gain Kp mode WebCylinder (side register controller)	R
r32SRRegCtrlIWebWeb	REAL	Controller integral time mode WebWeb (side register controller)	R
r32SRRegCtrlIWebCyl	REAL	Controller integral time mode WebCylinder (side register controller)	R
ar32TRC1PosToPU	'ARRAY [0..20] OF REAL'	Web length between printing unit and corresponding sensor head (head 1) [mm]	R
ar32TRC2PosToPU	'ARRAY [0..20] OF REAL'	Web length between printing unit and corresponding sensor head (head 2) [mm]	R
ar32PUToPU	'ARRAY [0..20] OF REAL'	Web length between printing units [mm] [2]: PU1 <-> PU2 [3]: PU2 <-> PU3 …	R
b16RegisterErrorHistory_Request	WORD	Trigger read register error history	
i16RegisterErrorHistory_SensorNr	INT	TRC selection for register error history	
sHMIInsettingCommandIn	sHMIInsettingCommandInType	Insetting substructure	
eTechnology	eLRegCtrlTechnology	Printing technology (GRAVURE, FLEXO)	R
eControlAlg	eLRegCtrlAlg	Register controller algorithm	R
boEnableIntActionLR	BOOL	Enable/Disable integral part of register control modes with integral part	R
boEnableIntActionSR	BOOL	Enable/Disable integral part of register control modes with integral part	R
sHMIPrintingUnit	sHMIPrintingUnitStruct	Substructure for tension adaption curve and speed level curve	R
sStatistic	sLRegCtrlStatisticIn	Substructure for register error statistic	R
ab8SetRegStarted	ARRAY OF BYTE	Function “SetReg” has been started from HMI	
ab8SetRefStarted	ARRAY OF BYTE	Function “SetRef” has been started from HMI	
eDRDMode	eLRegCtrlDecouplingMod	Selection of the DRD mode	R

Name	Type	Description	R
	e		
au8FilterDepth	ARRAY OF USINT	Averaging over the parameterized number of register error values	R

Table 4-15 sHMI_Command.OUT

Name	Type	Description
aboLifesignError	'ARRAY [0..20] OF BOOL'	TRC sign of life error
aboApplicationError	'ARRAY [0..20] OF BOOL'	Application error
au16ApplicationErrorID	'ARRAY [0..20] OF UINT'	Application error ID
ai32Parameter1	'ARRAY [0..20] OF DINT'	Additional parameter 1 application error id: system function function/parameter result
ai32Parameter2	'ARRAY [0..20] OF DINT'	Additional parameter 2 application error id: parameter number
aboSensorError	'ARRAY [0..20] OF BOOL'	TRC device error
au16SensorErrorID	'ARRAY [0..20] OF UINT'	TRC device error ID
aboRegisterFBErrorLR	'ARRAY [0..20] OF BOOL'	Register controller function block LR error
aboRegisterFBErrorSR	'ARRAY [0..20] OF BOOL'	Register controller function block SR error
au16RegisterFBErrorIDLR	'ARRAY [0..20] OF UINT'	Register controller function block LR error ID
au16RegisterFBErrorIDSR	'ARRAY [0..20] OF UINT'	Register controller function block SR error ID
b32GlobalError	DWORD	Bit 0: global error (one of the print units) Bit 2-x: local error (print unit x)
i32MachineSpeed	DINT	Actual web speed
sCommissioningDataToSimotion		
sActualSensorConfig	'sActualConfig'	Actual TRC configuration data
sActualSensorStatus	'sSensorStatusInfo'	Actual TRC status data
sActualSensorError	'sSensorError'	Actual TRC error information
End sCommissioningDataToSimotion		
b8TRCCalibrationStatus	BYTE	Actual TRC calibration status [0] inactive [1] use RefBox input white

Name	Type	Description
		[2] use RefBox input black [3] use RefBox input white 2nd [4] finished [5] error
i16FiberOpticLengthHead1	INT	Actual fiber optic length head 1 [25] 2.5 m [50] 5.0 m
i16FiberOpticLengthHead2	INT	Actual fiber optic length head 2 [25] 2.5 m [50] 5.0 m
ab8AGSStatus	'ARRAY [0..20] OF BYTE'	Status AGS [0] AGS search inactive [4] AGS valid [5] AGS failure
ab8SensorTeaching	'ARRAY [0..20] OF BYTE'	Status sensor teaching [0] sensor teaching inactive [4] sensor teaching done [5] sensor teaching failure
ab8ChangeGatePos	'ARRAY [0..20] OF BYTE'	Status change gate position [1] gate position update started [4] gate position update done [5] gate position update failed
ab8GateCenter	'ARRAY [0..20] OF BYTE'	Status center gate [1] center gate function active [4] center gate function done [5] gate center function failed
ab8SetTriggerSensor1	'ARRAY [0..20] OF BYTE'	Set trigger values head 1
ab8SetTriggerSensor2	'ARRAY [0..20] OF BYTE'	Set trigger values head 2
ar32ActRegisterOffset_LR	'ARRAY [0..20] OF REAL'	Register offset length register (register fine correction)
ar32ActRegisterOffset_SR	'ARRAY [0..20] OF REAL'	Register offset side register (register fine correction)
aboRegControlActive_LR	'ARRAY [0..20] OF BOOL'	Register controller LR active
aboRegControlActive_SR	'ARRAY [0..20] OF BOOL'	Register controller SR active
aboAlarmMarkPosition	'ARRAY [0..20] OF BOOL'	Bit “Alarm PM Position” (StatusWord2): FALSE: valid PM position and a valid number of active edges within the gate. TRUE: no (valid) PM detected within the gate for the last three printing cylinder revolutions (PM out of gate, no PM within the gate).
aboAlarmMarkWidth	'ARRAY [0..20] OF BOOL'	Bit “Alarm PM Width” (StatusWord2): FALSE: valid PM width within the gate and “Alarm PM Position” is FALSE.

Name	Type	Description
		TRUE: no valid PM width for three printing cylinder revolutions (width exceeds maximum/minimum width limit); “Alarm PM Position” = TRUE
aboGateActive	'ARRAY [0..20] OF BOOL'	Bit “Gate Setting Active” (StatusWord2): FALSE: At least three valid PM positions are detected after a gate setting. TRUE: Shift Gate Position, Set Gate Position.
aboAGSActive	'ARRAY [0..20] OF BOOL'	Bit “AGS Active” (StatusWord2): FALSE: AGS block mark detected, no AGS block mark detected after three print cylinder revolutions, maximum/minimum speed limit exceeded or stand still of the printing cylinder during AGS active TRUE: AGS active
ar32ActualSensorGatePosition	'ARRAY [0..20] OF REAL'	Actual gate position
ar32ActualSensorGateWidth	'ARRAY [0..20] OF REAL'	Actual gate width
ab8GateWidthChange	'ARRAY [0..20] OF BYTE'	Status change gate width [1] gate width update started [4] gate width update done [5] gate width update failed
ar32RegisterErrorHistoryLR	'ARRAY [0..199] OF REAL'	Array includes the last 200 register deviation values
ar32RegisterErrorHistorySR	'ARRAY [0..199] OF REAL'	Array includes the last 200 register deviation values
b16RegisterErrorHistoryTransfer	WORD	Transfer History array to HMI
asUploadParameter	'ARRAY [0..20] OF sUpload_structure'	Data structure for upload data (gate width, gate position, trigger values). With the upload button in the job screen these values can be transferred to the actual job setting which can be saved afterwards.
ar32RegCtrlErrorLR	'ARRAY [0..20] OF REAL'	Register error LR
ar32RegCtrlErrorLRGraphic	'ARRAY [0..20] OF REAL'	Register error LR (graphical view)
ar32RegCtrlErrorSR	'ARRAY [0..20] OF REAL'	Register error SR
ar32RegCtrlErrorSRGraphic	'ARRAY [0..20] OF REAL'	Register error SR (graphical view)
b8TRCGlobalJobDownload	BYTE	Status global job download [0] - [1] job download HMI to SIMOTION [3] job download SIMOTION to TRC [4] download successfully [5] error [6] job download necessary
ab8ctrlactstatus_SR	'ARRAY [0..20] OF	Register controller status (side register) (faceplate

Name	Type	Description
	BYTE'	HMI)
ab8ctrlactstatus_LR	'ARRAY [0..20] OF BYTE'	Register controller status (length register) (faceplate HMI)
sHMIInsettingCommandOut	sHMIInsettingCommandOutType	Insetting substructure
aboDecouplingError	BOOL	DRD decoupling error
au16DecouplingErrNo	UINT	DRD decoupling error number
aboShiftError	BOOL	DRD shift error
au16ShiftErrNo	UINT	DRD shift error number
sStatistic	sLRegCtrlStatisticOut	Substructure for register error statistic
ar32FBRegCtrlOutputMmLR	ARRAY OF REAL	[mm] register controller output (length register)
ar32FBRegCtrlOutputMmSR	ARRAY OF REAL	[mm] register controller output (side register)
ar32FBDRDOutputMm	ARRAY OF REAL	[mm] DRD output (just speed- /accel- depending shift)
aboSetRegActive	ARRAY OF BOOL	Feedback to HMI: Cylinder adjustment triggered by “SetReg” active
aboSetRefActive	ARRAY OF BOOL	Feedback to HMI: Cylinder adjustment triggered by “SetRef” active

NOTE The array length of the sHMI_command variables depends on the constant “NUMBER_OF_PUS” (xTypeDef). The WinCC flexible application works with a basic length of 20 print units and **should not be changed**. The number of active print units (SIMOTION) can be adapted by the constant NUMBER_OF_ACTIVE_PUS (S_Var_Gl) to the machine range.

4.3.3 sTrcHmiOcxCom

Send-data structure for WinCC flexible PrintMarkControl.

Table 4-16 sTrcHmiOcxCom

Name	Type	Description
abopComCommunicate	ARRAY OF BOOL	communication SIMOTION <-> PrintMarkControl active
au16pComSendDataLength	ARRAY OF UINT	data length of send data
aspComLComParameter	aspComLComParameterType	communication parameter
abopComDataReceived	ARRAY OFBOOL	data received indication
sTRC1000ScopeAOData	ARRAY OF BYTE	analog curve data
sTRC1000ScopeDOData	ARRAY OF INT	digital curve data
sTRC1000ScopeGateData	ARRAY OF INT	gate curve data

4.3.4 sTRCConfig

The TRCConfig structure contains the data information for the acyclic data transfer to the TRC1000 hardware. The parameter numbers will be pre-allocated in the startup program by the function “FCTRC1000DefValPresetting”. This function call is essential otherwise the parameter data transfer to the TRC1000 is not working.

The actual parameter values will be added by the application to the structure.

Further information about the TRC1000 parameters itself can be looked up in the Wiedeg documenation (IDS-PN User Manual en V2.0).

Table 4-17 TRCConfig

Name	Type	Description
DataSetUpdate1	sTypeDataSetWrite	Data set 1 for complete parameter download to TRC1000
DataSetUpdate2	sTypeDataSetWrite	Data set 2 for complete parameter download to TRC1000
DataSetUpdate3	sTypeDataSetWrite	Data set 3 for complete parameter download to TRC1000
DataSetReadActivePara	sTypeDataSetRead	Data set read actual TRC configuration data
DataSetReadStatus1	sTypeDataSetRead	Data set 1 read TRC actual status
DataSetReadStatus2	sTypeDataSetRead	Data set 2 read TRC actual status
DataSetReadError	sTypeDataSetRead	Data set read TRC actual error information
DataSetAODO1	sTypeDataSetAODO Request	Data set read TRC actual scope data (head 1)
DataSetAODO2	sTypeDataSetAODO Request	Data set read TRC actual scope data (head 2)

4.4 Function Blocks

4.4.1 FBTRC1000Backgr

Functionality

The function block “FBTRC1000Backgr” provides the basic data handling and TRC control. The main functionalities are:

1. Management of general TRC functions

   All functionalities of the TRC used in during operation will be controlled by this function block e.g. gate setting, trigger and offset setting etc.

2. Fault Handling

   Errors are separated into different error sources (Sensor faults, application faults, FBController faults, DRD- / Insetting faults).

   The categorization and acknowledgement is handled in this function block.

3. Conversion job settings to TRC structures

   The received data for the specific TRC jobs need to be converted to TRC data sets and every parameter needs to be assigned to a TRC parameter number. The function block distributes the values to the respective data set and prepares the data for the acyclic TRC communication.

4. Calibration handling

   To calibrate the fiber optics a specific routine in the TRC device needs to be started and controlled by the application.

5. Scope data reading and evaluation

   The scope data transfer from the TRC to SIMOTION is realized by acyclic-communication (one time per format/ print cylinder rotation). After receiving the data from the TRC device, the FB saves the data in the sHMIControlData structure.

   The data transfer to the WinCC flexible PrintMarkControl is realized by the FBs **FBCom** (connection establishment) and **FBTRC1000TcpIpHMI** (transfer data).

6. Acyclic communication (TRC parameterization)

   The whole job parameterization on the TRC will be done by acyclic communication. Moreover some operation values (e.g. gate changes) and status values need to be written or read by acyclic communication.

   The whole acyclic handling will be done in the background function block.

   The commands of the LDPV1 SIMOTION standard library are used for the acyclic communication.

NOTICE	**For every TRC device one separate FB call is necessary!**

Input and Output Parameters

Table 4-18 scheme of FBTRC1000Backgr

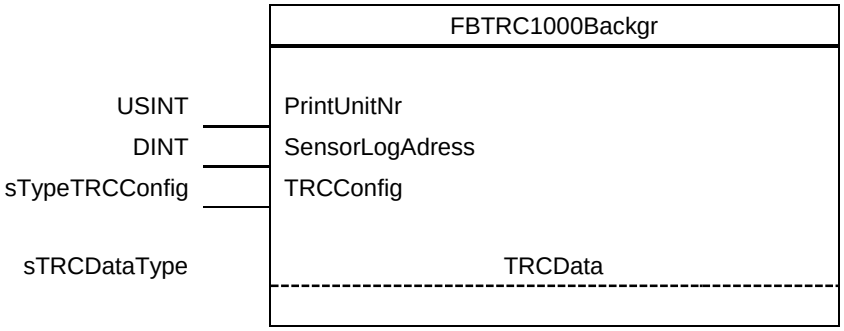


Table 4-19 interface parameters FBTRC1000Backgr

I/O	Name	Type	Description
IN	PrintUnitNr	USINT	Number of printing unit (TRC number)
IN	SensorLogAdress	DINT	Address of TRC device
IN	TRCConfig	sTypeTRCConfig	TRC config data structure
IO	TRCData	sTRCDataType	TRC data structure

Task

The function block has to run in a cyclic task.

Recommended task: background task

4.4.2 FBTRC1000Cyclic

Functionality

The handling of the cyclic data will be done in the “FBTRC1000Cyclic” function block. The main functionalities are:

1. Transfer of control and status word to the cyclic interface

   The input and output data of the TRC are byte array values. The cyclic function block converts the data to the status and control word.

2. Sign of life monitoring

   An internal function block does the monitoring of the TRC sign of life error. In case of error an application error will be created.

NOTICE	**For every TRC device one separate FB call is necessary!**

Input and Output Parameters

Table 4-20 scheme of FBTRC1000Cyclic

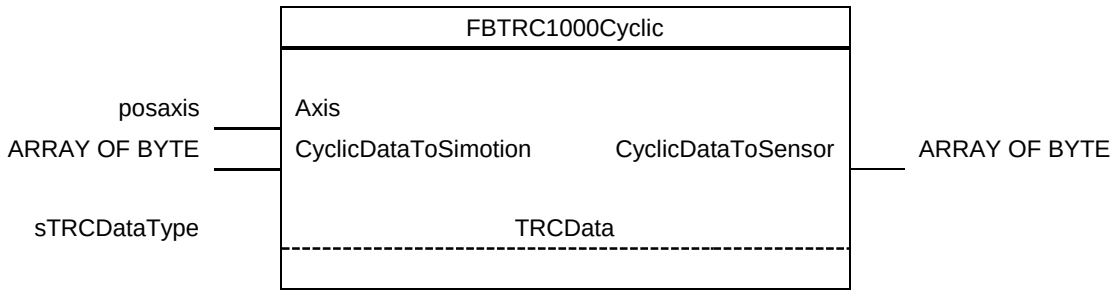


Table 4-21 interface parameters FBTRC1000Cyclic

I/O	Name	Type	Description
IN	Axis	posaxis	Printing cylinder axis
IN	CyclicDataToSimotion	ARRAY OF BYTE	I/O address variable
OUT	CyclicDataToSensor	ARRAY OF BYTE	I/O address variable
IO	TRCData	sTRCDataType	TRC data structure

Task

The function block has to run in a cyclic task.

Recommended task: IPO task

4.4.3 FBHMIDataTransfer

Functionality

The function block “FBHMIDataTransfer” copies data between the two strucutres sHMI_Command and sTRCData.

For HMI projecting it is much easier and more performant to work with a structure which consists of array variables. The variable strucutre sHMI_command is constructed like that.

In SIMOTION it’s easier to handle with single variables. Because of that a function block is necessary which converts the values to the respective print unit TRCData structure and other way round to the respective variable array element of the sHMI_command structure.

NOTICE	**For every TRC device one separate FB call is necessary!**

Input and Output Parameters

Table 4-22 scheme of FBHMIDataTransfer

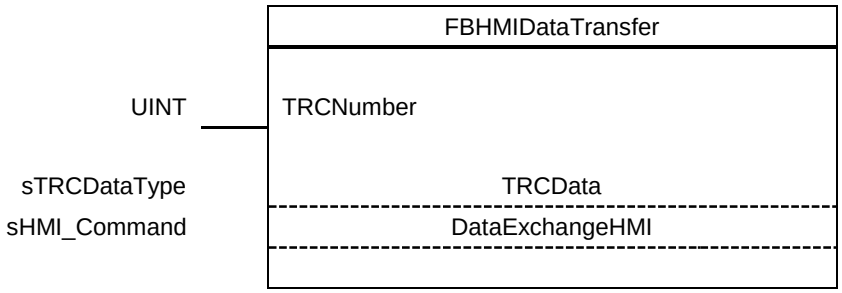


Table 4-23 interface parameters FBHMIDataTransfer

I/O	Name	Type	Description
IN	TRCNumber	UINT	Number of printing unit (TRC number)
IO	TRCData	sTRCDataType	TRC data structure
IO	DataExchangeHMI	sHMI_Command	HMI data structure

Task

The function block has to run in a cyclic task.

Recommended task: background task

4.4.4 FBCom (FBLComMachineCom)

Functionality

The function block is used to establish a TCP/IP connection between SIMOTION and WinCC flexible PrintMarkControl.

The function block is part of the LCom library.

NOTICE	For every TRC device one separate FB call is necessary!

Task

The function block has to run in a cyclic task.

Recommended task: background task

4.4.5 FBLTRC1000TcpIpHmi

Functionality

The function block sends the the curve and gate data to the WinCC flexible PrintMarkControl.

To safe system performance only the data of the active print unit screen on the HMI (scope screen opened) will be transferred.

NOTICE	For every TRC device one separate FB call is necessary!

Task

The function block has to run in a cyclic task.

Recommended task: background task

4.4.6 FBLRegCtrlController

Functionality

The “FBLRegCtrlController” (separate library “LRegCtrl”) contains the register control functionality. The function block is used for all register control solutions (TRC1000, TRC3000, TRC5000). A separate documentation of this function block with detailed information is available

The output signal of the function block is a speed setpoint signal. The value will be coupled to the additional object of the printing cylinder axis by the function block “FBTech”. The Print Standard documentation contains a detailed description of the “FBTech”.

NOTICE	**For every TRC device one separate call of the “FBLRegCtrlController” is necessary!**

Task

The function block has to run in a cyclic task.

Recommended task: IPO task

4.4.7 FBTech

Functionality

The functions block calculates the motion vector and transfers the resulting speed value to the additional object of the respective printing cylinder axis.

Further information about the setpoint coupling to an additional object is available in the Print Standard documentation.

NOTICE	**For every TRC device one separate call of the “FBTech” is necessary!**

Task

The function block has to run in a cyclic task.

Recommended task: IPO task

4.5 Sensor telegrams

The TRC1000 sensor device supports two telegram types:

- Telegram 300 24 Byte output data, 24 Byte input data

Word	Output	Input
1	CW1	STW1
2	CW2	STW2
3	Act. Position	Deviation LR
4		
5	Act. Velocity	Deviation SR
6		
7	Act. Acceleration	Reference mark position
8		
9	Correction Offset LR	Reference mark width
10		
11	Correction Offset SR	Encoder position
12		

- Telegram 301 24 Byte output data, 20 Byte input data

Word	Output	Input
1	CW1	STW1
2	CW2	STW2
3	Act. Position	Deviation LR
4		
5	Act. Velocity	Deviation SR
6		
7	Act. Acceleration	Reference mark position
8		
9	Correction Offset LR	Reference mark width
10		
11	Correction Offset SR	
12		

5 Use of the example application

The example application contains the configuration and parameterization for one TRC device.

The following part shows how to use the application, explaining the HMI masks and the velocity setpoint cascade inside the program.

5.1 Overview

The following figure 5-1 shows an overview about the example program for TRC1000 control and the setpoint cascade of the printing cylinder simulating axis.

The data for job parameterization will be copied from the HMI (sHMI_Command) in the background task to the TRC data structure (sTRCData) of the respective print unit number. The data values will be used by the TRC1000 function blocks.

The velocity correction output of the register controller is converted and transferred to the additional object of the printing cylinder axis and added to the main setpoint from the local master axis.

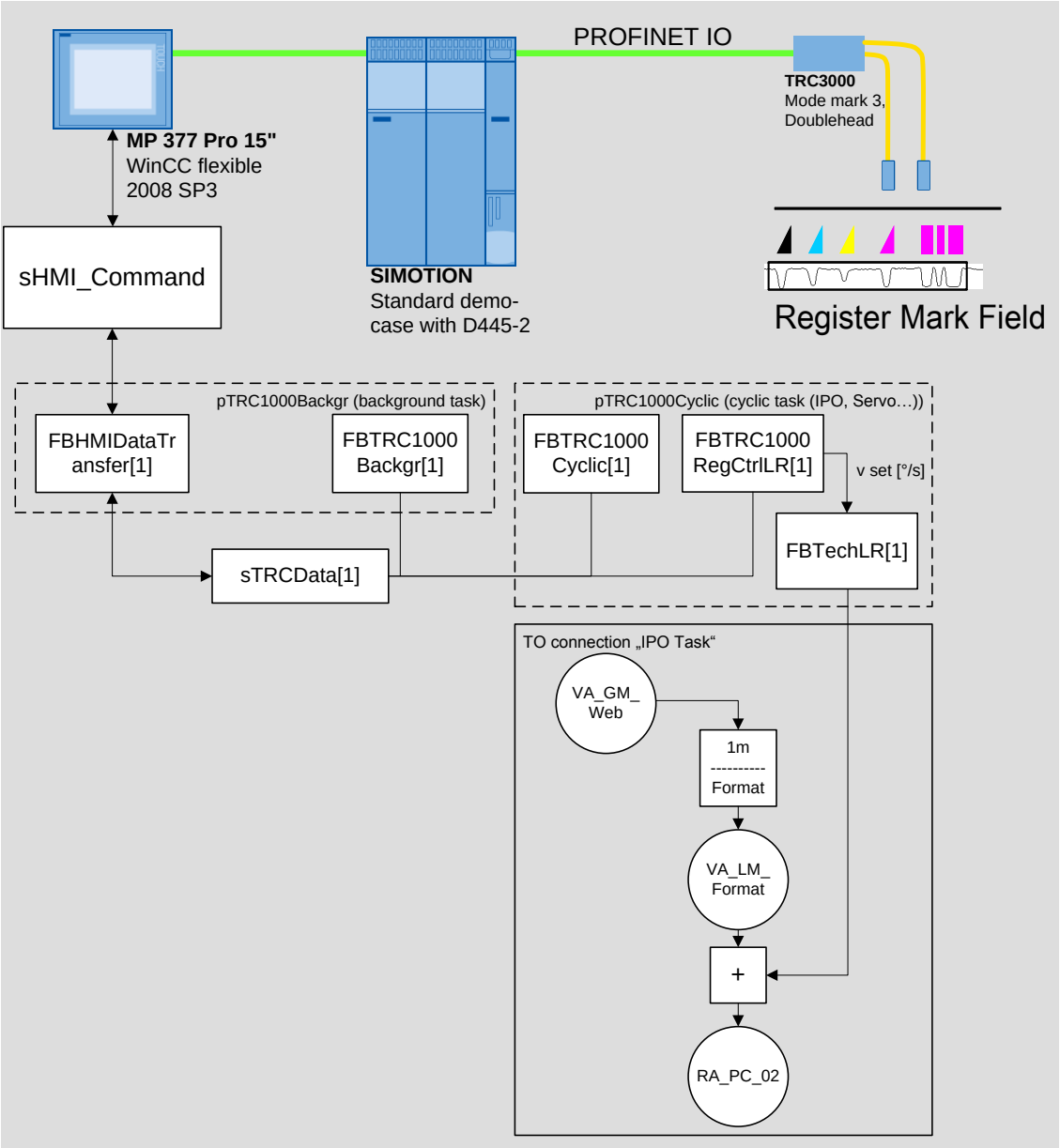
NOTE

The array start index in the TRC data structures and function block instances should be 1.

Generally the index number should correspondent to the respective printing unit. If print unit number one is not equipped with a measuring device the arrays can start with index number 2,3,… .

Index value 0 will be used from the HMI as default data set to safe default values.

Figure 5-1 Example application structure



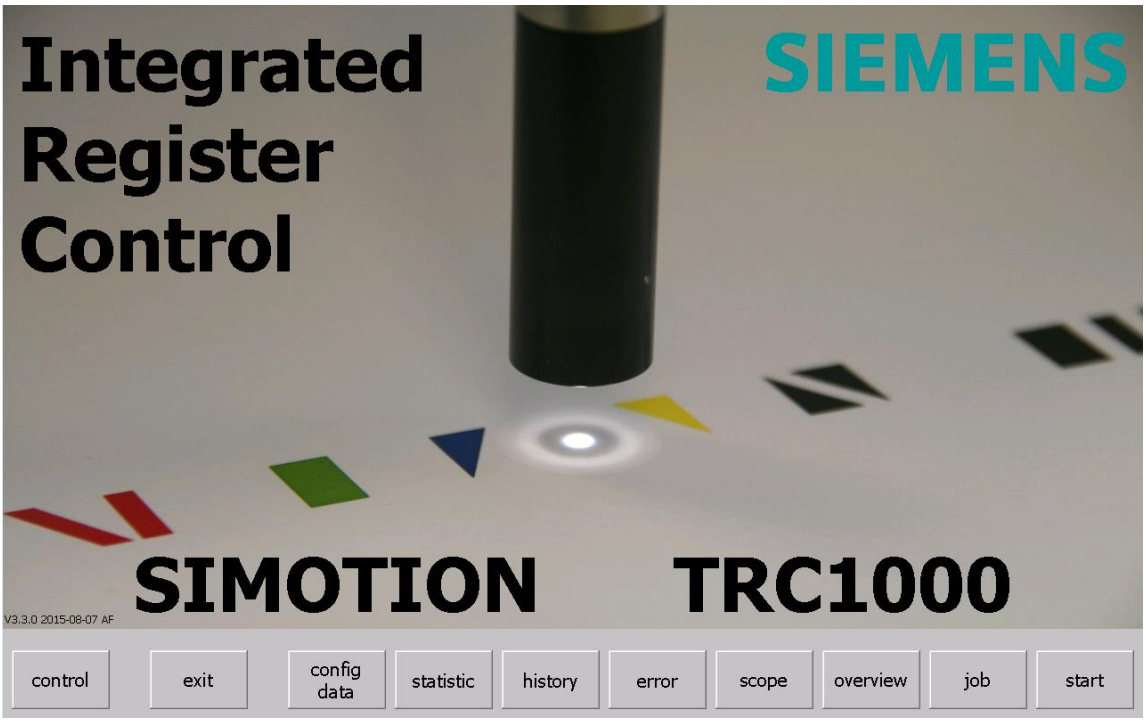
5.2 HMI screens

Start screen

From the start screen all other main-screens are reachable.

Furthermore the runtime can be closed with “exit” button on bottom right.

Figure 5-2 Start screen



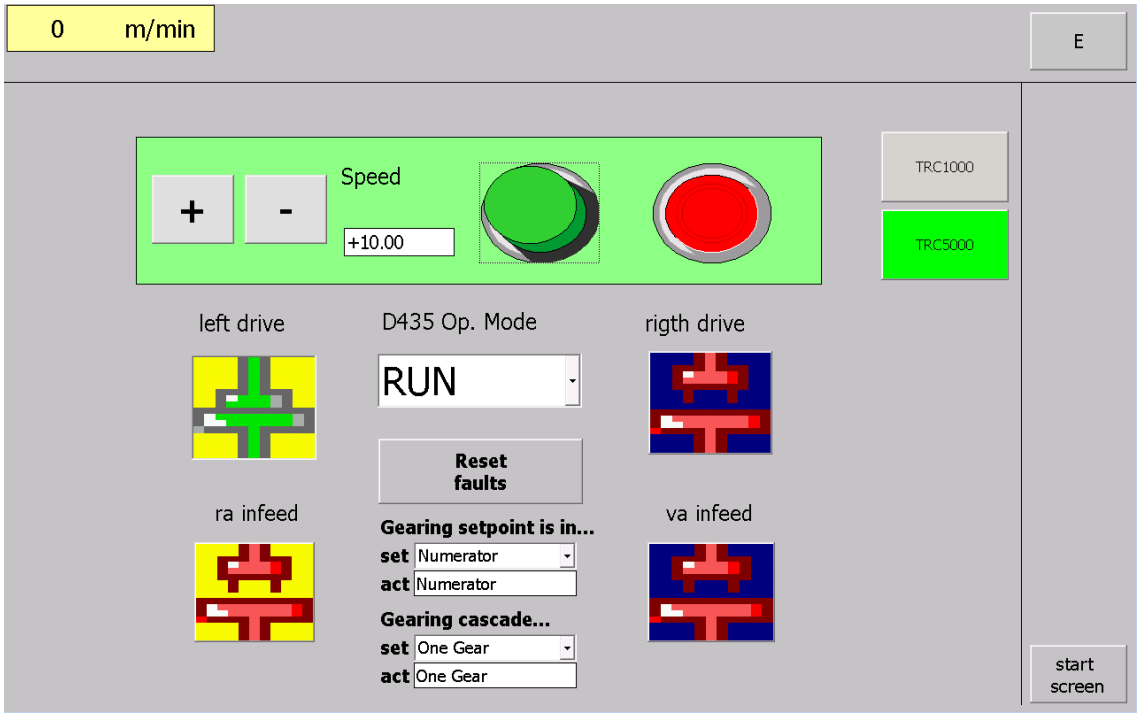
Machine control screen

NOTE This screen is for register control demo model control only!

The axes can be switch to mode 60 (gearing) (green clutch) and the global master axis can be started with the desired speed setpoint.

The TRC1000/TRC5000 switch can be used to switch the register controller influence from TRC1000 to TRC5000 which are both integrated into the demo model.

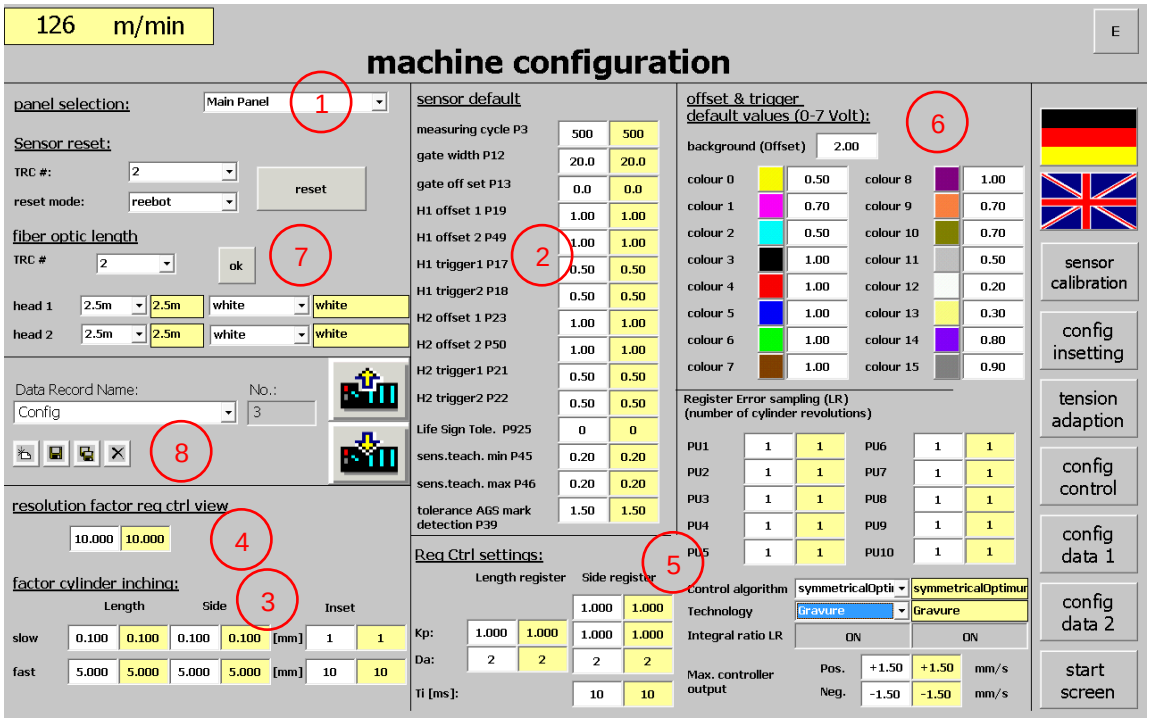
Figure 5-3 Machine Control screen



NOTE The speed input is percentage of a web speed of 600 m/min

Configuration screen 1

Figure 5-4 Configuration screen 1



- **Panel selection (1)**

  The panel selection is to decide if this panel is the main machine panel or one of the local panels at the print unit

- **TRC default settings (2)**

  Some global TRC settings need to be the same **on every unit**. These values will be entered here.

Table 5-1 Default settings

Name	Unit	Default value	Description
Measureing cycle P3	ms	500	AO/DO detection refresh cycle
Gate width P12	mm	20	Gate width (print mark detection)
Gate offset P13	mm	0	Gate shift offset
H1 offset P19	V	1.0	Offset level head 1 (IDS)
H1 trigger 1 P17	V	0.5	1. Threshold level head 1 (IDS)
H1 trigger 2 P18	V	0.5	2. Threshold level head 1 (IDS)
H2 offset P23	V	1.0	Offset level head 2 (DS)
H2 trigger 1 P21	V	0.5	1. Threshold level head 2 (DS)
H2 trigger 2 P22	V	0.5	2. Threshold level head 2 (DS)
sens. teach. min P45	V	0.2	Sensor teaching minimum voltage
sens. teach. max P46	V	0.2	Sensor teaching maximum variation

Name	Unit	Default value	Description
			voltage
Life Sign tole. P925	-	0	Number of tolerable life-sign-counter losses

- **Gate inching / Printing cylinder inching / Insetting gear ratio inching (3)**

  Defines the inch factor for gate/printing cylinder inching in [mm] and the factor for Insetting gear ratio inching [n] of the respective inch buttons (slow and fast).

- **Resolution factor reg ctrl view (4)**

  Resolution of the graphic register error

- **Register controller settings (5)**

  Settings for register controller gain factor and reset time for measuring modes “web web” and “web cylinder”.

- **Color dependent trigger values (6)**

  In the color table different trigger values can be assigned to different colors. The color table should be defined during the first commissioning. Then the color table can be used directly in the job definition.

  The values are saved in the HMI panel.

NOTE All values entered here are saved in the sTRCRetainDataSet structure!

- **Fiber optic length data set selection (7)**

  Read (actual) and write (new) fiber optic length data set dependent of the selected TRC number.

NOTE See chapter 7.1 Fiber optic length

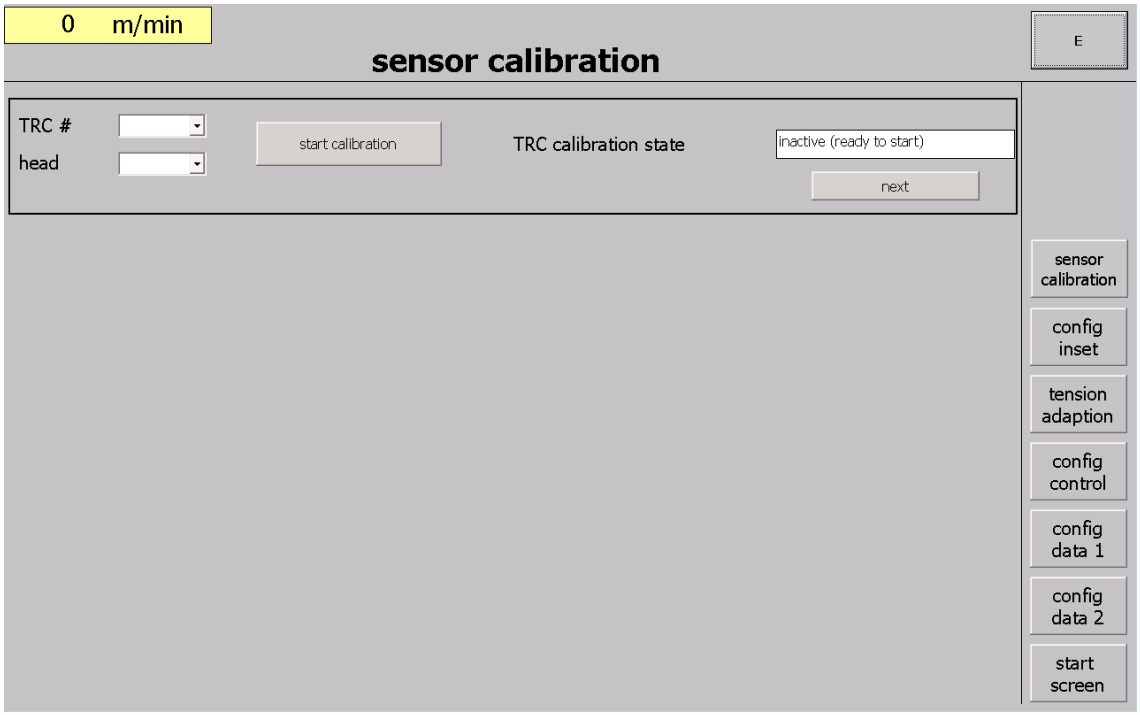
- **Configuration download/upload (8)**

Button	Description
	With these buttons a WinCC flexible recipe can be created and the configuration can be saved into it.
	With the download button the inputs in the white fields (WinCC flexible internal variables) will be copied to the SIMOTION variables (yellow fields).
	The upload button copies the SIMOTION variables to the internal WinCC flexible variables.

Sensor calibration screen

In this screen a sensor calibration can be performed.

Figure 5-5 Sensor calibration screen



NOTE For sensor calibration the Wiedeg sensor gauge is necessary!

Before starting the calibration, the TRC number you want to calibrate hast to be selected.
For calibration of the IDS-PN device select “head 1”, for calibration of the second head (DS) select “head 2”.

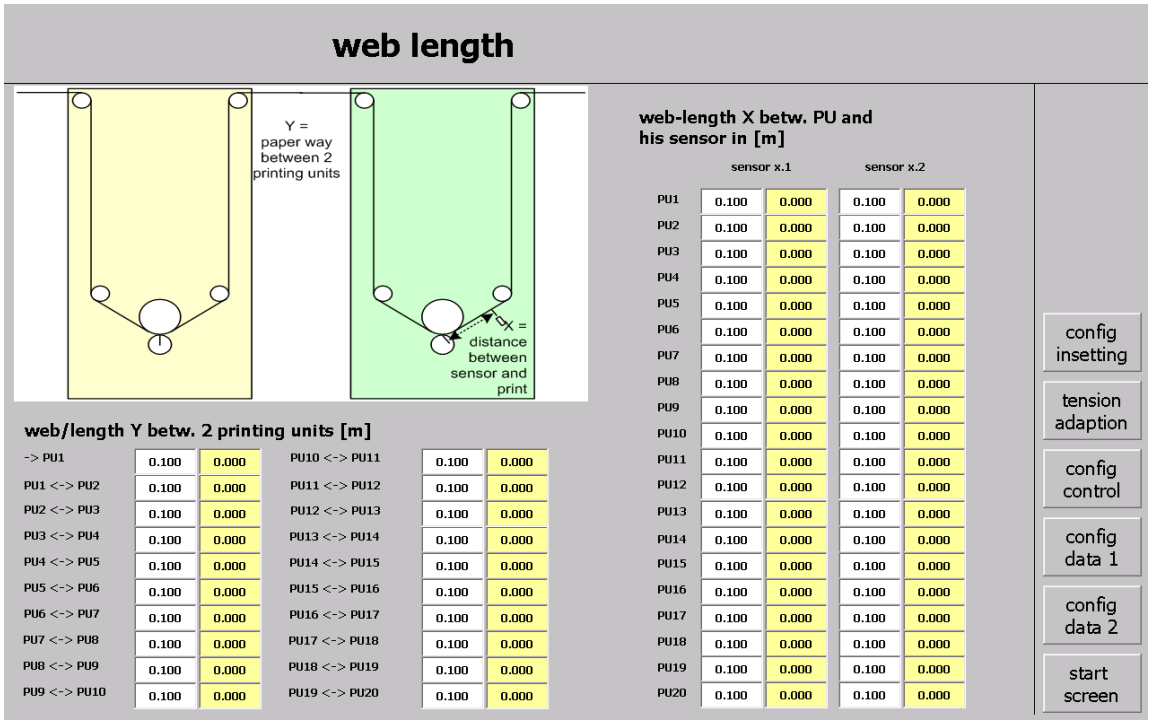
To start the calibration, press the “start calibration” button and follow the instructions.

Press the “next” button not until you have performed the displayed instruction!

Configuration screen 2

The screen “configuration 2” is used for entering the geometrical data of the web lengths of the machine. The lengths between the units and the distances between the print unit nip and the sensor head need to be known by the register controller for calculation of the register correction values.

Figure 5-6 Configuration screen 2



NOTE The values entered here are saved in the sTRCRetainDataSet structure, too!

NOTE The WinCC flexible internal variables (white fields) are copied together with the configuration of “config data 1” screen to the SIMOTION variables (yellow fields).

PrintMarkControl configuration screen

In this configuration screen you can adapt the appearance of the oscilloscope field in the “scope” screen. Attributes like zoom area color, grid color, grid width, etc. can be adapted.

On this screen only the configuration values for PU1, PU2 and PU20 is shown. You can change the values inside these columns directly. You can also adapt the values inside the column “OCX default” and transfer these values with the transfer button (2) to all print units 1-20.

Figure 5-7 PrintMarkControl screen

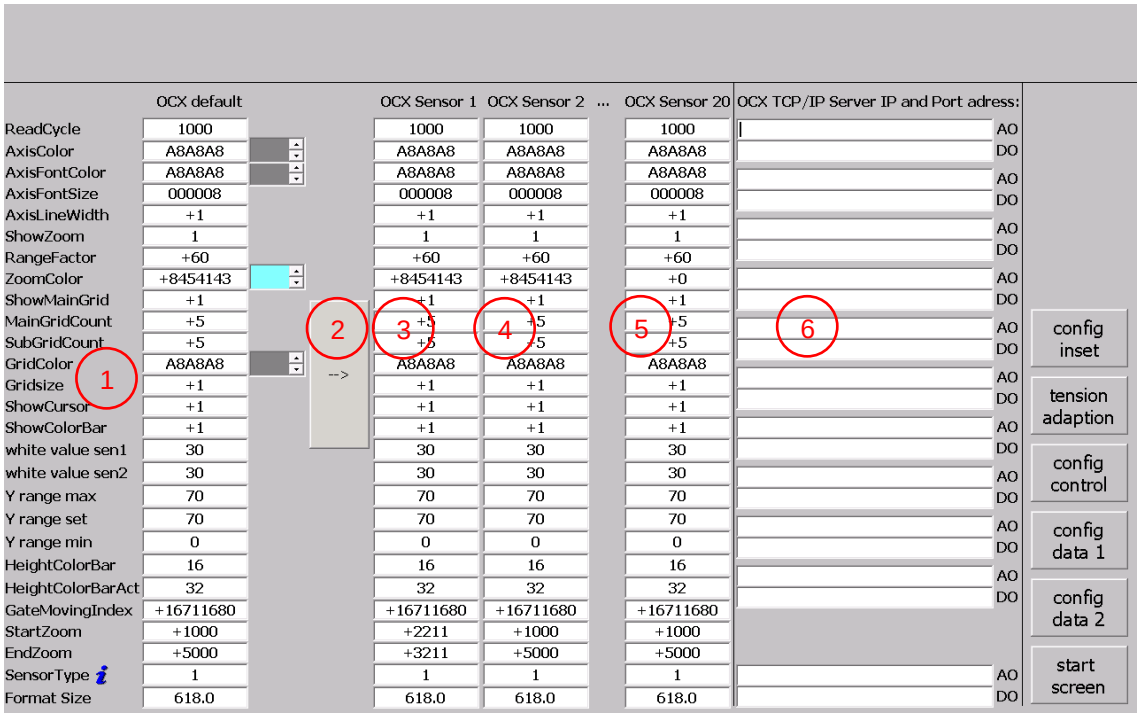


Table 5-2 Default settings

Number	Symbol	Default value Description
1	-	Default settings for OCX PrintMarkControl
1b		Select color from graphic list “OCXColorList”
2		Transfer default setting to OCX PrintMarkControl for sensor 1 to 20
3		settings for sensor1 OCX PrintMarkControl
4		….
5		settings for sensor20 OCX PrintMarkControl
6		TCP/IP server IP and port address set in script “OCX_InitPictureAnalogControlTRC” e.g.

Number	Symbol	Default value Description
		Sensor1 ⇔ “192.168.0.1:1024” Sensor2 ⇔ “192.168.0.1:1025” … Sensor20 ⇔ “192.168.0.1:1043”

Table 5-3 PrintMarkControl config parameter (config screen) (most important parameter)

parameter	default value	unit	description
ReadCycle	1000	[ms]	Cycle time to read curve values from SIMOTION
AxisColor	A8A8A8	[Hex]	Color of axis
AxisFontColor	A8A8A8	[Hex]	Color of axis font
AxisFontSize	8	[pixel]	Font size
AxisLineWidth	1	[pixel]	Line width of axis
ShowZoom	1	[BOOL]	1: show zoom area 0: hide zoom area
RangeFactor	60	[%]	Proportion standard view : zoom area (60 : 40)
ZoomColor	80FFFF	[Hex]	Color of the zoom area
ShowMainGrid	1	[BOOL]	1: show grid lines 0: hide grid lines
MainGridCount	5	[n]	Number of grid lines (main)
SubGridCount	5	[n]	Number of grid lines (sub)
GridColor	A8A8A8	[Hex]	Color of grid lines
GridSize	1	[pixel]	Size of grid line
ShowCursor	1	[BOOL]	1: show cursor 0: hide cursor
ShowColorBar	1	[BOOL]	1: show color bar 0: hide color bar
white value sen1	30	[-]	White balance color value (head 1)
white value sen2	30	[-]	White balance color value (head 2)
Y Range max	100	[-]	Y Zoom max value
Y Range set	100	[-]	Y Zoom set value
Y Range min	0	[-]	Y Zoom min value
HeigthColorBar	16	[pixel]	Height of the color bar when inactive
HeigthColorBarAct	32	[pixel]	Height of the color bar when active
GateMovingIndex	0	[n]	Actual gate index which has been moved
StartZoom	1000	[-]	Actual start position of zoom area
EndZoom	5000	[-]	Actual end position of zoom area
SensorType	1	[-]	Actual sensor type
FormatSize	618.0	[mm]	Actual format

Table 5-4 event of analog and digital PrintMarkControl

Name	Parameter	description
Change(LastFiredCustomEvent)		This event is triggered if of the following events were done: GateChanged = 2 CursorXChanged = 3 CursorYChanged = 4 ConnectionStateChanged = 5 ZoomRangeChanged = 6 Touched = 10 To react to this event a corresponding function list can be deposited at the control properties under events for change.

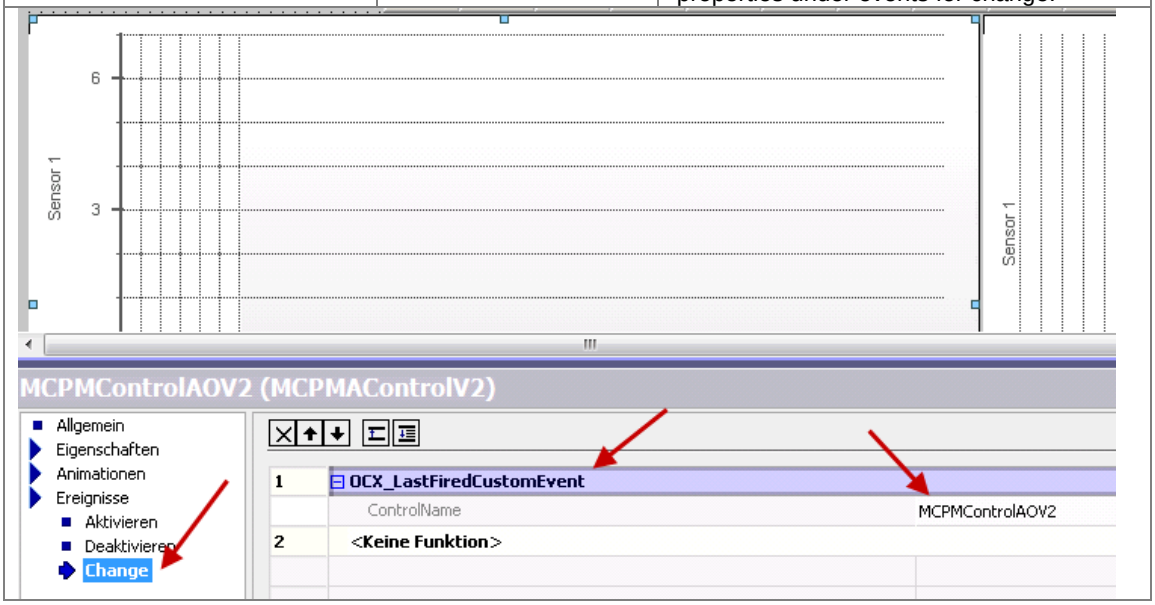


Table 5-5 change events of print mark control

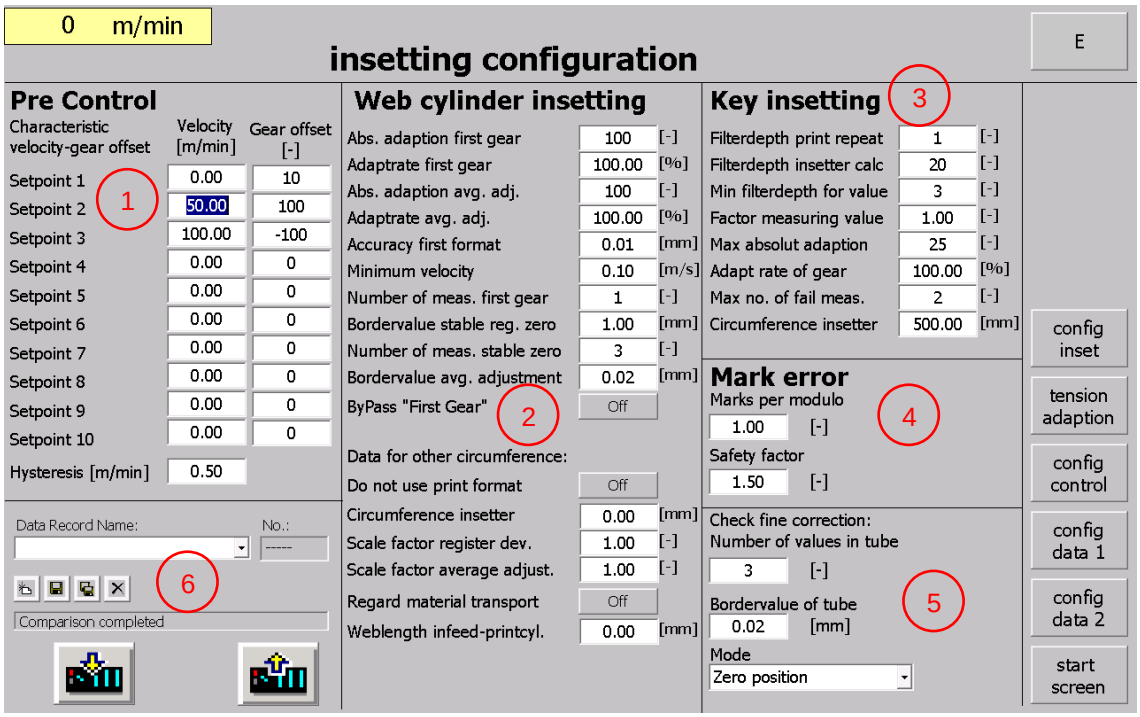
Name	Parameter	values / description
ZoomRangeChanged = 6	void	This event is triggered if position or size of the zoom area was changed. To react to this event a corresponding function list can be deposited in script OCX_LastFiredCustomEvent in select case 6:
GateChanged = 2	void	This event is triggered if the position of one gate was changed. To react to this event a corresponding function list can be deposited in script OCX_LastFiredCustomEvent in select case 2:
CursorXChanged = 3	void	This event is triggered if the position of the X-cursors was changed. To react to this event a corresponding function list can be deposited in script OCX_LastFiredCustomEvent in select case 3:
CursorYChanged = 4	void	This event is triggered if the position of the Y-cursors was changed.

		To react to this event a corresponding function list can be deposited in script OCX_LastFiredCustomEvent in select case 4:
ConnectionStateChanged = 5	void	This event is triggered if the status of TCP/IP connection to SIMONTION changed. To react to this event a corresponding function list can be deposited in script OCX_LastFiredCustomEvent in select case 5:
Touched = 10	void	This event is triggered if the after property *NotifyNextTouchPosition=1* was set. With Property*TouchPosition* the selected gate position can be read out. To react to this event a corresponding function list can be deposited in script OCX_LastFiredCustomEvent in select case 10:

Insetting configuration screen

In this config screen Insetting settings can be done.

Figure 5-8 Insetting Configuration screen



- **Gearing level pre control (1)**

  Speed depending format changes can be pre control with this curve.

- **Setting for web cylinder Insetting (2)**

  Web cylinder Insetting is that the gear of PU is adjusted and register control is active, where the TRC is connected to.

- **Setting for key Insetting (3)**

  Key Insetting is that the PU is not control where the TRC is connected to. In stat the tension unit is adjusted

- **Setting for** FBLInsetMarkValid **(4)**

SafetyFactor	Tolerance to find the mark. Example: SafetyFactor = 1.5: Waiting for 1.5 printing formats, until the message “NoMarkThisCycle” appears.
MarksPerModulo	Number of marks per modulo cycle

- **Setting for** FBLInsetCheckFineCorrection **(5)**

- **Configuration download/upload (8)**

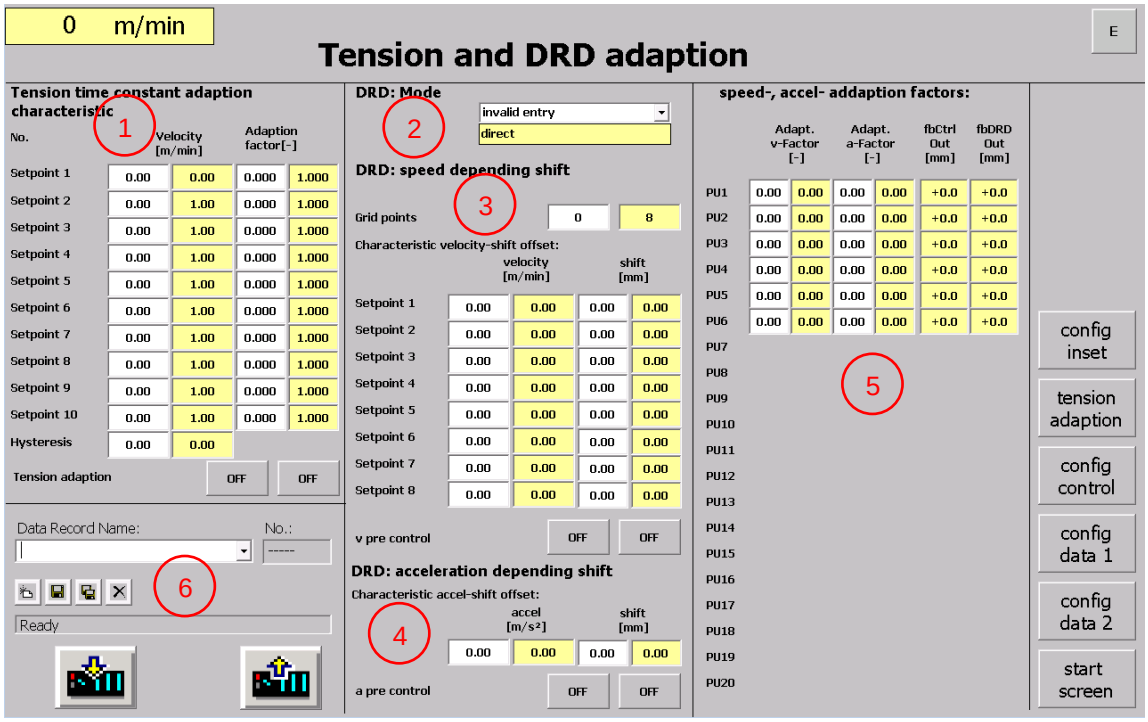
Button	Description
	With these buttons a WinCC flexible recipe can be created and the configuration can be saved into it.
	With the download button the inputs in the white fields (WinCC flexible internal variables) will be copied to the SIMOTION variables (yellow fields).
	The upload button copies the SIMOTION variables to the internal WinCC flexible variables.

NOTE For detailed information see “Print Standard Add-On Insetting” documentation.

Tension Adaption and speed depending shift screen

In this screen two speed depending curves for precontrol can be entered and activated.

Figure 5-9 Tension Adaption screen



- **Tension time constant adaption curve (1)**

  Input values for speed depending adaption for weblength between two PUs. Has influence on register controller on web-cylinder mode and gravure printing. Used for **FBTRC1000CharacteristicLR.**

  Values between two grid points are interpolated and after the last extrapolated. If tension adaption is of or an error in **FBTRC1000CharacteristicLR** the adaption factor is 1.0.

- **DRD mode selection (2)**

- **Speed depending shift curve (3)**

  Input values for speed depending shift curve of DRD function speed shift mode.

  Values between two grid points are interpolated and after the last extrapolated.

- **Acceleration depending shift curve (4)**

  Input values for acceleration depending shift curve of DRD function “accel shift mode”.

- **Adaption factors for following units (5)**

  Due to the fact the characteristic for speed depending shift and also the value for acceleration shift is the same except for an factor between the different units, there is only one characteristic. The characteristic of the following units is adjusted by the adaption factors.

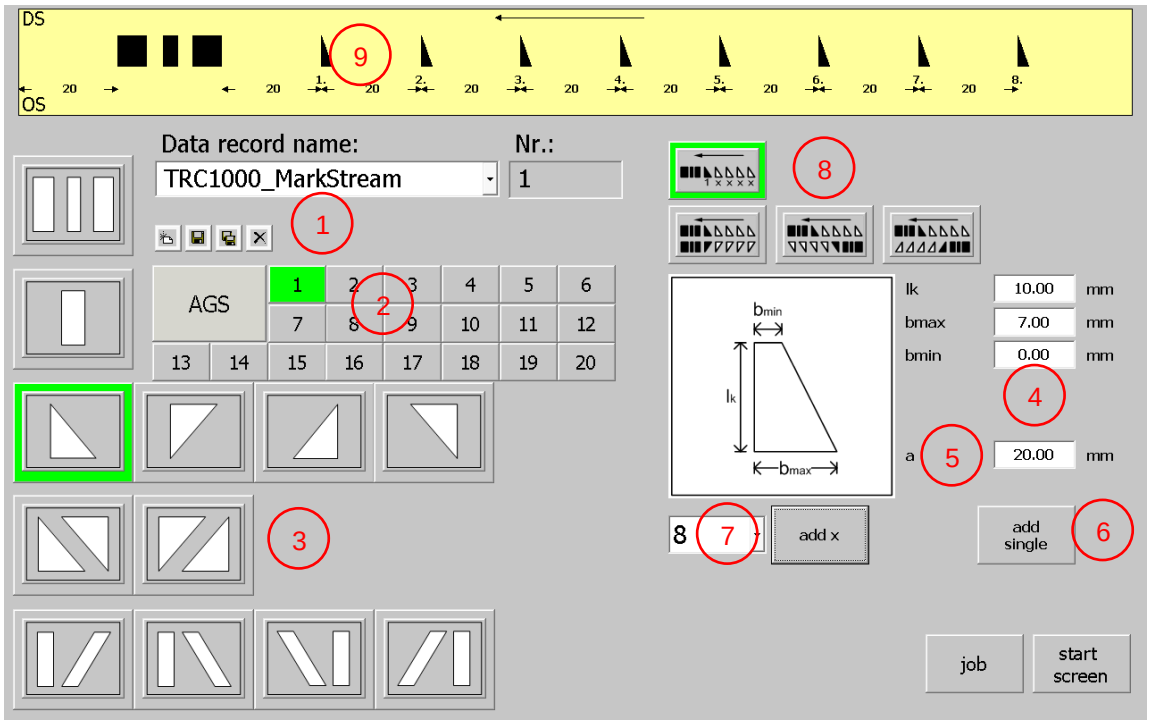
- **Configuration download/upload (6)**

Button	Description
	With these buttons a WinCC flexible recipe can be created and the configuration can be saved into it.
	With the download button the inputs in the white fields (WinCC flexible internal variables) will be copied to the SIMOTION variables (yellow fields).
	The upload button copies the SIMOTION variables to the internal WinCC flexible variables.

Mark stream screen

In the “mark stream” screen a mark stream for the printing job can be defined.

Figure 5-10 Mark stream screen



- **Create mark stream (1)**

Table 5-6 mark stream edit buttons

Button	Description
	Generate a new mark stream dataset
	Save the actual mark stream dataset (All data will be saved in the panel)
	Delete the selected mark stream dataset

- **Printing mark number selection (2)**
  - AGS: AGS block mark
  - 1...20: Printing mark number (maximum 20 marks / stream)

- **Printing mark type definition (3)**
  - Wedge
  - Double wedge
  - Block
  - Double block

NOTE A detailed description of the mark type definitions especially of the active edges can be found in the Wiedeg manual (IDS-PN User Manual en.pdf).

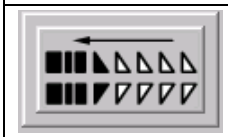
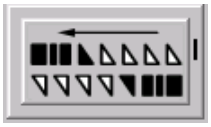
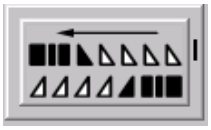
- **Printing mark geometry setting (4)**
  Geometry settings dependent on the selected mark type

- **Distance setting (5)**
  Distance to previous printing mark

- **Add single print mark (6)**
  Add printing mark to the mark stream

- **Add multiple print marks (7)**
  Shows the actual view of the created mark stream

- **Mark stream preview (8)**
  Shows the actual view of the created mark stream

Schaltfläche	Beschreibung
	The button can be used to reset one of the inverse mark stream selections to the standard orientation.
	This button changes the mark stream to mirrored stream in side direction.
	With this button the mark stream can be switched to a stream which results after mounting the cylinder in opposite direction (180°) into the print unit.
	If there is a mirrored mark field (in web direction) printed to the web, the mark stream can be switched to reverse with this button.

- **Mark stream preview (9)**
  Shows the actual view of the created mark stream

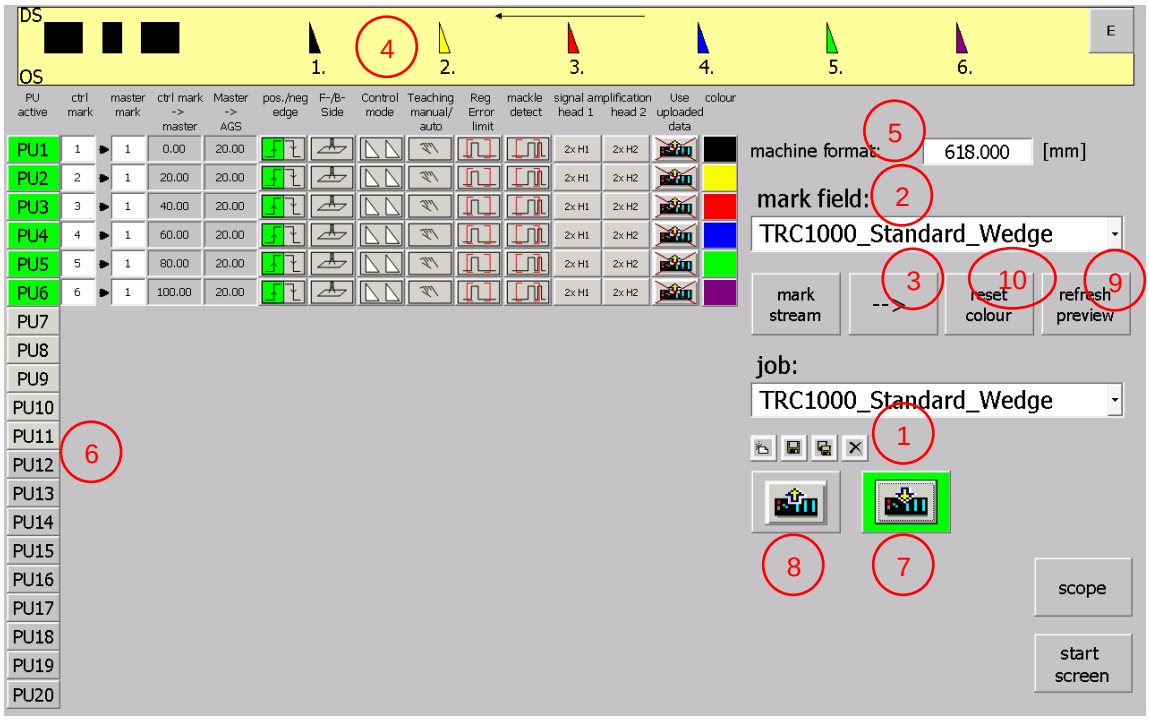
To create a new mark stream or insert a new mark to the stream the following steps need to be done:

- (Press the “new” button and forgive a name)
- Enter the machine format length (top right)
- **Select the mark numbe**r that should be added to the stream (AGS, 1, 2, …)
- **Selection of mark type** (block, wedge…)
- Entering of **mark geometry** values (length, width…)
- Entering the **distance** to previous mark
- Press the **“add”** button to shift the mark to the stream
- **Save** the finished mark stream to use it in the job screen

Job configuration screen

In the screen “job configuration” the printing job specific settings will be done.

Figure 5-11 Job configuration screen



<u>If a mark stream is available a job can be prepared:</u>

- **Create / select a (new) job (1)**

Table 5-7 mark stream edit buttons

Button	Description
	Generate a new job
	Save the actual job setting (All data will be saved in the panel)
	Delete the selected job

- **Select an existing mark stream (2)**

  Select a stored mark stream from the drop down menu.

NOTE Before a new job can be used the respective mark stream need to be created in the “mark stream” picture which can be reached by pressing the button “mark stream”.

- **Copy mark stream to the job (3)**

  With this button a defined mark stream data can be copied into the job definition.

- **Mark stream preview (4)**

  After copying a mark stream to the job (button 3) the mark stream preview shows the actual view of the created job.

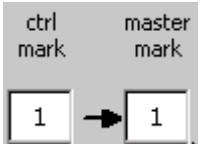
- **Machine format length (5)**

  Enter the machine format length (usually print cylinder circumference)

- **Activated printing units (6)**

  After copying a mark stream to the job (button 3) the (for this job) necessary printing units / sensor devices will be shown / activated.

Table 5-8 job settings buttons

Button	Description
	Activate (green) / deactivate (grey) TRC device
	Selection for register mark and reference mark: • ctrl mark: “own” printing mark printed by this printing unit • master mark: reference mark
	The distance between reference mark and print unit mark will be calculated (dependent on the mark stream) and shown in this fields.
	Mark edge setting: • Positive edge detection • Negative edge detection
	Sensor switch button: front side / back side Print mark assignment Sensor head1 / head2. The assignment depends on the register control mode. See table 26 on page 41 in the “IDS-PN User Manual en” documentation.
	Register control mode: • Web cylinder

Button	Description
	• Web web • Web web 2 (second sensor (DS) is required) • Insetting web cylinder • Insetting Key
	Sensor mode setting (Sensor Automatic): • Manual: No automatic sensor trigger value calculation. The trigger values have to be set by the user. • Automatic: Sensor trigger values will be calculated automatically via “travel measurement”. The “travel measurement” is started after every gate centering.
	Activate / deactivate register error limiting: The calculated register deviation (LR) is limited to the actual gate width to prevent bigger outliers. If such an outlier occurs, an internal counter (P219) is incremented and the deviation is set to 0. **This function is mainly used in the phase of development.** Default: deactivated
	Activate / deactivate mackle detection: If the limit of permitted active edges (either rising or falling edge, dependent on the actual PM type) is overrun, the PM alarm is released and the deviation is set to 0. As soon as a valid number of active edges is detected again, the PM alarm will reset. **This function is implemented mainly as operator support during the print setup operation.** Default: deactivated
	Use uploaded data: With this button the sensor configuration can be switched:  : use the actual machine job.  : use job default setting.
	Color button: With the “color” button the print unit mark color can be selected to get a better overview about the job colors on the HMI screens. **This color matches with the background color of the register controller screen!**

- **Job download (7)**

  After the job definition has been completed the actual job can be transferred to the TRC device(s) with the download-button.

  All data will be transferred to SIMOTION first and following loaded to the TRC devices.

  This can take some time!

Table 5-9 status colors

Color	Description
 **[flashing]**	New job download necessary. Start the job download by pressing the download button.
	Download from panel to SIMOTION
	Download from SIMOTION to TRC device
	Download successfully.
	An error has been occurred. See the error screen for detailed information.

- **Job upload (8)**

  With the upload-button the active job inside the TRC device can be uploaded to the panel.

- **Refresh preview (9)**

  Generally the preview should be refreshed automatically when using a touch screen if there was a change on the mark stream (print unit mark color, mark stream itself, etc.).

  If the refresh is not working automatically the button “preview refresh” button can be used.

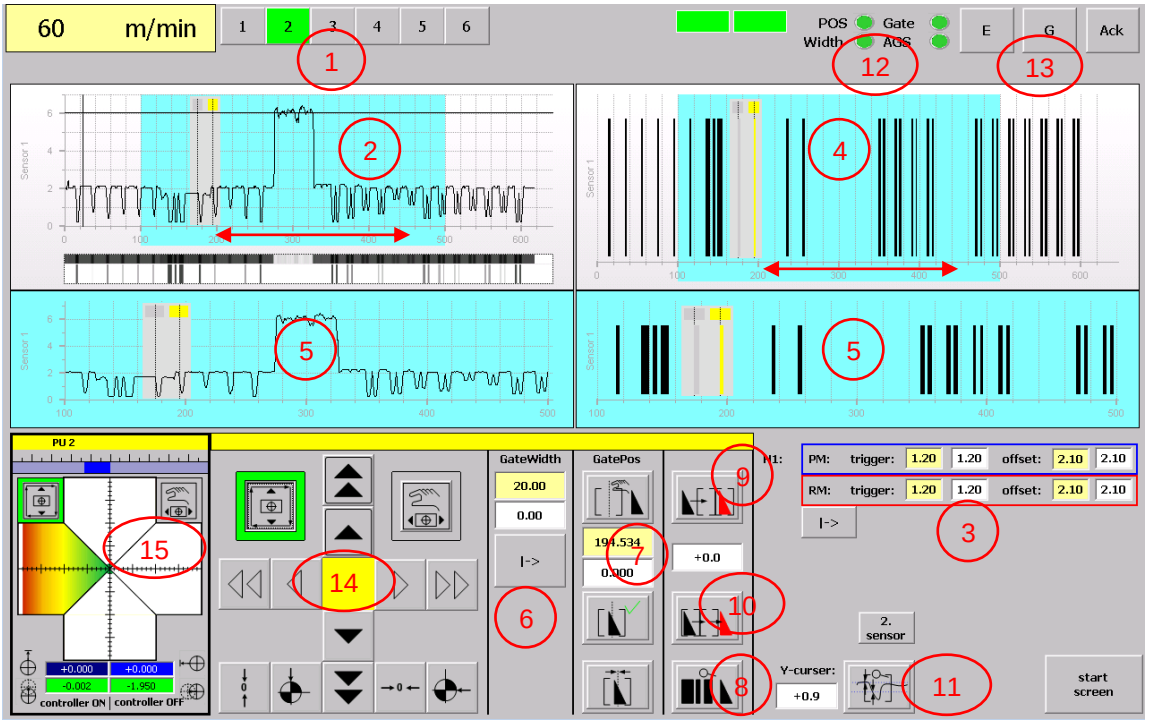
- **Reset color (10)**

  All print unit mark colors (defined by the user) will be reset to black.

Oscilloscope screen

This screen is to show the sensor oscilloscope as well as operating the register functionalities and register controller.

Figure 5-12 Analog/Digital oscilloscope screen



NOTE The request of the measured values will be started when changing to the scope screen. For measuring the printing cylinder axis need to be angular synchronous. Otherwise no values will be sent from the TRC to SIMOTION.

- **TRC number selection (1)**

  With the TRC number selection you can switch through the different scope screens of the active sensor devices.

- **Analog oscilloscope with cursors (2)**

  In this diagram the sensor analog signal curve will be displayed. The X-axis represents the format length (millimeter), the Y- axis the sensor analog signal (voltage).

NOTE If the register control mode “web web 2” is selected, the analog curve of the second sensor appears additionally (red color) in the analog oscilloscope filed.

- **Sensor trigger value setting (3)**

  Depending on the offset and trigger (threshold) values, the analog signal is sampled and converted into a digital signal which is necessary for the printing mark detection.

  Offset:

  The offset is the basic level of the material (background color).

  Set the offset value to that value the background level has (background color). You can read out this value of the analog oscilloscope scale:

  Figure 5-11: offset = **2.0 (**approximately**)**

  Only the reference mark offset can be set by the user. The printing mark offset is set automatically.

NOTICE	The easiest way to set the offset and trigger level is to bring one of the print marks of the analog oscilloscope inside the gate (set gate position) and start the sensor teaching functionality. If the sensor teaching doesn't work try again with another print mark. Alternatively the offset and trigger level can be set manually: see below!

Trigger (threshold):

With the trigger level the printing marks which will be converted into the digital curve is determined.

If the analog signal of a printing mark is higher than the trigger level (based on the offset level), the printing mark will be shown in the digital curve.

With the "2. Sensor" button the user has the possibility to adjust the trigger value setting for sensor 2 (head 2).

- **Digital oscilloscope (4)**

  Based on the sensor analog signal and the sensor trigger value setting (2), the digital signal curve will be generated.

NOTE	If the register control mode "web web 2" is selected and the trigger value setting for sensor 2 has been done, the digital curve of the second sensor appears additionally on top in the digital oscilloscope filed.

- **Zoom curve (5)**

  The zoom curve can be used for a better diagnostic of a specific area of the format length.

  The zoom area can be shifted left and right in this area only, the arrow is inserted in the screenshot.

  To increase or decrease the zoom area just grab the right or left edge and shift it.

With a double click in the zoom area, this area can be enlarged. The standard view will become smaller than.

With a double click in the standard view, the standard view will be enlarged again.

The proportion between the two areas can be defined in the control config screen with the value “RangeFactor”

(default: “60” = 60:40, standard view : zoom area)

- **Gate adjustment (6)**

  Inside the so called gate, the measuring of the print mark takes place.

NOTE To calculate the register deviation, the printing marks on which the calculation is based on, has to be within the gate!

The TRC differs between two different gates:

- Blue gate: for the register printing mark
- Red gate: for the reference printing mark

NOTE For web cylinder mode only the red gate for reference mark is available.

With the **“Cen.” (Center) button** the active edge of the printing mark can be centered within the gate. (The gate will be moved!)

NOTE Condition to use the center gate function is that the printing mark is (valid) within the gate!

NOTE The center gate functionality needs up to 4 printing cylinder revolutions!

With the gate width set button the gate width can be adjusted.

The yellow filed shows the actual gate width.

The white filed is for entering the new gate width.

- **Select Gate Position (7)**

  With this functionality the gate position (Control Mark gate) can be changed just by 3 clicks.

  After pressing the “select gate pos” button (button becomes green) the oscilloscope field gets sensitive for selecting a set position. With the second click the desired set position can be selected by clicking on the desired spot in the oscilloscope field. The third click on “set gate” sets the new gate position.

  The set value can also be entered into the input filed in between the two buttons or using the x-cursor.

- **AGS (Automatic Gate Setting) (8)**

  Using a special printing mark code (AGS block mark) at the beginning of the mark stream the sensor will try to detect this AGS block mark and set the gate automatically.

  With the “AGS start” button this functionality can be started.

  If the ABS mark was found the “AGS start” button becomes green for a short time. If no mark can be found, the “AGS start” button becomes red and a sensor error appears in the error screen.

- **Set Ref (9)**

NOTE This function is mainly applicable using the Print Standard Add-On **TRC1000**.

The reason is the gate setting procedure is started with **print mark gate**. After setting up the print mark gate the reference gate can be setup with “Set Ref”.

This functionality can be used if the reference gate is at the right position but the reference mark is not inside the gate. Click on the reference gate (only in the zoom area) and shift it to the reference mark. The value below the button shows the distance the cylinder has to move to bring the mark into the gate. The adjustment can be started by pressing the button “Set-Ref”.

The cylinder will move by the shortest way to bring the mark into the gate.

- **Set Reg (10)**

NOTE This function is mainly applicable using the Print Standard Add-On **TRC3000**.

The reason is the gate setting procedure is started with **reference gate**. After setting up the reference gate the print mark gate can be setup with “Set Reg”.

This functionality can be used if the reference gate and reference mark is at the right position (mark inside the gate), the control gate is in the defined distanced to the reference gate but the control mark is not inside the control gate. Click on the control gate (only in the zoom area) and shift it onto the control mark. The value above the button shows the distance the cylinder has to move to bring the mark into the gate. The adjustment can be started by pressing the button “Set-Reg”.

NOTE Just in WebWeb mode and register controller is switched off!

- **Sensor teaching (Travel measurement) (11)**

  Using this function the sensor calculates the offset and trigger values automatically.

NOTICE	**Condition to use the function "sensor teaching" is a print mark within the gate!** **Furthermore the parameter P45 and P46 need to be unequal 0!**

NOTE If "Sensor Automatic" is activated, the sensor teaching is executed after every gate centering.

- **Status signals (12)**

Table 5-10 TRC internal status

	**Bit "Alarm PM Position" (StatusWord2):** FALSE = GREEN: valid PM position and a valid number of active edges within the gate. TRUE = RED: no (valid) PM detected within the gate for the last three printing cylinder revolutions (PM out of gate, no PM within the gate).
	**Bit "Alarm PM Width" (StatusWord2):** FALSE = GREEN: valid PM width within the gate and "Alarm PM Position" is FALSE. TRUE = RED: no valid PM width for three printing cylinder revolutions (width exceeds maximum/minimum width limit); "Alarm PM Position" = TRUE
	**Bit "Gate Setting Active" (StatusWord2):** FALSE = GREEN: At least three valid PM positions are detected after a gate setting. TRUE = Yellow: Shift Gate Position, Set Gate Position.
	**Bit "AGS Active" (StatusWord2):** FALSE = GREEN: AGS block mark detected, no AGS block mark detected after three print cylinder revolutions, maximum/minimum speed limit exceeded or stand still of the printing cylinder during AGS active TRUE = Yellow: AGS active

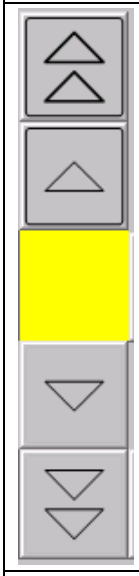
- **Error/Acknowledge (13)**

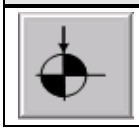
Table 5-11 error/acknowledge buttons

Button	Description
	Local error: Error at this printing unit / sensor device. Press the button to get detailed information.
	Global error: Error at any printing unit / sensor device. Press the button to get detailed information.
	Fault acknowledge

- **Register inching and controller buttons (14)**

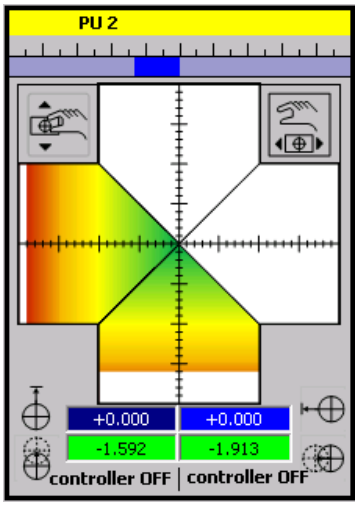
Table 5-12 register controller buttons

Button	Description
	enable/disable register controller
	These jerk buttons can have <u>two different</u> functionalities: • Register controller inactive (off): Printing cylinder jerk • Register controller active (on): Register fine correction (offset) Two jerk steps are available: • jerk slow: (default: 1mm per edge) • jerk fast: (default: 5mm per edge) The jerk steps can be changed in the “Configuration 1” screen.
	Set fine correction offset back to zero

Button	Description
	Set fine correction offset to the actual register error

- **Register Control Faceplate (15)**

The following picture shows the register control faceplate which shows the actual register error of length (vertical) and side (horizontal) register.



The scale of the diagram can be changed in the "Configuration 1" screen with the input "resolution factor reg control view".

Register offset length/side register in [mm]

Register deviation length/side register in [mm]

Register controller status

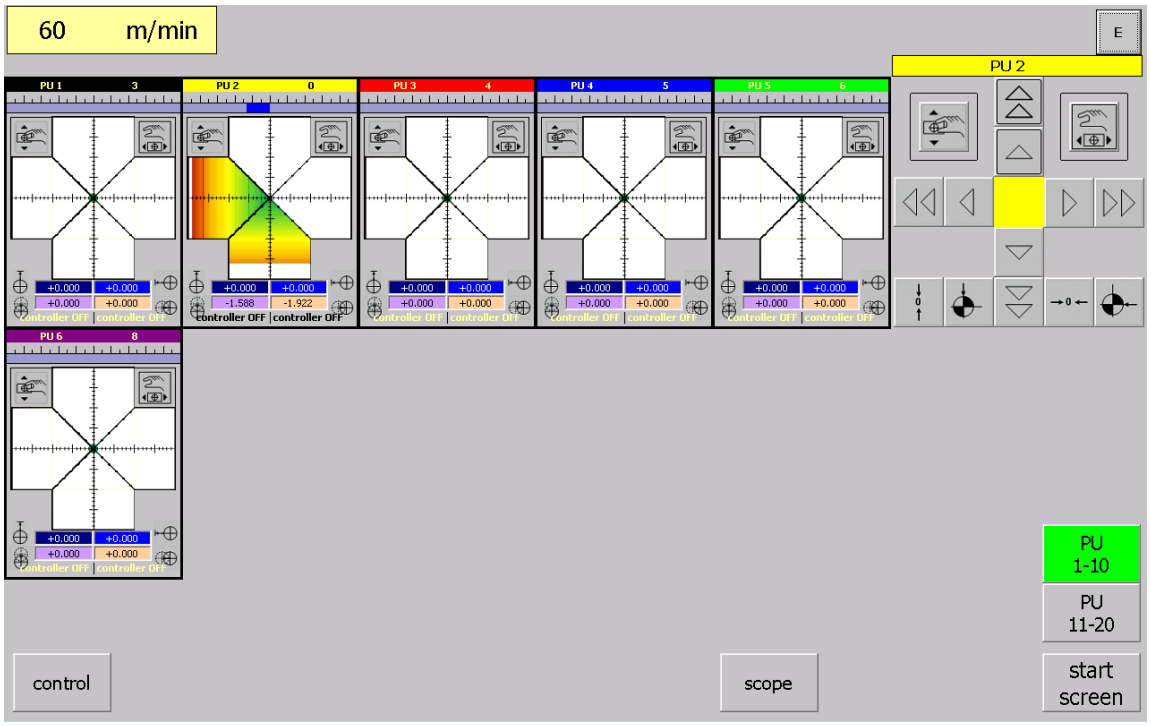
Register overview screen

On the “register overview” screen the operator gets an overview about all register controlled print units in one screen. The screen shows all print units which are currently active.

By clicking inside one of the register control faceplates, the respective print unit gets selected and can be controlled by the buttons on the right hand side.

By clicking on the “scope” button the register oscilloscope screen opens.

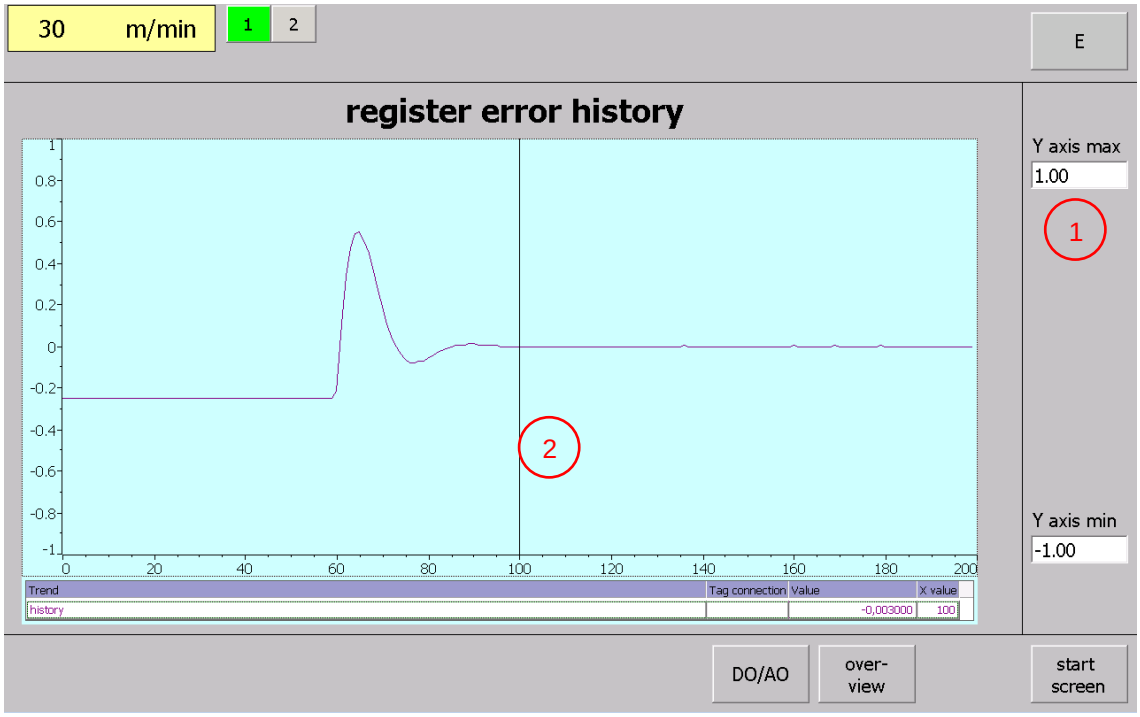
Figure 5-13 Register overview screen



Register error history screen

The “register error history” screen shows an example how statistic functionalities can be integrated to the project. The history picture safes the last 200 register controller error values and show them in the diagram. Because of the free architecture of the program and the screens the user is free to adapt its own statistic functionalities inside as detailed as he likes.

Figure 5-14 Register error history screen



- **y axis range (1)**

  The scaling can be changed by adapting the limits for y-axis.

- **Measuring cursor (2)**

  With the shift register, every single point can be selected, and the register error value will be displayed.

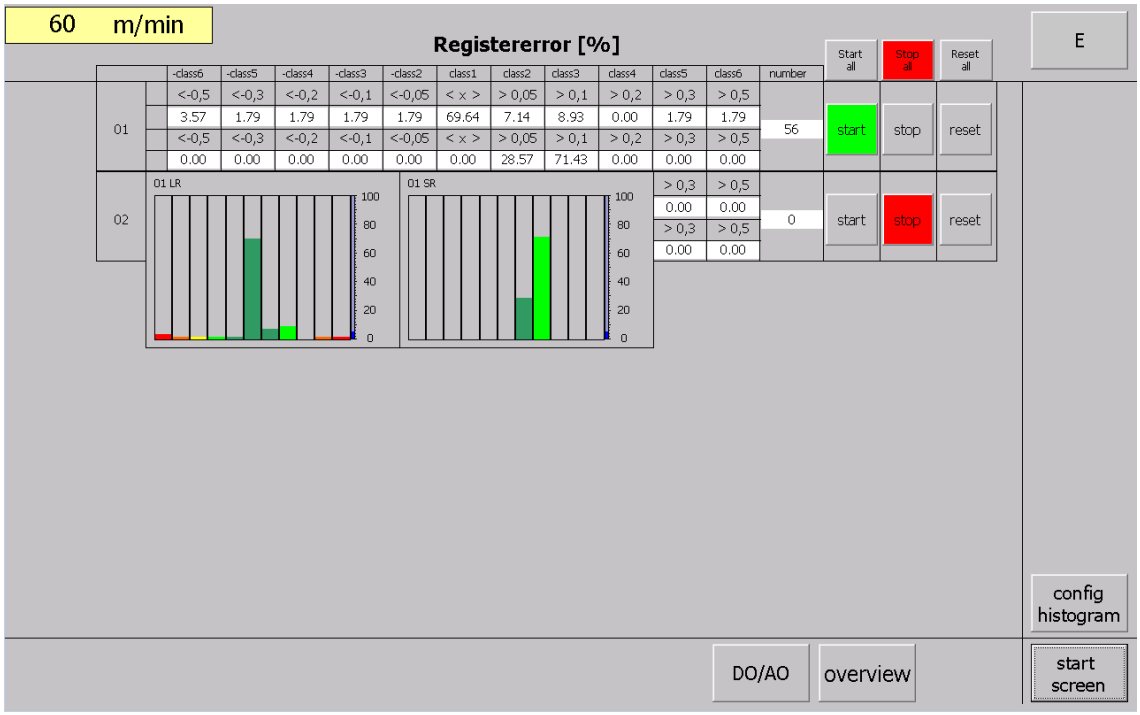
Statistic screen

In the statistic screen the distribution of the register error in different classes can be displayed.

The measurement can be started, stopped and reset for each print unit.

By clicking on the print unit number on the left hand side of the table, the actual chart of the distribution can be displayed.

Figure 5-15 Statistic screen



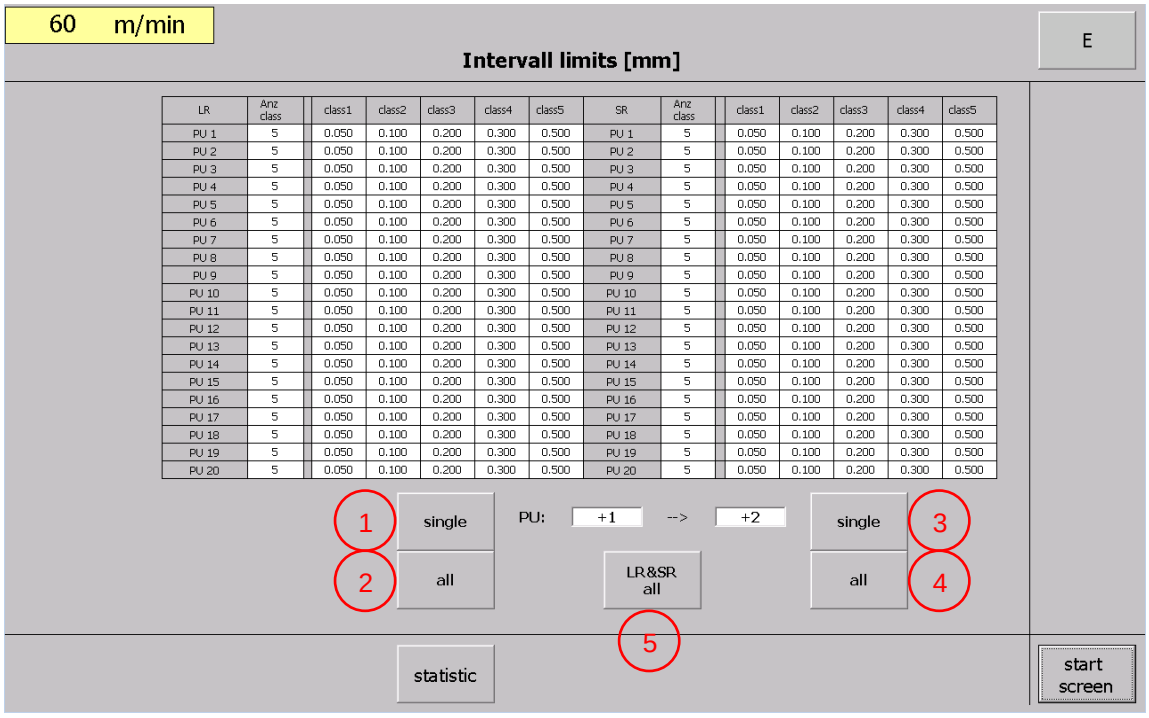
Statistic configuration screen

In this screen, the classes can be defined for each print unit.

e.g.

- Class 1: register error > 0.05 mm
- Class 2: register error > 0.1 mm
- Class 3: register error > 0.2 mm
- Class 4: register error > 0.3 mm
- Class 5: register error > 0.5 mm

Figure 5-16 Statistic Configuration screen



- **Copy single (1) (3)**
  This button copies the class definition form print unit x to y (input fields)
  (1): length register, (3): side register

- **Copy all (2) (4)**
  This button copies the class definition from PU1 to all other PU's
  (2): length register, (4): side register

- **Copy all LR & SR (5)**
  This button copies the class definition from PU1 to all other PU's
  (length and side register)

6 Alarms and error messages

The application differs the following kinds of fault messages:

- Application errors
- Sensor errors
- Register FB errors
- DRD errors
- Insetting errors
- LifeSign error

The error bits and errorID's will be shown in the respective variables in the gasTRCdata[ ].sStdcIO.OUT structure.

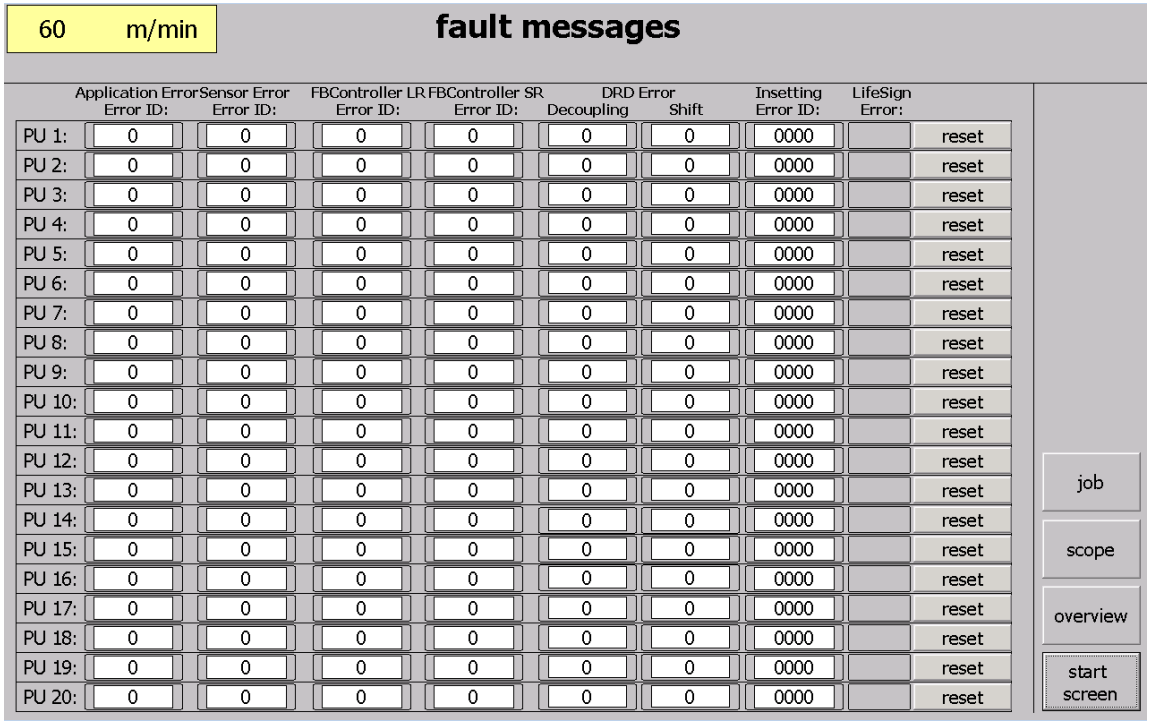
Additionally errorID and time stamp will be stored in an error history.

Actual errors will be shown on the HMI screen with number and text.

Pressing the respective error box, a window appears which shows the error massage in clear text. The error texts of the different errorID's are defined in the WinCC flexible project.

By pressing the "reset" button, all faults related to this print unit are being acknowledged.

Figure 6-1 Alarm screen (HMI)



6.1 Application errors

sStdcIO.out variables:

- boApplicationError
- u16ApplicationErrorID
- i32Parameter1 (“Para1”) = function/parameter result
- i32Parameter2 (“Para2”) = parameter number

Table 6-1 shows FBTRC1000Backgr internal errors. (HMI: Application ErrorID)
The ErrorID corresponds to the FBTRC1000Backgr variable “u16StepID”.

Table 6-1 Application error messages

ErrorID	Description
3020	Error during parameter update. (“Para 1” = function result; “Para 2” = Parameter-Number)
3022	Error during read sensor config
3023	Error during read sensor status
3024	Error during sensor reset
3025	Error during change command distance (WebWeb)
3026	Error during change gate width
3027	Error during read sensor error
3030	Error during center gate
3031	Error during change gate position
3034	Error during set head 1 trigger value
3035	Error during set head 2 trigger value
3036	Error during sensor teaching (trigger value teaching)
3040	Error during TRC calibration
3041	Error during set fiber optic length data set
3042	Error during read fiber optic length data set

Table 6-2 shows the errorID of the system function _writedDriveMultiParameter displayed in i32Parameter1 (HMI: “Para 1”).

The function FCGetWriteDriveMPError of the LTRC1000 library reads out the errorID of the system function in “Para 1” (and if available, the parameter number which causes the error in “Para 2”).

If “Para 2” is 0, a function internal error occurred (function result).

Table 6-2 function/parameter result of system function _writeDriveMultiParameter

Para1	Description
function result = function internal error	
16#00007001	Must be repeated in the next program cycle. Initial call, initiation of writing of parameters okay (only when command is issued asynchronously).
16#00007002	Must be repeated in the next program cycle. Intermediate call, writing of parameters is still active (only when command is issued asynchronously).
16#00007003	Writing of parameters aborted.
16#FFFF8110	Internal error, job aborted. The sum of the lengths specified by the user exceeds the maximum transferable length.
16#FFFF8111	Internal error, job aborted. The length specified by the user does not match the length calculated from the data type and number of elements in the case of at least one parameter.
16#FFFF8112	Internal error, job aborted. A valid format could not be found for at least one parameter.
16#FFFF8190	Internal error, job aborted. Specified logical base address invalid: No assignment is available in SDBs, or there is no base address.
16#FFFF8191	Internal error, job aborted. The _writeDriveMultiParameter function cannot reach the specified logical base address.
16#FFFF8192	Internal error, job aborted. Error in response identifier.
16#FFFF8193	Internal error, job aborted. The number of parameters to be read is not permissible.
16#FFFF819D	Internal error, job aborted. An I slave/I device interface is unable to issue a parameter job to the higher-level master/controller.
16#FFFF819E	Internal error, job aborted. Attempt to abort a non-active function.
16#FFFF819F	Internal error, job aborted. Function not executable.
16#FFFF81C3	Error, can be repeated in the next program cycle. Required resources are presently occupied: - In the _writeDriveMultiParameter function - In the module
16#FFFF81C5	Error, can be repeated in the next program cycle. Distributed I/O not available.
16#FFFF81C7	Error, can be repeated in the next program cycle. Another parameter job has already been issued to the DP station. - based on the user program - from a system-internal component
16#FFFF81CF	Error, can be repeated in the next program cycle. Another parameter job call is currently active under this 'commandId'.
16#FFFF8290	Internal error, job aborted. Specified logical base address invalid: No assignment is available in SDBs, or there is no base address.
16#FFFF8291	Internal error, job aborted. The _writeDriveMultiParameter function cannot reach the specified logical base address.
16#FFFF82A2	Error during data set transfer, job aborted. Error in Layer2: - Station failure - Timeout

Para1	Description
16#FFFF82A3	Error during data set transfer, job aborted. Error in user interface/user: - Protocol error - Station failure - Timeout
16#FFFF82B0	Error during data set transfer, job aborted. Parameter jobs are not supported by the addressed module.
16#FFFF82B2	Error during data set transfer, job aborted. Module reports access to an invalid slot/subslot.
16#FFFF82B5	Error during data set transfer, can be repeated in the next program cycle. The system function cannot be executed due to an internal operating state of the module.
16#FFFF82B7	Error during data set transfer, job aborted. The job could not be dispatched due to a job error.
16#FFFF82C1	Error during data set transfer, command can be repeated immediately. Data of the preceding write job on the module for the same data set has not yet been processed by the module.
16#FFFF82C2	Error during data set transfer, command can be repeated immediately. The module is currently executing the maximum possible jobs for one CPU.
16#FFFF82C3	Error during data set transfer, command can be repeated immediately. Required resources are presently occupied: - In the _writeRecord function - In the module
16#FFFF82C4	Error during data set transfer, command can be repeated immediately. Communication errors: - Parity error - SW Ready not set - Error in block length administration - Checksum error on CPU side - Checksum error on module side
16#FFFF82C5	Error during data set transfer, can be repeated in the next program cycle. Distributed I/O not available.
16#FFFF82C6	Error during data set transfer, can be repeated in the next program cycle. Data record transfer has been aborted due to priority class abort (restart or background).
parameter result = parameter specific error	
16#FFFF8000	Parameter error, job aborted. Access to a non-existent parameter.
16#FFFF8001	Parameter error, job aborted. Change access to a parameter that cannot be modified.
16#FFFF8002	Parameter error, job aborted. Change access with value outside value limits.
16#FFFF8003	Parameter error, job aborted. Access to a non-existent subindex.
16#FFFF8004	Parameter error, job aborted. Access with subindex to non-indexed parameter (not an array parameter).
16#FFFF8005	Parameter error, job aborted. Change access with value that does not match the parameter data type.
16#FFFF8006	Parameter error, job aborted. Modification access with value other than 0 where this is not allowed.
16#FFFF800B	Parameter error, job aborted. Change access without parameter change rights.
16#FFFF8011	Parameter error, job aborted. Job cannot be executed due to operating state.
16#FFFF8014	Parameter error, job aborted. Modification access with value that is within value limits, but illegal for some other sustainable reason (parameter with defined individual values).
16#FFFF8016	Parameter error, job aborted. Illegal or unsupported value for attribute, number of elements, parameter number, or subindex, or a combination of these.
16#FFFF8017	Parameter error, job aborted. Illegal format.
16#FFFF8018	Parameter error, job aborted. Number of values of parameter data does not match the number of elements in the parameter address.
16#FFFF8019	Parameter error, job aborted. Access to a non-existing axis or an invalid drive object.

Table 6-3 shows the errorID of the system function _readDriveMultiParameter displayed in i32Parameter1 (HMI: “Para 1”).

The function FCGetReadDriveMPError of the LTRC1000 library reads out the errorID of the system function in “Para 1” (and if available, the parameter number which causes the error in “Para 2”).

If “Para 2” is 0, an function internal error occurred (function result).

Table 6-3 function/parameter result of system function _readDriveMultiParameter

Para1	Description
function result = function internal error	
16#00007001	Must be repeated in the next program cycle. Initial call, initiation of reading of parameters okay (only when command is issued asynchronously).
16#00007002	Must be repeated in the next program cycle. Intermediate call, reading of parameter description is still active (only when command is issued asynchronously).
16#00007003	Reading of parameters aborted.
16#FFFF8190	Internal error, job aborted. Specified logical base address invalid: No assignment is available in SDBs, or there is no base address.
16#FFFF8191	Internal error, job aborted. The _readDriveMultiParameter function cannot reach the specified logical base address.
16#FFFF8192	Internal error, job aborted. Error in response identifier.
16#FFFF8193	Internal error, job aborted. The number of parameters to be read is not permissible.
16#FFFF819D	Internal error, job aborted. An I slave/I device interface is unable to issue a parameter job to the higher-level master/controller.
16#FFFF819E	Internal error, job aborted. Attempt to abort a non-active function.
16#FFFF819F	Internal error, job aborted. Function not executable.
16#FFFF81C3	Error, can be repeated in the next program cycle. Required resources are presently occupied: - In the _readDriveMultiParameter function - In the module
16#FFFF81C5	Error, can be repeated in the next program cycle. Distributed I/O not available.
16#FFFF81C7	Error, can be repeated in the next program cycle. Another parameter job has already been issued to the DP station. - based on the user program - from a system-internal component
16#FFFF81CF	Error, can be repeated in the next program cycle. Another parameter job call is currently active under this 'commandId'.
16#FFFF8290	Internal error, job aborted. Specified logical base address invalid: No assignment is available in SDBs, or there is no base address.
16#FFFF8291	Internal error, job aborted. The _readDriveMultiParameter function cannot reach the specified logical base address.
16#FFFF82A2	Error during data set transfer, job aborted. Error in Layer2: - Station failure – Timeout
16#FFFF82A3	Error during data set transfer, job aborted. Error in user interface/user: - Protocol error - Station failure - Timeout
16#FFFF82B0	Error during data set transfer, job aborted. Parameter jobs are not supported by the addressed module.
16#FFFF82B2	Error during data set transfer, job aborted. Module reports access to an invalid slot/subslot.

Para1	Description
16#FFFF82B5	Error during data set transfer, can be repeated in the next program cycle. The system function cannot be executed due to an internal operating state of the module.
16#FFFF82B7	Error during data set transfer, job aborted. The job could not be dispatched due to a job error.
16#FFFF82C0	Error during data set transfer, can be repeated in the next program cycle. The module is feeding the data record, but the module does not yet have any read data.
16#FFFF82C2	Error during data set transfer, command can be repeated immediately. The module is currently executing the maximum possible jobs for one CPU.
16#FFFF82C3	Error during data set transfer, command can be repeated immediately. Required resources are presently occupied: - In the _readRecord function - In the module
16#FFFF82C4	Error during data set transfer, command can be repeated immediately. Communication errors: - Parity error - SW Ready not set - Error in block length administration - Checksum error on CPU side - Checksum error on module side
16#FFFF82C5	Error during data set transfer, Can be repeated in the next program cycle. Distributed I/O not available.
16#FFFF82C5	Error during data set transfer, Can be repeated in the next program cycle. Distributed I/O not available.
parameter result = parameter specific error	
16#FFFF8000	Parameter error, job aborted. Access to a non-existent parameter.
16#FFFF8003	Parameter error, job aborted. Access to a non-existent subindex.
16#FFFF8004	Parameter error, job aborted. Access with subindex to non-indexed parameter (not an array parameter).
16#FFFF800B	Parameter error, job aborted. Change access without parameter change rights.
16#FFFF8015	Parameter error, job aborted. The length of the current response exceeds the maximum transferable length.
16#FFFF8016	Parameter error, job aborted. Illegal or unsupported value for attribute, number of elements, parameter number, or subindex, or a combination of these.
16#FFFF8017	Parameter error, job aborted. Illegal or unsupported parameter format.
16#FFFF8019	Parameter error, job aborted. Access to a non-existing axis or an invalid drive object.

Table 6-4 shows errors of the system function block _writeVariableRecord. (HMI: Application ErrorID)

The function FCGetWriteDOError of the LTRC1000 library generates the decimal errorID dependent on the hexadecimal errorID of the system function block.

Table 6-4 System function block _writeVariableRecord errors

ErrorID	Description
2300	16#FFFF8090 Specified logical base address invalid.
2301	16#FFFF8091 Specified logical base address cannot be reached.
2302	16#FFFF809C internal collision LDPV1
2303	16#FFFF809D An I slave / I device interface cannot read any data sets.
2304	16#FFFF809E Error, job aborted. Attempt to abort a non-active function.
2305	16#FFFF809F Error, job aborted. Function not executable.
2306	16#FFFF80A1 Negative acknowledgment when writing to the module.
2307	16#FFFF80A2 Protocol error in Layer2: - Module not available
2308	16#FFFF80A3 Protocol error involving user interface/user: - Module not available
2309	16#FFFF80A8 Error because of version conflict
2310	16#FFFF80A9 Function not supported by the module.
2311	16#FFFF80B0 Data record unknown to module.
2312	16#FFFF80B1 Incorrect length specified in 'LEN' parameter.
2313	16#FFFF80B2 Module reports access to an invalid slot/subslot.
2314	16#FFFF80B3 Module reports type conflict.
2315	16#FFFF80B4 Module reports access to an invalid area.
2316	16#FFFF80B5 Module is not ready.
2317	16#FFFF80B6 Module rejects access.
2318	16#FFFF80B7 Module reports an illegal range for a parameter or value.
2319	16#FFFF80B8 Module reports an invalid parameter.
2320	16#FFFF80B9 Module reports an invalid type.
2321	16#FFFF80C1 Data of the preceding write job on the module for the same data set has not yet been processed by the module.
2322	16#FFFF80C2 The module is currently processing the possible maximum of jobs.
2323	16#FFFF80C3 Required resources are presently occupied.
2324	16#FFFF80C4 Communication error
2325	16#FFFF80C5 Distributed I/O not available.
2326	16#FFFF80C6 Data set transfer has been aborted because of priority class abort.
2327	16#FFFF80C7 The function block is currently processing another job.

Table 6-5 shows errors of the system function block _readVariableRecord. (HMI: Application ErrorID)

The function FCGetReadDOError of the LTRC1000 library generates the decimal errorID dependent on the hexadecimal errorID of the system function block.

Table 6-5 System function block _readVariableRecord errors

ErrorID	Description
2350	16#FFFF8090 Specified logical base address invalid.
2351	16#FFFF8091 Specified logical base address cannot be reached.
2352	16#FFFF809B The target memory provided is not large enough.
2353	16#FFFF809C Internal temporary collision of DPV1 jobs.
2354	16#FFFF809D An I slave / I device interface cannot read any data sets.
2355	16#FFFF809E Error, job aborted. Attempt to abort a non-active function.
2356	16#FFFF809F Error, job aborted. Function not executable.
2357	16#FFFF80A0 Negative acknowledgement when reading from the module
2358	16#FFFF80A2 Error during data set transfer, job aborted. Protocol error in Layer2: - Module not available
2360	16#FFFF80A8 Error because of version conflict
2361	16#FFFF80A9 Function not supported by the module.
2362	16#FFFF80B0 Data record unknown to module.
2363	16#FFFF80B1 Incorrect length specified in 'dataLength' parameter.
2364	16#FFFF80B2 Module reports access to an invalid slot/subslot.
2365	16#FFFF80B3 Module reports type conflict.
2366	16#FFFF80B4 Module reports access to an invalid area.
2367	16#FFFF80B5 Module is not ready.
2368	16#FFFF80B6 Module rejects access.
2369	16#FFFF80B7 Module reports an illegal range for a parameter or value.
2370	16#FFFF80B8 Module reports an invalid parameter.
2371	16#FFFF80B9 Module reports an invalid type.
2372	16#FFFF80C0 The module has the data set, but no read data is available.
2373	16#FFFF80C2 The module is currently processing the possible maximum of jobs.
2374	16#FFFF80C3 Required resources are presently occupied.
2375	16#FFFF80C4 Communication error
2376	16#FFFF80C5 Distributed I/O not available.
2377	16#FFFF80C6 Data set transfer has been aborted because of priority class abort.
2378	16#FFFF80C7 The function block is currently processing another job.

6.2 Sensor error messages

sStdcIO.out variables:

- boSensorError
- u16SensorErrorID

Table 6-6 TRC1000 error messages

ErrorID	Error code (Sensor internal)	Description
2010	0x0004 (4)	Address error (load or command call)
2011	0x0005 (5)	Address error (store)
2012	0x0006 (6)	Bus error (command call)
2013	0x0007 (7)	Bus error (load or store)
2014	0x0008 (8)	System call
2015	0x000A (10)	Reserved instruction
2016	0x000B (11)	Coprocessor unusable
2017	0x000C (12)	Arithmetic overflow
2018	0x000D (13)	Trap (e.g. division by 0)
2019	0x0103 (259)	R/W parameter command unknown
2020	0x0104 (260)	R/W parameter command buffer overflow
2021	0x0105 (261)	R/W parameter command system error
2022	0x0106 (262)	I/O data loss of Controller life sign
2023	0x0107 (263)	I/O data timeout
2024	0x0108 (264)	I/O data read error netX
2025	0x0109 (265)	I/O data write error netX
2026	0x0200 (512)	AGS speed limit or standstill
2027	0x0201 (513)	AGS block mark not detected
2028	0x0202 (514)	Sensor head 1 Sensor Automatic, error sensor teaching (Check P45 and P46 unequal 0!)
2029	0x0203 (515)	Sensor head 2 Sensor Automatic, error sensor teaching (Check P45 and P46 unequal 0!, selected printing mark within the gate)
2030	0x0300 (768)	Sensor head 1 input voltage 0V
2031	0x0301 (769)	Sensor head 2 input voltage 0V
2032	0x0302 (770)	Sensor head 1 calibration threshold value exceeded
2033	0x0303 (771)	Sensor head 2 calibration threshold value exceeded

For the conversion from the sensor internal error code to the (user-) ErrorID the function “FCGetP947Error” is used.

6.3 Register FB error messages

sStdcIO.out variables:

- boRegisterFBErrorLR
- boRegisterFBErrorSR
- u16RegisterFBErrorIDLR
- u16RegisterFBErrorIDSR

Table 6-7 Register FB error messages

ErrorID	Description
2050	T_A out of range
2051	sPu.v_setpoint > CONST_V_MAX
2052	sPu.s_web_Length out of range
2053	T_mot out of range
2054	sPu.s_format out of range
2055	r32TIWebWeb out of range
2056	transport delay error
2057	input or output filter error
2058	controller disabled: amplitude x1_k exceeds bounds
2059	controller disabled: amplitude x3_k exceeds bounds
2060	Prediction output invalid x4
2061	fast rate differentiation of predictor output error
2062	paramError OR oFiltOutputLimited OR iFiltOutputLimited
2063	Output filter invalid value x6
2064	warning integrator overflow
2065	warning buffer size (transport delay) → check web speed <> 0, check distance PU <-> Sensor
2066	warning negative time (transport delay)
2067	warning initial ramp up
2068	output filter k_outLtd: x5 is out of range and was limited.
2069	output P-controller k_intLtd: x_int out of range and was limited
2070	requested ramp time is active after controller enable
2071	predictEnabled

6.4 Insetting/DRD error messages

NOTE A list of all error and warning messages can be found in the respective Add-On documentation.

6.5 LifeSign error

sStdcIO.out variable:

- boLifeSignError

In the sensor control word and status word there are respectively 4 bits (bit 12..15) to observe the communication between sensor and controller (SIMOTION). This is realized by a counter (Sing-Of-Life Counter).

In case of communication time out the error bit boLifeSignError in the gasTRCdata[ ].sStdcIO.OUT structure gets TRUE. In the error screen the life sign error box becomes red.

6.6 Error history and fault buffer

Sensor internal a fault buffer which contains up to 8 **sensor messages** is integrated. If more than 8 messages occur without acknowledge, the last message will be overwritten, so that the first 7 messages are preserved.

An error message consists of fault number (P947) and corresponding fault time (P948). The fault message counter (P944) displays the number of messages occurred and is increased with each error message.

After acknowledge the fault buffer will be erased (the error counter remains on it's old value). To reset the error counter a factory reset is necessary.

Additionally these sensor messages will be stored in a SIMOTION internal error history in the gasTRCdata[ ].sStdcIO.OUT structure which is constructed as a circular buffer**.** This error history persists after fault acknowledge in the sensor.

The error history (sErrorHistory) consists of:

- errorID (au16ErrorNumber_P947)
- time stamp the error occurred (adtTimeStamp) (SIMOTION time stamp)

7 Sensor calibration

Before using the TRC for printing mark detection, the correct data set for the fiber optic cable length has to be selected.

In some cases a sensor white-adjustment could be also useful.

Both settings can be done by the HMI or alternatively via the RS232 command interface.

7.1 Fiber optic length

The fiber optic cable between head and sensor device is available in two lengths:

- 2.5m
- 5.0m

For these two lengths the sensor is delivered with two data sets.

NOTE In the factory settings, data set 1 (2.5m) is selected.

For a 2.5m fiber optic cable length, no data set change is necessary!

Using a 5.0m fiber optic cable, data set 2 has to be selected.

For IDS (head 1) and DS (head 2) different data sets can be selected.

After changing the data set, the new data set is taken over and saved permanently in the flash memory of the device. At next power up the previously selected data set is loaded.

NOTE A factory reset of the sensor resets both data sets (IDS and DS) to data set 1 (2.5m)

7.1.1 Data set change via HMI

1) Go to the config data screen

2) Select the sensor (TRC #) you want to change the data sets

3) Select the cable length for head 1 and head 2 (even if head 2 is not existent, select 2.5m)

4) Press the “OK” button

The actual selected data set is displayed in the boxes after selecting the respectively TRC number.

NOTE Together with the dataset for the fiber optic cable length, the data set for the white adjustment is sent to the sensor.

See also chapter 7.2

7.1.2 Data set change via RS232 interface

NOTE To use the RS232 command interface, see chapter 9.

The command to change the data set is **6010**. Additionally to the command you have to enter the data set number to change the data set of the IDS and DS. Possible value entries are:

0 → 2.5m data set

1 → 5.0m data set

Hyper Terminal command to change the data set:

6010 <DS_value><IDS_value>

Example:

6010 01 (meaning: DS 2.5m; IDS 5.0m)

The actual selected data set is stored in P150 and can be read out via the RS232 interface (command **1000 150**). The result is a hexadecimal value: **00000001h**

7.2 White-adjustment

The white adjustment is used to calibrate the voltage evaluation of the sensor in respect to different colors and fiber optic cables.

NOTE The factory setting of each sensor is calibrated to white background.

It is recommended to repeat the calibration for each sensor together with respective fiber optic cable before use.

NOTE With sensor firmware V2.2.0 and Print Standard Add-On TRC1000 version V3.3.0 it's possible to calibrate the sensor to mirroring background also.

NOTICE **Before starting the calibration the correct cabel length data set as well as the data set for white or mirroring background need to be selected!**

To perform the sensor adjustment a special gauge is necessary.

The calibration itself consists of three steps:

- White adjustment
- Black adjustment
- White adjustment (once more)

7.2.1 Sensor adjustment via HMI

1) Go to the config data screen
2) Go to the sensor calibration screen
3) Select the sensor (TRC #) you want to adjust
4) Select the head you want to adjust
5) Press the "Start calibration" button
6) The TRC calibration state shows the actual state of the calibration

   Follow the instructions!

   Press the "Next" button not before the sensor head sticks in the sensor gauge!

7.2.2 Sensor adjustment via RS232 interface

NOTE To use the RS232 command interface, see chapter 9.

To adjust the sensor, three commands per sensor head are necessary:

Table 7-1 sensor adjust commands

Steps	RS232 command	
	IDS	DS
(1) White adjustment	6004	7004
(2) Black adjustment	6005	7005
(3) White adjustment (once more)	6004	7004

(1) Insert the objective above the white substrate up to the limit and hold it there.

Send command 6004 for automatic white adjustment.

(2) Insert the objective above the black substrate up to the limit and hold it there.

Send command 6005 for automatic black adjustment.

(3) Insert the objective above the white substrate up to the limit again and hold it there.

Send command 6004 for automatic white adjustment again.

NOTE The procedure for the DS is the same as described above, but with command 7004 for white and 7005 for black adjustment.

8 Command interface (RS232)

With the RS232 interface of the sensor, all parameter within the acyclic data can be read and written.

To connect your computer with the sensor the following hardware and software is necessary:

- RS232 cable
- RS232 Port (computer) or USB to RS232 adapter
- Windows Hyper Terminal or “Tera Term” (free-ware tool)

For a detailed description how to start the communication between computer and sensor device, see Wiedeg IDS-PN User Manual or Wiedeg IDS-PN Reference Manual.

Read and write commands of the RS232 interface:

Table 8-1 TRC1000 RS232 read/write commands

Command/Parameter	Description
1000 <parameter number>	Parameter read (PNU ≥ 100)
1002 <parameter number>	Parameter read (PNU < 100)
1003 <parameter number> <value>	Parameter write (PNU < 100)
1004 <parameter number> <value>	Parameter write (PNU ≥ 100)

For a more detailed list of all possible parameter, see Wiedeg IDS-PN User Manual or Wiedeg IDS-PN Reference Manual.

9 Analog monitor (DIAG interface)

With the analog monitor various actual values of the process respectively logical states of one or more bits of a status value can be read out as analog values with a refresh rate of 0.5 ms. All parameter with a parameter number PNU > 100 can be read out that way.

Necessary hardware/software:

- Measuring adapter (connect to the DIAG interface of the sensor)
- Oscilloscope
- Hard- and software for the RS232 interface to parameterize the analog monitor (see chapter 9)

The measuring adapter has two analog pins, so it's possible two read out two parameters at the same time. The parameterization which parameter should be displayed on the analog monitor output pins has to be done by the RS232 command interface. The commands are listed in the following table:

Table 9-1 TRC1000 RS232 commands to parameterize the analog monitor

Command/Parameter	Description
5001 <channel> <parameter number> <reference> <offset>	Parameterize analog monitor channel: 1 or 2 (two measuring pins on the measuring adapter) PNU: ≥ 100 reference: value referenced to 10V offset: DC offset with the corresponding unit of the value Example: 5001 1 312 0.001 0
5003	Read out actual analog monitor settings

10 Firmware update

There are two possibilities to update the sensor firmware:

1) Firmware update by the manufacturer Wiedeg

2) Do-it-your-self way with the (German) manual “IDS-PN Firmware Update PIC32 mit ICD 3”

NOTICE	**The second way (do-it-your-self) is recommended for experienced persons only!** **Furthermore there is hardware necessary which is obtainable by the manufacturer (Wiedeg) only.**

11 Sensor reset

Two reset-modes are available:

- soft reset
  - Reboot of the device.
  - All settings remain unchanged

- factory reset
  - Reset factory settings
    - E.g. sensor name, fiber optic data set, etc.

11.1.1 Sensor reset via HMI

1) Go to the “Config 1” screen
2) Select the sensor (TRC #) you want to reset
3) Select the reset mode (factory reset, soft reset)
4) Press the “sensor reset” button

NOTICE	**It’s recommended to perform a reboot (soft reset) of the TRC device after a factory reset!**

11.1.2 Sensor reset via RS232 interface

NOTE	To use the RS232 command interface, see chapter 9.

Table 11-1 sensor reset commands

	RS232 commands	
	IDS	**DS**
soft reset	9999	
factory reset	8888	7777

NOTICE	**It’s recommended to perform a reboot (soft reset) of the TRC device after a factory reset!**

12 Abbreviations

Table 12-1 Abbreviations

Abbreviation	Description
FOC	Fiber optic cable
TRC	Technology module Register Control
PN	PROFINET
AMR	Automatic mark recognition (vgl. AGS)
AGS	Automatic Gate Setting
LR	Length register
SR	Side register
TO	SIMOTION Technology object
HW	Hardware
ATEX	**At**mosphère **Ex**plosive (ATEX guideline)
IRT	Isochronous-Real-Time
RGB	Red, Green, Blue color model
FW	Firmware
PNU	Parameter number
IDS	Intelligent detection sensor (Head 1)
DS	Detection Sensor (Head 2)
OCX	“OLE custom controls” is a TCP/IP client which connects to the SIMOTION (TCP/IP server)

13 Related literature

13.1 Bibliography

This list is not complete and only represents a selection of relevant literature.

Table 13-1 Related literature

	Subject	Title
1	Wiedeg IDS-PN User Manual	IDS-PN_User_Manual_en_V2.0
2	Wiedeg IDS-PN Reference Manual	IDS-PN_Reference_Manual_de_V2.0 (more detailed user manual, only German version available!)
3	Wiedeg firmware update manual	IDS-PN Firmware Update PIC32 mit ICD 3 (only German version available)
4	SIMOTION Print Standard	Print Standard V2210 en.pdf
5	Insetting	Print Standard AddOn Insetting
6	DRD	Print Standard AddOn DRD
7	SIMOTION	SIMOTION Manual 02/2012
8	SINAMICS	SINAMICS Manual 06/2012

13.2 Internet link specifications

This list is not complete and only represents a selection of relevant information.

Table 13-2

	Subject	Title
1	Reference to the entry	http://support.automation.siemens.com/WW/view/en/EntryID
2	Siemens Industry Online Support	http://support.automation.siemens.com
3	Siemens Industry Printing	http://www.siemens.com/printing
4	Siemens Industry APC	http://www.siemens.com/motioncontrol/apc
5	SIMOTION	http://www.siemens.com/simotion
6	SINAMICS	http://www.siemens.com/sinamics

14 Document-Version History

Table 14-1 Version History

Date	AddOn Version	PrintStandard Version	Change
08/2012	V2.2.1.2.1	V2.2.1	Preliminary version
27.06.2013	V2.2.1.2.2	V2.2.1	First official version
	As of now: New versioning!		
28.02.2014	V3.0.0	V2.2.1	HMI Control, Updates
26.01.2015	V3.1.0	V2.2.1	corrections and updates
20.04.2015	V3.2.0	V2.2.1 / V3.1.0	corrections and updates
10.08.2015	V3.3.0	V3.1.1	Reverse markfield; Graphical adaptions to the HMI screens; Calibration to mirroring background;

15 Contact

Application Center

SIEMENS

Siemens AG
DF FA PMA APC
Frauenauracher Str. 80
91056 Erlangen
Fax: 09131-98-1297
mailto: tech.team.motioncontrol@siemens.com