

Learn-/Training Document

Siemens Automation Cooperates with Education (SCE) | From Version V14 SP1

TIA Portal Module 031-100 Basics of FC Programming with SIMATIC S7-1200

siemens.com/sce



Matching SCE Trainer Packages for these Learn-/Training Document

- SIMATIC S7-1200 AC/DC/RELAY (set of 6) "TIA Portal" Order no.: 6ES7214-1BE30-4AB3
- SIMATIC S7-1200 DC/DC/DC (set of 6) "TIA Portal" Order no.: 6ES7214-1AE30-4AB3
- Upgrade SIMATIC STEP 7 BASIC V14 SP1 (for S7-1200) (set of 6) "TIA Portal" Order no.: 6ES7822-0AA04-4YE5

Note that these trainer packages are replaced with successor packages when necessary. An overview of the currently available SCE packages is available at: <u>siemens.com/sce/tp</u>

Continued training

For regional Siemens SCE continued training, get in touch with your regional SCE contact siemens.com/sce/contact

Additional information regarding SCE

siemens.com/sce

Information regarding use

The SCE Learn-/Training Document for the integrated automation solution Totally Integrated Automation (TIA) was prepared for the program "Siemens Automation Cooperates with Education (SCE)" specifically for training purposes for public educational facilities and R&D institutions. Siemens AG does not guarantee the contents.

This document is to be used only for initial training on Siemens products/systems, which means it can be copied in whole or part and given to those being trained for use within the scope of their training. Circulation or copying this Learn-/Training Document and sharing its content is permitted within public training and advanced training facilities for training purposes.

Exceptions require written consent from the Siemens AG contact person: Roland Scheuerer roland.scheuerer@siemens.com.

Offenders will be held liable. All rights including translation are reserved, particularly if a patent is granted or a utility model or design is registered.

Use for industrial customer courses is explicitly not permitted. We do not consent to commercial use of the Learn-/Training Document.

We wish to thank the TU Dresden, particularly Prof. Dr.-Ing. Leon Urbas and the Michael Dziallas Engineering Corporation and all other involved persons for their support during the preparation of this Learn-/Training Document.

Table of contents

1	Goa	al	5
2	Prer	requisite	5
3	Req	uired hardware and software	6
4	The	ory	7
	4.1	Operating system and application program	7
	4.2	Organization blocks	8
	4.3	Process image and cyclic program processing	9
	4.4	Functions1	1
	4.5	Function blocks and instance data blocks1	2
	4.6	Global data blocks1	3
	4.7	Library-compatible logic blocks1	4
	4.8	Programming languages1	5
5	Tas	k1	6
6	Plar	nning1	6
	6.1	EMERGENCY STOP1	6
	6.2	Manual mode – Conveyor motor in manual mode1	6
	6.3	Technology diagram1	7
	6.4	Reference list1	8
7	Stru	ctured step-by-step instructions1	9
	7.1	Retrieve an existing project1	9
	7.2	Create a new tag table2	20
	7.3	Create new tags within a tag table2	2
	7.4	Import "Tag_table_sorting_station"2	3
	7.5	Create function FC1 "MOTOR_MANUAL" for the conveyor motor in manual mode2	7
	7.6	Define the Interface of function FC1 "MOTOR_MANUAL"2	9
	7.7	Program FC1: MOTOR_MANUAL	2
	7.8	Program the organization block OB1 – Control conveyor tracking forwards in manual mode3	9
	7.9	Save and compile the program4	4
	7.10	Download the program4	5
	7.11	Monitor program blocks4	6
	7.12	Archive the project4	8
	7.13	Checklist4	9

8	Exe	rcise	50
	8.1	Task – Exercise	.50
	8.2	Technology diagram	.50
	8.3	Reference list	.51
	8.4	Planning	.51
	8.5	Checklist – Exercise	.52
9	Add	itional information	.53

Basics of FC Programming

1 Goal

In this chapter, you will get to know the basic elements of a control program – the **organization blocks (OBs), functions (FCs)**, **function blocks (FBs)** and **data blocks (DBs)**. In addition, you will be introduced to **library-compatible** function und function block programming. You will get to know the **Function Block Diagram (FBD)** programming language and use it to program a function (FC1) and an organization block (OB1).

The SIMATIC S7 controllers listed in Chapter 3 can be used.

2 Prerequisite

This chapter builds on the hardware configuration of SIMATIC S7 CPU1214C. However, other hardware configurations that have digital input and output cards can be used. For this chapter, you can use the following project, for example:

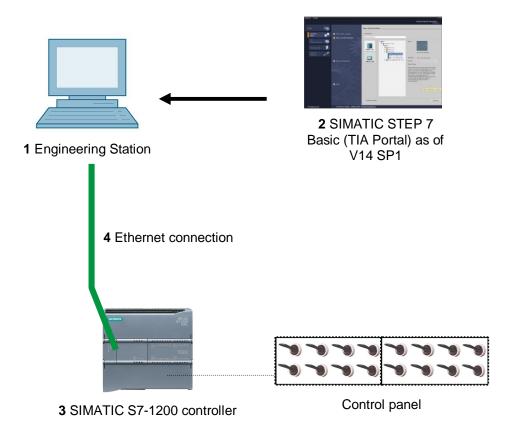
SCE_EN_011_101_Hardware_Configuration_CPU1214C.zap14

3 Required hardware and software

- 1 Engineering station: requirements include hardware and operating system (for additional information, see Readme on the TIA Portal Installation DVDs)
- 2 SIMATIC STEP 7 Basic software in TIA Portal as of V14 SP1
- 3 SIMATIC S7-1200 controller, e.g. CPU 1214C DC/DC/DC with ANALOG OUTPUT SB1232 signal board, 1 AO Firmware as of V4.2.1

Note: The digital inputs should be fed out to a control panel.

4 Ethernet connection between engineering station and controller



4 Theory

4.1 Operating system and application program

Every controller (CPU) contains an **operating system**, which organizes all functions and sequences of the CPU that are not associated with a specific control task. The tasks of the operating system include the following:

- Performing a warm restart
- Updating the process image of the inputs and output
- Cyclically calling the user program
- Detecting interrupts and calling interrupt OBs
- Detecting and handling errors
- Managing memory areas

The operating system is an integral component of the CPU and comes pre-installed.

The **user program** contains all functions that are necessary for executing your specific automation task. The tasks of the user program include the following:

- Checking the basic requirements for a warm restart using startup OBs
- Processing of process data, i.e. activation of output signals as a function of the input signal states
- Reaction to interrupts and interrupt inputs
- Error handling during normal program execution

4.2 Organization blocks

Organization blocks (OBs) form the interface between the operating system of the controller (CPU) and the application program. They are called from the operating system and control the following operations:

- Cyclic program processing (e.g. OB1)
- Startup characteristics of the controller
- Interrupt-driven program processing
- Error handling

A project must have, at a minimum, *an organization block for cyclic program processing*. An OB is called by a *start event* as shown in Figure 1. In addition, the individual OBs have defined priorities so that, for example, an OB82 for error handling can interrupt the cyclic OB1.

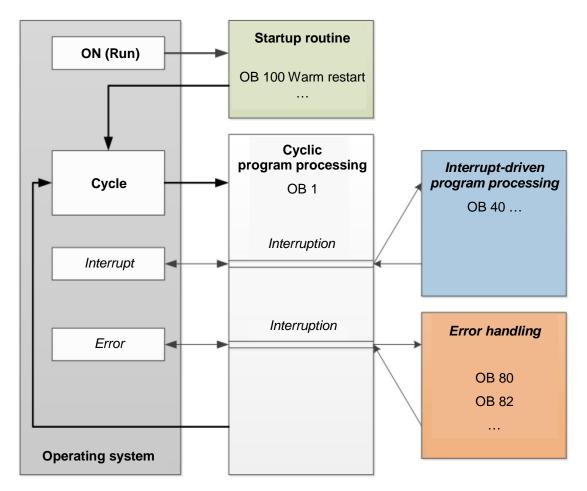


Figure 1: Start events in the operating system and OB call

When a start event occurs, the following reactions are possible:

- If an OB has been assigned to the event, this event triggers the execution of the assigned OB. If the priority of the assigned OB is greater than the priority of the OB that is currently being executed, it is executed immediately (interrupt). If not, the assigned OB waits until the higher-priority OB has been completely executed
- If you have not assigned an OB to the event, the default system reaction is performed.

Table 1 shows examples for various start events for a SIMATIC S7-1200. Also shown are the possible OB number(s) and the default system reactions that occur when the respective organization block (OB) is not present in the controller.

Start event	Possible OB numbers	Default system reaction
Startup	100, ³ 123	Ignore
Cyclic program	1, ³ 123	Ignore
Time-of-day interrupt	10 to 11	-
Update interrupt	56	Ignore
Scan cycle monitoring time exceeded once	80	Ignore
Scan cycle monitoring time exceeded twice	80	STOP
Diagnostic interrupt	82	Ignore

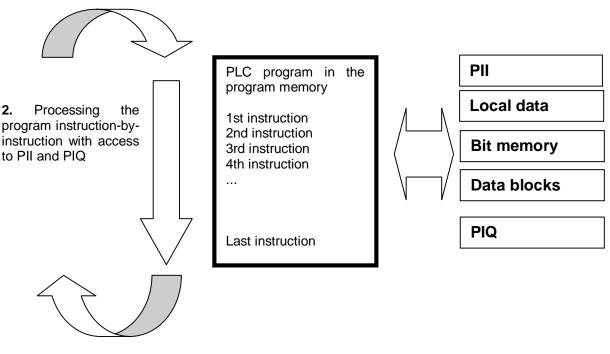
Table 1: OB numbers for various start events

4.3 Process image and cyclic program processing

When the cyclic user program addresses the inputs (I) and outputs (O), it does not query the signal states directly from the input/output modules. Instead, it accesses a memory area of the CPU. This memory area contains an image of the signal states and is called the **process image**.

The cyclic program processing sequence is as follows:

- At the start of the cyclic program, a query is sent to determine whether or not the individual inputs are energized. This status of the inputs is stored in the process image of the inputs (PII). In doing so, the information 1 or "High" is stored for energized inputs and the information 0 or "Low" for de-energized inputs.
- The CPU then executes the program stored in the cyclic organization block. For the required input information, the CPU accesses the previously read process image of the inputs (PII) and the results of logic operation (RLOs) are written to a so-called process image of the outputs (PIQ).
- 3. At the end of the cycle, the **process image of the outputs** (**PIQ**) is transferred as the signal state to the output modules and these are energized or de-energized. The sequence then continues again with Item 1.
 - 1. Save status of inputs in PII.



3. Transfer status from the PIQ to the outputs.

Figure 2: Cyclic program processing

Note: The time the CPU needs for this sequence is called cycle time. This depends, in turn, on the number and type of instructions and the processor performance of the controller.

4.4 Functions

Functions (FCs) are logic blocks without memory. They **have no data memory** in which values of block parameters can be stored. Therefore, all interface parameters must be connected when a function is called. To store data permanently, global data blocks must be created beforehand.

A function contains a program that is executed whenever the function is called from another logic block.

Functions can be used, for example, for the following purposes:

- Math functions that return a result dependent on input values.
- Technological functions such as individual controls with binary logic operations.

A function can also be called several times at different points within a program.

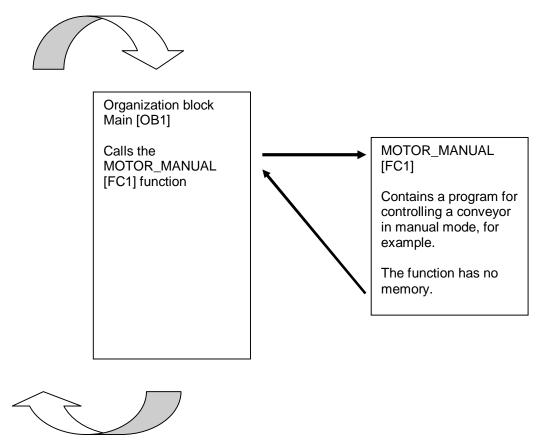


Figure 3: Function with call from organization block Main [OB1]

4.5 Function blocks and instance data blocks

Function blocks are logic blocks that store their input, output and in-out tags as well as static tags permanently in instance data blocks, so that they **are available after the block has been executed**. For this reason, they are also referred to as blocks with "memory".

Function blocks can also operate with temporary tags. Temporary tags are not stored in the instance DB, however. Instead, they are only available for one cycle.

Function blocks are used for tasks that cannot be implemented with functions:

- Whenever timers and counters are required in the blocks, or
- When information must be stored in the program, such as preselection of the operating mode with a button.

Function blocks are always executed when called from another logic block. A function block can also be called several times at different points within a program. This facilitates the programming of frequently recurring complex functions.

A call of a function block is referred to as an instance. Each instance of a function block is assigned a memory area that contains the data that the function block uses. This memory is made available by data blocks created automatically by the software.

It is also possible to provide memory for multiple instances in one data block in the form of a **multi-instance**. The maximum size of instance data blocks varies depending on the CPU. The tags declared in the function block determine the structure of the instance data block.

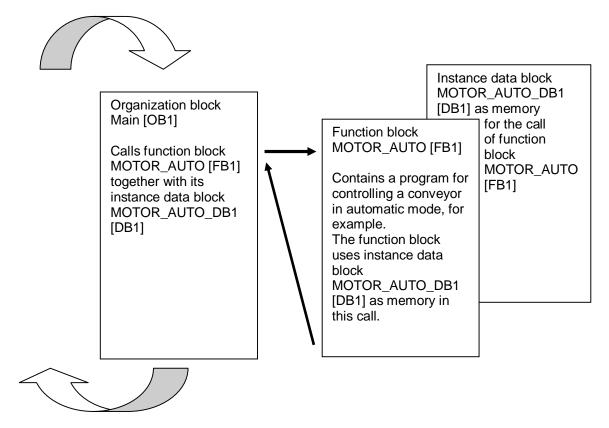


Figure 4: Function block and instance with call from organization block Main [OB1]

4.6 Global data blocks

In contrast to logic blocks, data blocks contain no instructions. Rather, they serve as memory for user data.

Data blocks thus contain variable data that is used by the user program. You can define the structure of global data blocks as required.

Global data blocks store data that can be used **by all other blocks** (see Figure 5). Only the associated function block should access instance data blocks. The maximum size of data blocks varies depending on the CPU.

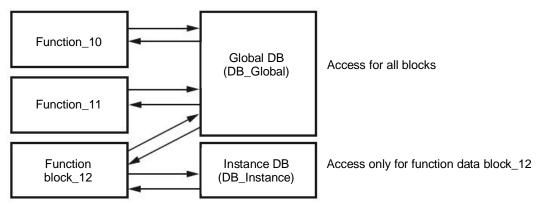


Figure 5: Difference between global DB and instance DB.

Application examples for global data blocks are:

- Saving of information about a storage system. "Which product is located where?"
- Saving of recipes for particular products.

4.7 Library-compatible logic blocks

A user program can be created with linear or structured programming. *Linear programming* writes the entire user program in the cycle OB, but is only suitable for very simple programs for which other less expensive control systems, such as LOGO!, can now be used.

For more complex programs, *structured programming* is always recommended. Here, the overall automation task can be broken down into small sub-tasks in order to implement a solution for them in functions and function blocks.

In this case, library-compatible logic blocks should be created preferentially. This means that the input and output parameters of a function or function block are defined generally and only supplied with the current global tags (inputs/outputs) when the block is used.

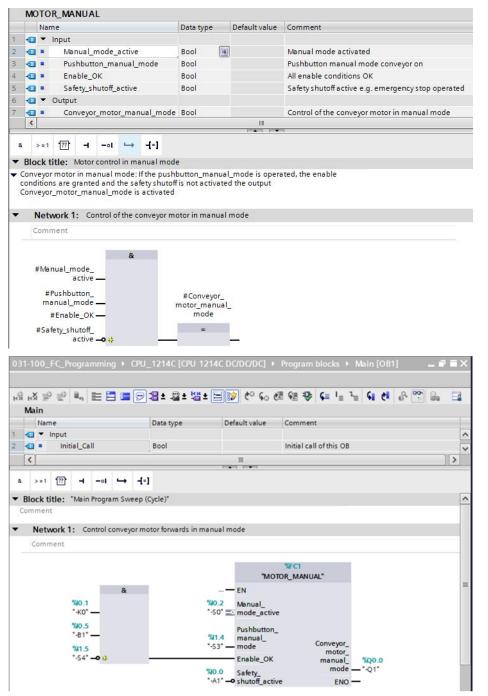


Figure 6: Library-compatible function with call in OB1

4.8 Programming languages

For SIMATIC S7-1200, the available programming languages for programming functions and function blocks are Function Block Diagram (FBD), Ladder Logic (LAD) and Structured Control Language (SCL).

The Function Block Diagram (FBD) programming language will be presented in the following.

FBD is a graphical programming language. The representation is based on electronic switching systems. The program is mapped in networks. A network contains one or more logic operation paths. Binary and analog signals are linked together by boxes. The graphical logic symbols known from Boolean algebra are used to represent the binary logic.

You can use binary functions to query binary operands and to logically combine their signal states. The following instructions are examples of binary functions: "AND operation", "OR operation" and "EXCLUSIVE OR operation". These are shown in Figure 7.

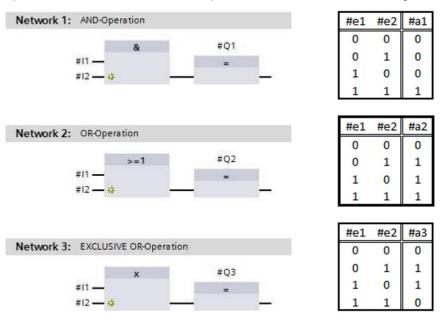


Figure 7: Binary functions in FBD and associated logic table

You can thus use simple instructions, for example, to control binary outputs, evaluate edges and execute jump functions in the program.

Program elements such as IEC timers and IEC counters provide complex instructions.

The empty box serves as a placeholder in which you can select the required instruction.

Enable input EN (enable) / Enable output ENO (enable output) mechanism:

- An instruction without EN/ENO mechanism is executed independent of the signal state at the box inputs.
- Instructions with EN/ENO mechanism are only executed if enable input "EN input has signal state "1". When the box is processed correctly, enable output "ENO" has signal state "1". If an error occurs during processing, the enable output "ENO" is reset. If enable input EN is not connected, the box is always executed.

5 Task

The following functions of the sorting station process description will be planned, programmed and tested in this chapter:

- Manual mode - Control of conveyor tracking forwards in manual/jog mode

6 Planning

The programming of all functions in OB1 is not recommended for reasons of clarity and reusability. The majority of the program code will therefore be moved into functions (FCs) and function blocks (FBs). The decision on which functions is to be moved to FCs and which is to run in OB 1 is planned below.

6.1 EMERGENCY STOP

The EMERGENCY STOP does not require a separate function. Just like the operating mode, the current state of the EMERGENCY STOP relay can be used directly at the blocks.

6.2 Manual mode – Conveyor motor in manual mode

Manual mode of the conveyor motor is to be encapsulated in a function (FC) "MOTOR_MANUAL". On the one hand, this preserves the clarity of OB1. On the other hand, it enables reuse if another conveyor belt is added to the station. Table 2 lists the planned parameters.

Input	Data type	Comment
Manual_mode_active	BOOL	Manual mode activated
Pushbutton_manual_mode	BOOL	Pushbutton to switch on conveyor in manual mode
Enable_OK	BOOL	All enable conditions OK
Safety_shutoff_active	BOOL	Safety shutoff active e.g. emergency stop operated
Output		
Conveyor_motor_manual_mode	BOOL	Control of the conveyor motor in manual mode

Table 2: Parameters for FC "MOTOR_MANUAL"

Output Conveyor_motor_manual_mode is ON as long as Pushbutton_manual_mode is pressed, manual mode is activated, the enable conditions are met and the safety shutoff is not active.

6.3 Technology diagram

Here, you see the technology diagram for the task.

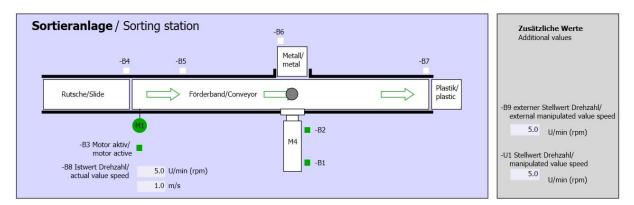


Figure 8: Technology diagram

Schalter der Sortieranlage	Automatikbetrieb	Handbetrieb / Manual mode
Switches of sorting station -P1 ein/on	Automatic mode -P5 gestartet/started	-S3 Tippbetrieb -M1 vorwärts/ Manual -M1 forwards
-Q0 Hauptschalter/Main switch	-S1 Start/start	-S4 Tippbetrieb -M1 rückwärts/ Manual -M1 backwards
-P4 akuleucuve -P4 Akuleucuve -P4 Akuleucuve -P4 Akuleucuve -P2 Hand/manual -P3 Auto/auto	-S2 Stopp/stop	-P7 ausgefahren/extended -S6 Zylinder -M4 ausfahren/ cylinder -M4 extend
-S0 Betriebsart/operating mode		-S5 Zylinder -M4 einfahren/ cylinder -M4 einfahren/ cylinder -M4 retract

Figure 9: Control panel

6.4 Reference list

DI	Туре	Identifier	Function	NC/NO
I	BOOL	-A1	Return signal emergency stop ok	NC
I	BOOL	-K0	Main switch "ON"	NO
І 0.2	BOOL	-S0	Mode selector manual (0)/ automatic (1)	Manual = 0 Auto = 1
I	BOOL	-B1	Sensor cylinder M4 retracted	NO
І 1.4	BOOL	-S3	Pushbutton manual mode conveyor M1 forward	NO
І 1.5	BOOL	-S4	Pushbutton manual mode conveyor M1 reverse	NO

The following signals are needed as operands for this task.

DO	Туре	Identifier	Function	
Q	BOOL	-Q1	Conveyor motor M1 forwards fixed speed	

Legend for reference list

DI	Digital Input	DO	Digital Output

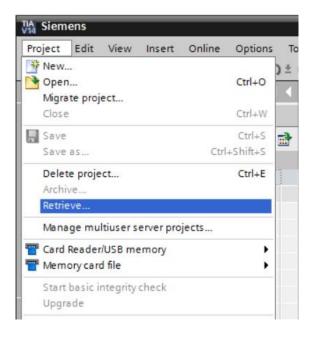
- AI Analog Input AO Analog Output
- I Input Q Output
- NC Normally Closed
- NO Normally Open

7 Structured step-by-step instructions

You can find instructions on how to carry out planning below. If you already have a good understanding of everything, it is sufficient to focus on the numbered steps. Otherwise, simply follow the detailed steps in the instructions.

7.1 Retrieve an existing project

Before we can start programming the function (FC) "MOTOR_MANUAL", we need a project with a hardware configuration (e.g. SCE_EN_011_101_Hardware_Configuration_ CPU1214C.zap14). To retrieve an existing project that has been archived, you must select the relevant archive with ® Project ® Retrieve in the project view. Confirm your selection with Open. (® Project ® Retrieve ® Select a .zap archive ® Open).



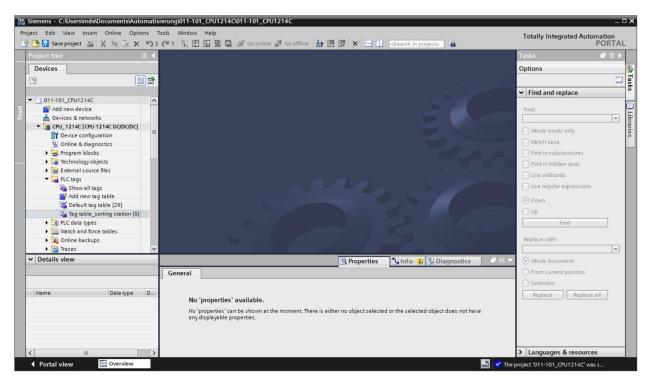
® The next step is to select the target directory where the retrieved project will be stored. Confirm your selection with "OK". (® Target directory ® OK)

7.2 Create a new tag table

In the project view, navigate to the
PLC tags of your controller and create a new tag table by double-clicking
Add new tag table.

🐫 Siemens - C:\Users\mde\Documents\Automatisierung\011-101_CPU1214C\011-101_CPU1214C		_ 🗆 X
Project Edit View Insert Online Options Tools Window Help	Totally Integrated Automation	
🚯 🐚 🗛 Save project 🚇 🙏 🗐 🚡 🗙 崎 🛨 🍘 📜 🚺 🔛 🖉 🚱 🕼 🖉 🌆 🖉 🕼 🖉 Search in j		TAL
Project tree	Tasks 🖬 🛙	
Devices	Options	
		Tasks
	✓ Find and replace	ks
▼ 1011-101_CPU1214C ▲		
₩ Add new device	Find:	-
🛱 🊠 Devices & networks		- ii
CPU_1214C [CPU 1214C DC/DC/DC]	Whole words only	Libraries
Configuration		es
U Online & diagnostics	Match case	
🕨 🖕 Program blocks	Find in substructures	
E Technology objects	Find in hidden texts	
External source files	Use wildcards	
▼ 🔄 PLC tags	Use regular expressions	
Show all tags		
Add new tag table	Down	
Gefault tag table [29]		
Qan PLC data types Match and force tables		
Ggi Watch and force tables	Find	
	Replace with:	
Traces		-
		<u> </u>
	Diagnostics 💿 🖃 🖉 💿 Whole document	
Module General Cross-references Compile Energy Suite	O From current position	
3 A 0 Show all messages	Selection	
Name Story and the story and the story of th	Replace Replace all	
Device configuration		
U Online & diagnostics	Go to ? Errors War	
as Program blocks		
Technology objects		
(a) External source files		
PLC tags		
PLC data types V <	> > Languages & resources	_
✓ Portal view	📑 🗹 The project '011-101_CPU1214C' was s	

Rename the tag table you just created as "Tag_table_sorting_station" (
 right-click
 "Tag_table_1" ® "Rename" ® Tag_table_sorting_station).



® Open this tag table with a double-click. (® Tag_table_sorting_station)

Ma Siemens - C:\Users\mde\Documents\Automat	isierung\011-10	1_CPU1214C\011-10	1_CPU1214C								- 0
📑 📴 🔚 Save project 📑 🐰 🏥 🗐 🗙 🍤	Vect Edit View Insert Online Options Totally Integrated Automation Project tree View V										
Project tree 🔲 🗸	011-101_CP	U1214C + CPU_12	14C [CPU 1214C D	C/DC/DC] > PLC ta	gs ► Ta	g table_	sorting	station	[0] = 🔳 🔳 🕽	K Tasks	
Devices							- Tags		User constants	Options	
	🥩 🥐 🖻	🕂 😤 🕅							_		
<u>ت</u>	Tag table	_sorting station								✓ Find and replace	
- 011-101_CPU1214C	Nam	e	Data type	Address	Retain	Acces	Writa	Visibl	Comment		
	1	:Add new>				1	V	1		Find:	
										Whole words only	
										Match case	
											- 1
External source files											- 1
											- 1
										Use regular expressions	
										Down	- 1
										Qup	- 1
										1.0.04	- 1
										Replace with:	- 1
🕨 💽 Traces	<									>	-
✓ Details view				Q Prope	rties	1 Info	1 V I	Diagnos	tics	Whole document	
	Conoral	7						5		From current position	
	-										- 1
Name Data and D	lag		Tag								
Name Data type D			General							Replace Replace all	
			General								
		Ē		Name:							- 1
		•		Data type:					1/21		
				and the second						> 1	_
		_			etained						

7.3 Create new tags within a tag table

® Add the name Q1 and confirm the entry with the Enter key. If you have not yet created additional tags, TIA Portal now automatically assigns data type "Bool" and address %I0.0 (I 0.0) (® <Add> ® Q1 ® Enter).

						📶 Tags	El	Jser constants
) 1	11 😤 🗹 🗲							1
Tag	table sorting station							
iay								
Tay	Name	Data type	Address	Retain	Acces	Writa	Visibl	Comment
Tay	Name		Address	Retain	Acces	Writa	Visibl	Comment

® Change the address to %Q0.0 (Q 0.0) by entering this directly or by clicking the drop-down arrow to open the Addressing menu. Change the operand identifier to Q and confirm with Enter or by clicking the check mark (® %I0.0 ® Operand identifier® Q ®

								Tage		lser const	
Ŷ	¥.	🕈 🛃 📽 🛍									t,
	Tag ta	able_sorting_station									
	1	Name	Data type	Address		Retain	Acces	Writa	Visibl	Comment	
1		Q1	Bool	%10.0	-						
		<add new=""></add>									
				O	pera	nd ident	tifter:			-	
					C	perand	type: Q				
						Add	ress: M				
						Bit num	ber: 0				

® Enter the "Conveyor motor M1 forwards fixed speed" comment for the tag.

										🕢 Tags	User con	istants
10	*	b	r 🗊									E
		The second second										
3	ag t	able_so	ting statio	n								
3		able_son	Data type		Address	Retain	Acces	Writa	Visibl	Comment		
1					Address %Q0.0	Retain	Acces	Writa	Visibl	Comment conveyor motor -N	11 forwards fixed	speed

 R Add a new Q2 tag in line 2. TIA Portal has automatically assigned the same data type as the one in line 1 and has incremented the address by 1 to %Q0.1 (Q0.1). Enter the comment "Conveyor motor M1 backwards fixed speed".

								🕣 Tags	User constants
	Name	Data type	Address	Retain	Acces	Writa	Visible in	Comment	
-0			%Q0.0					conveyor motor -M1	forwards fixed speed
-	07	Real	8001					converse motor MI	backwards fixed speed

7.4 Import "Tag_table_sorting_station"

 To insert an existing symbol table, right-click on an empty field of the created "Tag_table_sorting_station". Select "Import file" in the shortcut menu. (® Right-click in an empty field of the tag table ® Import file)

	-101_	CPU1214C	CPU_1214	4C [CPU 1214C	DC/DC/D	C] 🕨 PL	C tags	 Tag t 	able_sorting_st	ation [2] 🛛 🗖 🗖	×
									🕣 Tags	User constants	
#	e 🛃	• 🕂 약 (11							E	d
đ	ag ta	ble_sorting	_station								
	N	lame	Data type	Address	Retain	Acces	Writa	Visibl	Comment		
1	-	Q1	Bool	%10.0					conveyor motor -	M1 forwards fixed speed	
2		Q2	Bool 🔳						conveyor motor -	M1 backwards fixed spee	d
3			🚔 Insert row 💣 Add row				V	1			
			X Cut		Ctrl+	×					
			Copy		Ctrl+	c					
			Paste		Ctrl+	V					
			X Delete		De	el					
			Rename		F.	2					
			X Cross-refe	rences	F1	1					
			Cross-refe	rence information	n Shift+F1	1					
			Monitor all								
			Import file								
			Export file								
	<		💁 Properties								>

- Select the desired symbol table (e.g. in .xlsx format) and confirm the selection with "Open".
 (® SCE_EN_020-100_Tag_table_sorting_station... ® Open)
- When the import is finished, you will see a confirmation window and have an opportunity to view the log file for the import. Click
 OK.
 O

Import co	ompleted. (0032:000001) X
	Import completed successfully.
	Detailed information is shown in the import log file.
	Click here to view the log file.
	ОК

- You can see that some addresses have been highlighted in orange. These are duplicate
 addresses and the names of the associated tags have been numbered automatically to avoid
 confusion.
- ® Delete the duplicate tags by selecting the lines and pressing the Del key on your keyboard or selecting "Delete" in the shortcut menu. (® Right-click on selected tags ® Delete)

4	E							°° 🗊		-	-
								orting station	able_so	Tag t	1
	Comment	Visibl	Writa	Acces	Retain		Address	Data type	Name		
	conveyor motor -M1 forwards fixed speed						%Q0.0	Bool	Q1		1
ĺ	conveyor motor -M1 forwards fixed speed						%Q0.1	Bool	Q2		2
	return signal emergency stop ok (nc)					•	%10.0	Bool	-A1	-	3
	main switch "ON" (no)						%10.1	Bool	-КО	-	ļ.
	mode selector manual(0) / automatic(1)						%10.2	Bool	-50		5
	pushbutton automatic start (no)	\checkmark					%10.3	Bool	-51		6
	pushbutton automatic stop (nc)						%10.4	Bool	-S2	-	7
	sensor cylinder -M4 retracted (no)						%10.5	Bool	-B1	-00	8
	sensor cylinder -M4 extended (nc)						%10.6	Bool	-B2		Э
	sensor motor -M1 actice (pulse signal for						%10.7	Bool	-B3		10
	sensor part at slide (no)						%11.0	Bool	-84	-	11
	sensor metal part (no)						%11.1	Bool	-85	-	12
	sensor part in front of cylinder -M4 (no)						%11.2	Bool	-86	-	13
	sensor part at end of conveyor (no)						%11.3	Bool	- 87		14
	pushbutton manual mode conveyor -M1						%11.4	Bool	-53	-	15
	pushbutton manual mode conveyor -M1						%11.5	Bool	-S4	-0	16
	pushbutton manual mode cylinder -M4 re						%11.6	Bool	-55		17
	pushbutton manual mode cylinder -M4 ex.						%11.7	Bool	-56		18
	conveyor motor -M1 forwards fixed speed						%Q0.0	Bool	-Q1	-00	19
d	conveyor motor -M1 backwards fixed speed						%Q0.1	Bool	-Q2	-	20
	conveyor motor -M1 variable speed						%Q0.2	Bool	-Q3		21
	cylinder -M4 retract						%Q0.3	Bool	-M2		22
	cylinder -M4 extend						%Q0.4	Bool	-MB	-	23
	display "main switch on"						%Q0.5	Bool	-P1	-	24
	display "manual mode"						%Q0.6	Bool	-P2	-	25
	display "automatic mode"						%Q0.7	Bool	-P3	-	26

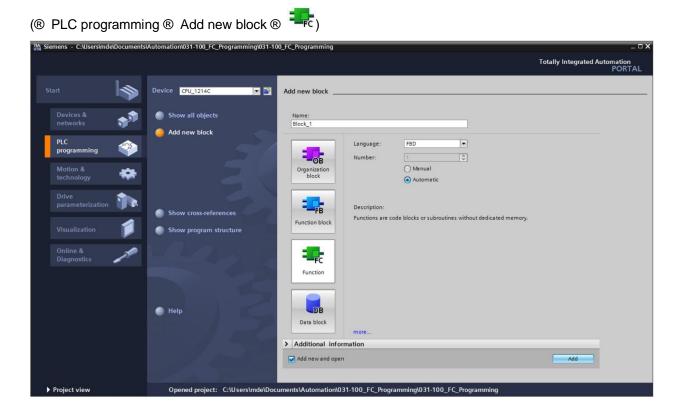
 You now have a complete symbol table of the digital inputs and outputs in front of you. Save your project under the name 031-100_FC Programming.

roje	ect tree	0	< 01	11-10	1_CPU12	214C + CPL	J_1214C [CPU 1	214C	DC/DC/	DC] 🕨	PLC tags	s 🕨 Tag	g table_sorting station [28]		_ # =>
De	vices													🕣 Tags	User constants
N)			3 🚽	(ř 👘		***									
				Tag	table so	orting statio	n								
· Гэ	011-101_CPU1214	c	~	-	Name	Data type	Address		Retain	Acces	Writa	Visibl	Comment		
	Add new device	6	1	4	-A1	Bool	%10.0	-					return signal emergency stop ok (nc)		
	Devices & netw	orks	2			Bool	%I0.1						main switch _ON" (no)		
•	CPU_1214C [CP	U 1214C DC/DC/DC]	3	4	-50	Bool	%10.2						mode selector manual(0) / automatic(1)		
	Device confi	guration	≡ 4	-	-51	Bool	%10.3						pushbutton automatic start (no)		
	😵 Online & dia	gnostics	5	4	-52	Bool	%10.4						pushbutton automatic stop (nc)		
	+ 🛃 Program blo	cks	6	-	-B1	Bool	%10.5						sensor cylinder -M4 retracted (no)		
	Technology	objects	7	•	-B2	Bool	%10.6						sensor cylinder -M4 extended (nc)		
	External sou	rce files	8	-	-B3	Bool	%10.7						sensor motor -M1 actice (pulse signal for		
	▼ 🞑 PLC tags		9	-	-B4	Bool	%11.0						sensor part at slide (no)		
	how all	tags	10		-85	Bool	%11.1						sensor metal part (no)		
	Add new	tag table	11	•	-86	Bool	%11.2						sensor part in front of cylinder -M4 (no)		
	👫 Default ti	ag table [29]	12	-	-87	Bool	%11.3						sensor part at end of conveyor (no)		
	🍇 Tag table	_sorting station [28	13	-	-53	Bool	%11.4						pushbutton manual mode conveyor –M1		
	PLC data typ	es	14	•	-54	Bool	%11.5						pushbutton manual mode conveyor -M1		
	 Watch and for 	orce tables	15	•	-55	Bool	%11.6						pushbutton manual mode cylinder -M4 re		
	🕨 🛐 Online back	ups	16		-56	Bool	%11.7						pushbutton manual mode cylinder -M4 ex		
	🕨 🚰 Traces		¥ 17	•	-Q1	Bool	%Q0.0						conveyor motor -M1 forwards fixed speed		
D	etails view		18		-Q2	Bool	%Q0.1						conveyor motor -M1 backwards fixed speed		
			19	•	-Q3	Bool	%Q0.2						conveyor motor -M1 variable speed		
			20		-M2	Bool	%Q0.3						cylinder -M4 retract		
			21	•	-MB	Bool	%Q0.4						cylinder -M4 extend		
Na	me	Data type	22	-	-P1	Bool	%Q0.5						display "main switch on"		
1	-Q1	Bool 9	^ 23	•	-P2	Bool	%Q0.6						display "manual mode"		
1	-Q2	Bool 9	24	•	-P3	Bool	%Q0.7						display "automatic mode"		
1	-Q3	Bool 9	25	-	-P4	Bool	%Q1.0						display "emergency stop activated"		
	-50	Bool 9	≡ 26	•	-P5	Bool	%Q1.1						display "automatic mode started"		
	-S1	Bool 9	27	•	-P6	Bool	%Q1.2						display cylinder -M4 "retracted"		
1	-52	Bool 9	~	_	_		_	_		-			Properties	nfo 追 况 Di	

(® Project ® Save as ... ® 031-100_FC Programming ® Save)

7.5 Create function FC1 "MOTOR_MANUAL" for the conveyor motor in manual mode

In the PLC programming section of the portal view, click "Add new block" to create a new function.



Rename your new block to: "MOTOR_MANUAL", set the language to FBD and keep automatic assignment of the number. Select the "Add new and open" check box. You will thus be taken automatically to your created function block in the project view. Click "Add".
 (® Name: MOTOR_MANUAL® Language: FBD ® Number: Automatic ® Add new and open ® Add)

Name:		
MOTOR_MANUAL	~	
-	Language:	FBD
OB	Number:	1 I I I I I I I I I I I I I I I I I I I
Organization block		Automatic
	Description:	
Function block		code blocks or subroutines without dedicated memory.
Function		
Data block		

7.6 Define the Interface of function FC1 "MOTOR_MANUAL"

If you selected "Add new and open", the project view opens with a window for creating the block you just added.

			R_MANUAL			≝ 48 4≱ 6≡ != 1= 61 61 64 0° 0° 8	
		Na	me	Data type	Default value	Comment	
		•	Input				
2			<add new=""></add>				
2 3 4	-	•	Output				
			<add new=""></add>				
5	-	•	InOut				
6			<add new=""></add>				
7	-	•	Temp				
8			<add new=""></add>				
9			Constant				
10			<add new=""></add>				
			Return				
12	-		MOTOR_MANUAL	Void			
	<				1111		>
_					how and how and		
8		> = 1		=]			
•	Blo	ck	title:				^
C	lom	mei	nt				10
		lat	work 1:				=
•							_
•	C	om	ment				
•	C	om	ment				~

 A binary output signal is needed for controlling the conveyor motor. For this reason, we first create local output tag #Conveyor_motor_manual_mode of the "Bool" type. Enter comment "Control of the conveyor motor in manual mode" for the parameter.

(
 Output: Conveyor_motor_manual_mode
 Bool
 Control of the conveyor motor in manual mode)

F	C_P	prog	gramming CPU_1214C [CPU 1	214C DC/DC/DC]	 Program bloc 	ks ▶ MOTOR_MANUAL[FC1] 📃 🖬 🖬 🕇
Ŕ			후 관 🔍 🖿 🚍 🗩 溜 ± DR_MANUAL	8 t 19 t 🖃 😥) ¢° 60 ¢8 98	⇒会传到可能广产Ⅲ, 日
		Na	me	Data type	Default value	Comment
1	-	-	Input			
2			<add new=""></add>			
3	-	•	Output			
4	-		Conveyor_motor_manual_mode	Bool		Control of the conveyor motor in manual mode
5	1		<add new=""></add>			
5	-	•	InOut			
7			<add new=""></add>			
8	-	•	Temp			
9			<add new=""></add>			
10	-	•	Constant			
11			<add new=""></add>			
12	-	•	Return			
13	-		MOTOR_MANUAL	Void		

R Add parameter #Manual_mode_active as the input interface under Input and confirm the entry with the Enter key or by exiting the entry field. Data type "Bool" is assigned automatically. This will be retained. Next, enter the associated comment "Manual mode activated".

(
 Manual_mode_active
 Enter
 Bool
 Manual mode activated)

® Continue by adding parameters #Pushbutton_manual_mode, #Enable_OK and #Safety_shutoff_active as additional binary input parameters and check their data types. Add descriptive comments.

				1	1 2		
		Na	me	Data type	Defau	Comment	
1		•	Input				1
2	-		Manual_mode_active	Bool		Manual mode activated	
3	-		Pushbutton_manual_mode	Bool		Pushbutton manual mode conveyor on	
4	-		Enable_OK	Bool		All enable conditions OK	
5	-		Safety_shutoff_active	Bool		Safety shutoff active e.g. emergency stop operated	=
6			<add new=""></add>				
7		•	Output				
8	-		Conveyor_motor_manual_mode	Bool		Control of the conveyor motor in manual mode	
9			<add new=""></add>				
10		•	InOut				
11			<add new=""></add>				
12	-	•	Temp				
13			<add new=""></add>				
14	-	•	Constant				
15			<add new=""></add>				
16	-	-	Return				V

 For purposes of program documentation, assign the block title, a block comment and a helpful network title for Network 1.

(
 Block title: Motor control in manual mode
 Network 1: Control of the conveyor motor in manual mode)

		Na	me	Data type	Default value	Comment	-
1	-	•	Input				1
2	-		Manual_mode_active	Bool		Manual mode activated	-
3		•	Pushbutton_manual_mode	Bool		Pushbutton manual mode conveyor on	L
Ļ			Enable_OK	Bool		All enable conditions OK	
5			Safety_shutoff_active	Bool		Safety shutoff active e.g. emergency stop operated	
5		•	Output				
1		•	Conveyor_motor_manual_mode	Bool		Control of the conveyor motor in manual mode	
	<						>
8		>=1	1 → -ol → -[=] title: Motor control in manual mode				1
1	con	ditio	or motor in manual mode: If the pushb ns are granted and the safety shutoff or_motor_manual_mode is activated			rated, the enable	1
•		Net	work 1: Control of the conveyor mo	tor in manua	al mode		
		-	ment				

7.7 Program FC1: MOTOR_MANUAL

Below the interface description, you see a toolbar in the programming window with various logic functions and below that an area with networks. We have already specified the block title and the title for the first network there. Programming is performed within the networks using individual logic blocks. Distribution among multiple networks helps to preserve the clarity of the program. In the following, you will get to know the various ways you can insert logic blocks.

® On the right side of your programming window is a list of instructions you can use in the program. Under ® Basic instructions ® Bit logic operations, find function –[=] (Assignment) and use a drag & drop operation to move it to Network 1 (green line appears, mouse pointer with + symbol).

(
 Instructions
 Basic instructions
 Bit logic operations
 -[=])

?	C_	Pr	ogr	ar	nn	ning	, ,	(CPU	_1	214C	[CPU	1214	C DC/I	DC/DC]	► F	Progra	m blocks	s 🕨	MO	TOR_	MAN	IUAI	L [FC	[1]	_		K Inst	ructio	ons	;		3	7 0	Þ	
																												Opt	ions							
ŝ	H	X	1	-	ŝ.	ii,				8		28	.21	·121 ±		0	0 6 0	e	₽	¢≡	l _≡ 1	= 6	i el	1 8	00		-	1				inil	. •			Instructions
			OF									-	_								-	- 1						V	avor	ites			-			truc
		N	lam	e									Dat	a type		C	Defau	Commen	nt										uvon	rie s					-	10
1	-	1.	- 1	np	ut																							^ ⁸	> =	1	??	-	-01	→		SU
2	-				Ма	nua	L	nor	de_a	octiv	ve		Во	ol				Manualn	mod	e acti	ivated	Ľ														
з	-				Pu:	hb	utte	on_	mai	nua	I_mod	e	Во	ol				Pushbutt	ton r	manu	almo	de cor	nvey	oron				-{-								ų
4	-	1			En	able		к					Bo	ol				Allenabl	le co	nditio	ons O	к														og lesting
5	-	1			Sa	fety	sh	uto	off_a	ctiv	/e		Bo	ol				Safetysh	nuto	ffacti	ve e.g	. eme	rgen	ncy sto	op op	erated	1								_	SLIL
6					<a< td=""><td>dd i</td><td>iev</td><td>1></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>ins</td><td>tructi</td><td>ons</td><td></td><td></td><td></td><td>61</td></a<>	dd i	iev	1>																						ins	tructi	ons				61
7	-	1	• (Dut	pu	t																						Nam						Des		
8	-	1			Co	nve	yor	m	otor	r_m	anual	mode	Bo	ol				Control o	ofthe	e con	veyor	motor	r in m	nanua	almo	de		~ · [Gen						^	
	<																										>			-	opera	tions				lasks
					_	-		1									•											-	8					AND I		KS
2		>	• 1	1	?	2	•	-	0	-	+ -{	1																	•					ORIO	1	
-	RI	h	k ti	tle		Mo	tor	col	atrol	Lin	manu	al mor	e																E ×	100				EXCLU		L
														manı	ial moi	de is r	operate	ed, the ena	ahle										• -					Assig	-	5
															ted the				a lone										•					Nega		Libraries
C	on	ve	yor_	m	oto	r_n	han	ua	l_m	ode	is act	ivated																	0-					Reset		Te
																													1					Seto		"
•		N	etw	or	k '	:	Co	ntr	ol of	fthe	e conv	eyor m	otor i	manu	almod	e													🗉 s	_				Set bi		E
		Co	mm	er	t																								E R		T_BF			Reset		
																													🗉 s					Set/re	1	
			_		-	_	_	_																		-			E R					Reset	1	
																													•					Scan		L
																													•					Scan		1
																													•	-[P]-	-			Set of	v	
																												<			100			>		

- ® Now use drag & drop to move your output parameter #Conveyor_motor_manual_mode onto <??.?> above the block you just inserted. The best way to select a parameter in the interface description is by "grabbing" it at the blue symbol
 - (
 Conveyor_motor_manual_mode)

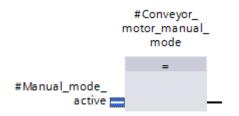
• 1	nput							
- P	Mai	nual_m	node_	active		Bool		Manual mode activated
•	Pus	hbutto	on_ma	nual_	mode	Bool		Pushbutton manual mode conveyor on
	Ena	ble_0	K			Bool		All enable conditions OK
•			_	active		Bool		Safety shutoff active e.g. emergency stop operated
•	<ad< td=""><td>d new</td><td>/></td><td></td><td></td><td></td><td></td><td></td></ad<>	d new	/>					
• (unningente							
•	Cor	iveyor_	_moto	r_mar	nual_mode	Bool	(terest	Control of the conveyor motor in manual mode
							June Ton And	
yor ion: yor	motor s are g _moto	in ma ranteo r_man	nual n d and t ual_m	node: the sa tode is	If the push fety shuto activated	ibutton_manual_ fis not activated	the output	d, the enable
mm	nent							
			_	. 19.3	_			
-	a a a a a a a a a a a a a a a a a a a	Safi Add Output Con	Safety_sh Add new Output Conveyor Conveyor Conveyor Trip - A title: Motor yor motor in ma ons are granted yor_motor_man etwork 1: Co	Add news Output Conveyor_moto Conveyor_moto Conveyor_moto Time - of Conveyor motor in manual r cons are granted and yor_motor_manual_m	A Safety_shutoff_active Add new> Output Conveyor_motor_mar Conveyor_motor_mar T T T T T T T T T T T T T T T T T T T	A Safety_shutoff_active Add new> Output Conveyor_motor_manual_mode Conveyor_motor_manual_mode T Tol	A Safety_shutoff_active Add new> Output Output Conveyor_motor_manual_mode Bool Conveyor_motor_manual_mode Bool Conveyor_motor_manual_mode title: Motor control in manual mode yor motor in manual mode: If the pushbutton_manual_ ions are granted and the safety shutoff is not activated yor_motor_manual_mode is activated etwork 1: Control of the conveyor motor in manual note	A Safety_shutoff_active Add new> Output Output Conveyor_motor_manual_mode Bool III III III III III III III

® This determines that the #Conveyor_motor_manual_mode parameter is written by this block. Still missing, however, are the input conditions so that this actually happens. For this, use drag & drop to move input parameter #Manual_mode_active to the left side of the assignment block.

(
 Manual_mode_active)

-	Nar	ne	Data type	Defau	Comment		
-	-	Input					
-		Manual_mode_active	Bool		Manual mode activated		
-		Pushbutton_manual_mode	Bool		Pushbutton manual mode conveyor on		
-		Enable_OK	Bool		All enable conditions OK		
-		Safety_shutoff_active	Bool		Safety shutoff active e.g. emergency stop operated		
		<add new=""></add>					
-	-	Output					
-00		Conveyor_motor_manual_mode	Bool		Control of the conveyor motor in manual mode		
<				II		>	ŝ
Blo		Image: matrix in the second secon			d the enable		
Blo	eyo litio		utton_manual_mode i	s operate	d, the enable		
Blo Conv Conv	eyo lition yeyo	itle: Motor control in manual mode r motor in manual mode: If the pushb is are granted and the safety shutoff i	utton_manual_mode i s not activated the out	s operate	d, the enable		
Blo Conv Conv	eyo lition eyo Net	itle: Motor control in manual mode r motor in manual mode: If the pushb is are granted and the safety shutoff i r_motor_manual_mode is activated	utton_manual_mode i s not activated the out	s operate	d, the enable		

® The input of the assignment block will also be logically combined with other parameters by an AND logic operation. To do this, first click the input of the block to which #Manual_mode_active is already connected, so that the input line has a blue background.



® Click the sicon in your logic toolbar to insert an AND logic operation between the #Manual_mode_active tag and your assignment block.

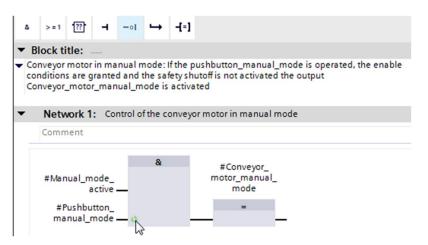
8	& >=1 [??] -I -oI → -[=]	
- í	AND logic operation [Shift+F2]	
- Co		button_manual_mode is operated, the enable f is not activated the output
•	Network 1: Control of the conveyor m	otor in manual mode
_	Comment	
	#Conveyor_ motor_manual_ mode #Manual_mode_ active	

® Double-click the second input of the & logic operation <??.?> and enter the letter "P" in the field that appears in order to see a list of available tags starting with "P". Click the #Pushbutton_manual_mode tag and apply with ® Enter.

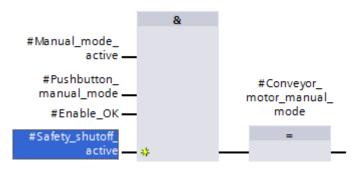
		Nar	me	Data type	Defau	Comment	-	
	-	-	Input					
			Manual_mode_active	Bool		Manual mode activated		
			Pushbutton_manual_mode	Bool		Pushbutton manual mode conveyor on		
1			Enable_OK	Bool		All enable conditions OK		
-	-		Safety_shutoff_active	Bool		Safety shutoff active e.g. emergency stop operated		
l			<add new=""></add>					
1	-	-	Output					
ł	-		Conveyor_motor_manual_mode	Bool		Control of the conveyor motor in manual mode		
ſ	<				H))	j
	ond		Image: The second s	s not activated the out				
	ond	ition	ns are granted and the safety shutoff i	s not activated the out				
Co	ond onv	ition eyoi Netr	rns are granted and the satety shutoff in r_motor_manual_mode is activated	s not activated the out				
Co	ond onv	itior eyoi Netv	work 1: Control of the conveyor mo	s not activated the out				

Note: When assigning tags in this way, there is a risk of a mix-up with the global tags from the tag table. The previously presented procedure using drag & drop from the interface description should therefore be used preferentially.

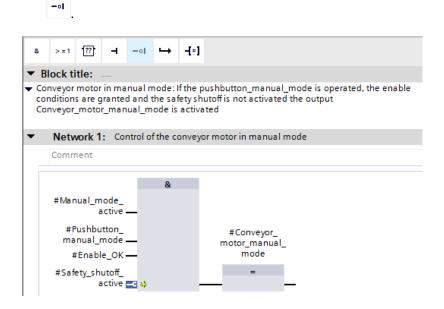
It is not active, the #Enable_OK and #Safety_shutoff_active input tags are logically combined with the AND logic operation. To do this, click twice on the yellow star of your AND block to add two additional inputs.



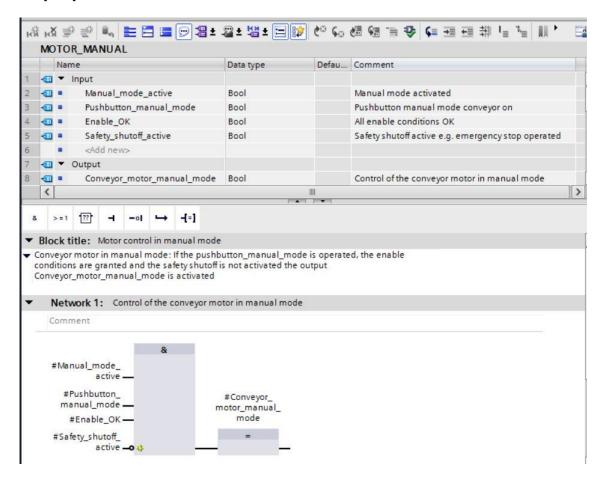
 R Add input tags #Enable_OK and #Safety_shutoff_active to your newly created inputs of the AND block.



® Negate the input connected to parameter #Safety_shutoff_active by selecting it and clicking



® Do not forget to click Save project regularly. The finished function "MOTOR_MANUAL" [FC1] in FBD is shown below.



® Under "General" in the properties of the block, you can change the "Language" to LAD (Ladder Logic) (® Properties ® General ® Language: LAD)

MOTOR_MANUAL [FC	1]	Q. Pr	roperties	🗓 Info 🔋 🗓 Diagnostics	
General					
General	General				^
Information	General				
Time stamps					
Compilation		Name:	MOTOR_MAN	NUAL	
Protection	4	Type:	FC		
Attributes	-		-		
	•	Language:	FBD		
		Number:	LAD		
			Manual		
			Automati	ic	*

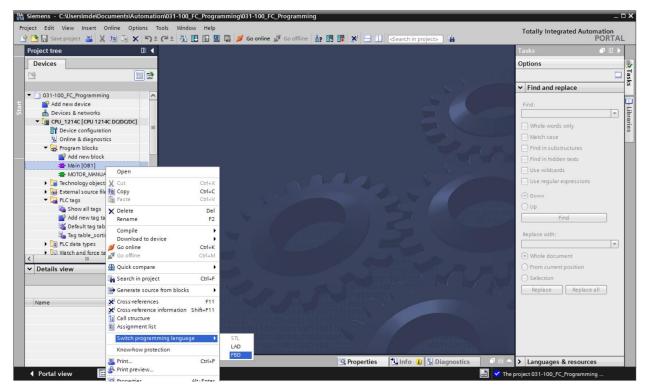
® The program has the following appearance in LAD.

		Nar	me	Data type	Defau	Comment		
		•	Input					
			Manual_mode_active	Bool		Manual mode activated		
			Pushbutton_manual_mode	Bool		Pushbutton manual mode conveyor on		
			Enable_OK	Bool		All enable conditions OK		
			Safety_shutoff_active	Bool		Safety shutoff active e.g. emergency stop operated		
			<add new=""></add>					
		•	Output					
			Conveyor_motor_manual_mode	Bool		Control of the conveyor motor in manual mode		
	<							
	Bloo	c k 1 eyo	title: Motor control in manual mode r motor in manual mode: If the pushb	utton_manual_mo	ode is operate	d, the enable		
	Bloo	eyo itior eyo	title: Motor control in manual mode r motor in manual mode: If the pushb ns are granted and the safety shutoff i r_motor_manual_mode is activated	utton_manual_mo s not activated the	ode is operate e output	d, the enable	-	
	Bloc onvi onvi onvi	ck t eyo ition eyo	title: Motor control in manual mode r motor in manual mode: If the pushb ns are granted and the safety shutoff i r_motor_manual_mode is activated work 1: Control of the conveyor mo	utton_manual_mo s not activated the	ode is operate e output	d, the enable		
E	Bloc onvi onvi onvi	ck t eyo ition eyo	title: Motor control in manual mode r motor in manual mode: If the pushb ns are granted and the safety shutoff i r_motor_manual_mode is activated	utton_manual_mo s not activated the	ode is operate e output	d, the enable		

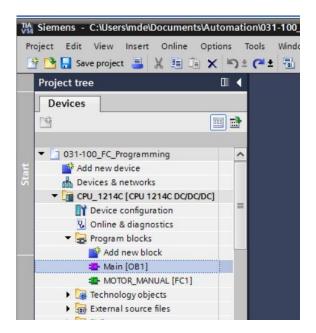
7.8 Program the organization block OB1 – Control conveyor tracking forwards in manual mode

Before programming organization block "Main [OB1]", we switch the programming language to FBD (Function Block Diagram). To do so, first click on "Main [OB1]" in the "Program blocks" folder.

(® CPU_1214C [CPU 1214C DC/DC/DC ® Program blocks ® Main [OB1] ® Switch programming language ® FBD)



® Open the "Main [OB1]" organization block with a double-click.



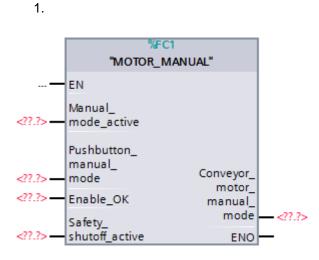
Retwork 1 the name "Control conveyor tracking forwards in manual/jog mode"
 (Retwork 1:... Recontrol conveyor tracking forwards in manual/jog mode)

											@ @ ♥ ♥ ⊨ '= '= ♥ ₩ ₽ ♥ ₪		4
	M	lai								Default value			
	1		Nar						Data type	Default value	Comment		
	4			Input	tial_Ca	u.			Bool		Initial call of this OB		4
					maner				Bool		=True, if remanent data are available		
				Temp	maniei	ice			5001		- nde, internation data are available		L
					dd nev	V>							
	-		-	Const	ant								
	1			<a< td=""><td>dd nev</td><td>V></td><td></td><td></td><td></td><td></td><td></td><td></td><td>1</td></a<>	dd nev	V>							1
	<								1). 	Ш		>	Ţ
-			_		1					heekend here			
2		>	= 1	??	-	-0	-	-[=]					
,	BI	oc	k t	itle:	"Main	Progra	m Sw	eep ((Cvcle)*				1
			nen					eete v					ľ
													J
•		N	et	vork	1: Co	ontrol c	onvey	ormo	tor forwards in r	manual mode			ų
		Co	m	nent									

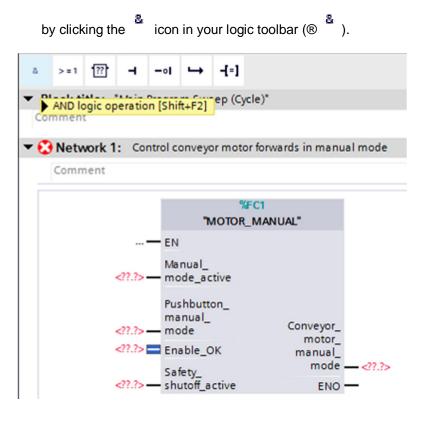
® Use drag & drop to move your "MOTOR_MANUAL [FC1]" function onto the green line in Network 1.

Devices	1	ية. اقر		5 =	67 an	B	= 2	1 2		<u>웹 t </u>	± 🖃 🎓 ና
		147	Ma						2	<u>u-u-u</u>	
031-100_FC_Programming	^			Na	me					Data type	Default value
Add new device		1	-	-	Input						
Devices & networks		2	-		Initia	al_Cal	I.			Bool	
CPU_1214C [CPU 1214C DC/DC/DC]		з	-		Rem	anen	ce			Bool	
Device configuration		4	-	-	Temp						
😵 Online & diagnostics		5			<adi< td=""><td>d new</td><td>></td><td></td><td></td><td></td><td></td></adi<>	d new	>				
🔻 🔂 Program blocks		6	-	-	Consta	nt					
📑 Add new block	-	7			<ad< td=""><td>d new</td><td>></td><td></td><td></td><td></td><td></td></ad<>	d new	>				
- Main [OB1]			<							11	Ш
- MOTOR_MANUAL [FC1]					[]]						
Technology objects		8		> = 1	??	-	-0	↦	-[=]		
External source files		-	Blo	ck	title: "	Main	Progra	m Sw	een (C	Vrle)*	
PLC tags		-	Com			NO IT I	rogiu		cep (c	Jele)	
PLC data types		10									
Watch and force tables		-	1	Net	work 1:	Co	ntrol c	onvey	or mo	tor forwards in n	nanual mode
🕨 🙀 Online backups			0	om	ment						
🕨 🔀 Traces											
Device proxy data	~		-			_	_	_	_		
Details view								-00.1	IOTOR	R MANUAL [FC1]	
	_							-104	10101	Turnene [i ci]	

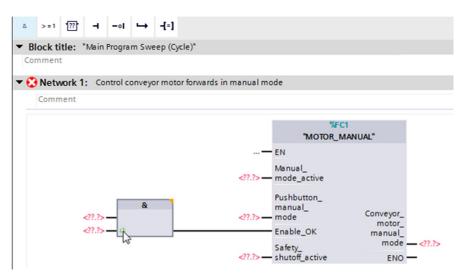
A block with the interface you defined and connections EN and ENO are inserted in Network



® To insert an AND before input parameter "Enable_OK", select this input and insert the AND



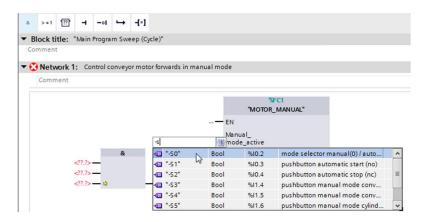
® Click the yellow star ** of the AND block to add another input (® **).



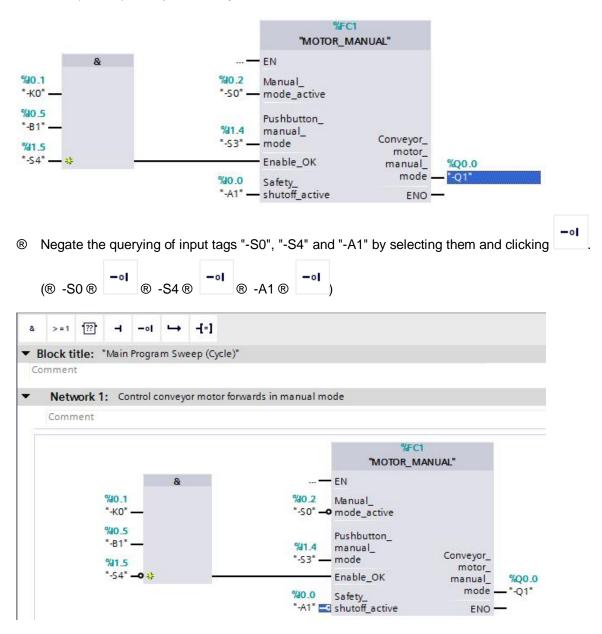
- It is the block to the global tags from "Tag_table_sorting_station", we have two options:
- Beither select the "Tag_table_sorting_station" in the project tree and use drag & drop to move the desired global tag from the Details view to the interface of FC1 (Beitag_table_sorting_station Beitag View. Beitag_Solutions Manual_mode_active)

	a Show all				Comment	
	💥 Default ti	ag table [28]				%FC1
	惧 Tag table	_sorting station [28	3]	~		"MOTOR_MANUAL"
	Details view	Data type				— EN Manual_ — mode_active
01	-Q1	Bool		^	& ²⁵	Pushbutton_
1	-Q2	Bool			?.? —	manual Conveyor
	-Q3	Bool				motor
	-S0	Bool			?.? *	Enable_OK manual_
		B				Safety_ mode — </td
0	-S1	Bool		_		- shutoff_active ENO -

® Or, enter the starting letters (e.g. "-S") of the desired global tag for <??.?> and select the global input tag "-S0" (%I0.2) from the displayed list (® Manual_mode_active ® -S ® -S0).

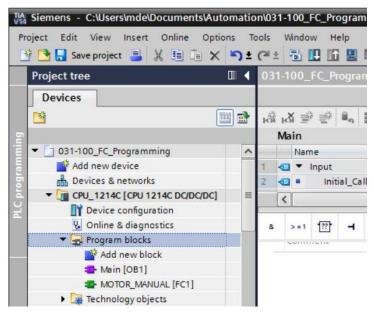


Insert the other input tags "-S3", "-K0", "-B1", "-S4" and "-A1" and then insert output tag "-Q1" (%Q0.0) at output "Conveyor_motor_manual_mode".



7.9 Save and compile the program

® To save your project, select the Save project button in the menu. To compile all blocks, click the "Program blocks" folder and select the icon for compiling in the menu (® Save project ® Program blocks ®).



® The "Info", "Compile" area shows which blocks were successfully compiled.

			Q Prope	ties Linfo	o 追 🗓 Diagr	nostics	
General (1) Cros	ss-references	Compile	Energy Suite	Syntax			
🕄 🛕 🕕 Show all n	nessanes		u				
Compiling finished (err	-						
					C	2 5	18/2
Path	U	escription			Go to	? Errors	i Wa
✓ ▼ CPU_1214C					~	0	0
📀 🛛 🔻 Program blo	cks				7	0	0
MOTOR_I	MANUAL (FC1) B	lock was succes	sfully compiled.		~		
Main (OE	1) B	lock was succes	sfully compiled.		~		
9	C	omniling finishe	d (errors: 0; warnings	0)			

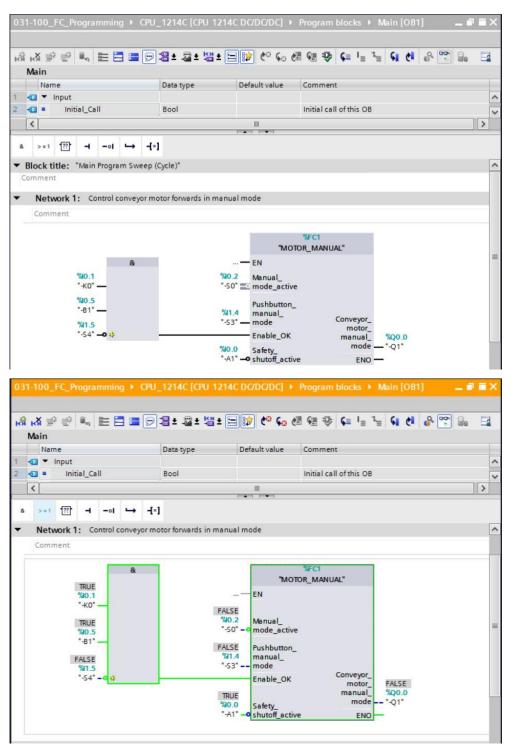
7.10 Download the program

Siemens - C:\Users\mde\Documents\Automat	on\031-100_FC_Programming\031-100_FC_Programming	_ 🗆 >
Project Edit View Insert Online Options	ools Window Help	Totally Integrated Automation
📑 🎦 🔚 Save project 昌 🐰 🏥 🛅 🗙 🎝 :	(# ± 🖥 🛄 📅 🚆 📮 💋 Go online 🖉 Go offline 🛔 🌆 🖪 👫 🗶 🖃 🛄 📀earch in projects 🕼	PORTAL
Project tree	031-100_FC_Programming > CPU_1214C [CPU 1214C DC/DC/DC] > Program blocks > Main [OB1] 💫 🗖 🖬 🗙	Instructions 📑 🗉 🕨
Devices		Options
	(청 전 후 속 🐂 돈 🔚 🖻 월 # 월 # 월 # 달 달 한 한 6 6 6 6 9 9 6 6 년 달 달 6 6 8 9 9 8 - 글	int • 🗔 🔤
5	Main	
▼ 3 031-100_FC_Programming		· Tavontes
Add new device	1 🕣 🔻 Input	& >=1 1??? -1 -01 └→
Bevices & networks	2 🕣 🖷 Initial_Call Bool Initial call of this OB	
CPU_1214C [CPU 1214C DC/DC/DC]		-{-1
Device configuration		
🖳 Online & diagnostics	a >=1 [??] -I01 -→ -[=]	1 •)
Program blocks	▼ Block title: "Main Program Sweep (Cycle)"	
Add new block	Comment	× Basic instructions
MOTOR_MANUAL [FC1]		✓ Basic instructions
Technology objects	Network 1: Control conveyor motor forwards in manual mode	Name Descri
External source files	Comment	General General General General
🕶 🌄 PLC tags		O Timer operations
Show all tags	%FC1 "MOTOR MANUAL"	+1 Counter operations
Add new tag table		Math functions Math functions
💥 Default tag table [29]	& EN	Math functions
✓ Details view	*#0.1 *#0.2 Manual_ *#0* **50* =: mode_active	Move operations
	Rin F	🕨 🏧 Conversion operations
	-81	Program control operati
	%1.5 "-S3" — mode Conveyor_ motor	Word logic operations
Name Details	*-54* ── Enable_OK manual_ %Q0.0	Shift and rotate
Add new block	\$40.0 Safety_ mode *-Q1*	
MOTOR MANUAL FC1	"-A1" — shutoff_active ENO —	
		<
	▼ Network 2:	> Extended instructions
		> Technology
	< III > 100%	> Communication
	🔍 Properties 🚺 Info 🚺 况 Diagnostics 💷 🗕 📥	> Optional packages
Portal view Overview	Main (OB1)	project 031-100_FC_Programming

7.11 Monitor program blocks

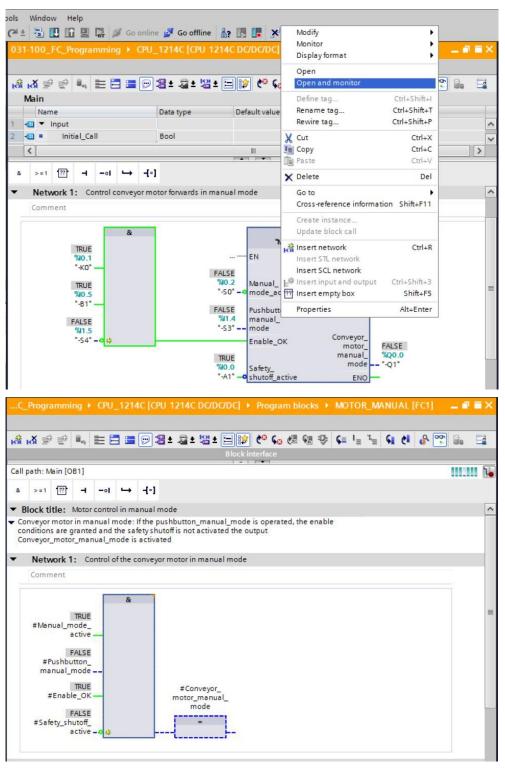
® The desired block must be open for monitoring the downloaded program. The monitoring can

now be activated/deactivated by clicking the 🔛 icon (® Main [OB1] ® 🖭).



Note: The monitoring here is signal-related and controller-dependent. The signal states at the terminals are indicated with TRUE or FALSE.

® The "MOTOR_MANUAL" [FC1] function called in the "Main [OB1]" organization block can be selected directly for "Open and monitor" after right-clicking (® "MOTOR_MANUAL" [FC1] ® Open and monitor).



Note: The monitoring here is function-related and controller-independent. The actuation of sensors and the station status are shown here with TRUE or FALSE.

If a particular point of use of the "MOTOR_MANUAL" [FC1] function is to be monitored, the call environment can be selected using the selected using the call environment (a OK)

Call	environment Dependency structure	Address	Details
1	Main	OB1	@Main > NW1 (Control conveyor motor forwards in manual mod
2			
			Transfer to "adjusted manually"

7.12 Archive the project

R As the final step, we want to archive the complete project. Select the
 R "Archive ..." command in the
 Project" menu. Select a folder where you want to archive your project and save it with the file type "TIA Portal project archive". (
 Project
 Archive
 TIA Portal project archive
 Archive
 TIA Portal project archive
 Archive
 TIA Portal project archive
 Archive
 TIA Portal
 project archive
 Archive
 Archive
 TIA Portal
 project archive
 Archive
 Archive
 TIA Portal
 project
 Archive
 Ar

IA Siemens - C:\Users\mde\Document	ts\Automat	on\031-100_FC_Programming\031-100_FC_Programming	_ ¤ ×
Project Edit View Insert Online		ools Window Help (*** 🖏 🗓 🕼 😰 📪 🍠 Goonline 🖉 Gooffline 🎎 🕞 🕞 🛠 🖃 🛄 Kearch in projects 👍	Totally Integrated Automation PORTAL
Open Migrate project	Ctrl+O	031-100_FC_Programming CPU_1214C [CPU 1214C DC/DC/DC] Program blocks Main [OB1]	Instructions
Close	Ctrl+W		Options
Save	Ctrl+S =	(2) (2) (2) (2) (2) (2) (2) (2) (2) (2)	int , 🗖 🔤 🚆
Save as Ctrl+	-Shift+S		Favorites A >=1 1 1 1 + -ei → 0 5
Delete project	Ctrl+E	Name Data type Default value Comment	v ravontes
Archive		1 🕣 🔻 Input	、 & >=1 ??? -I -oI 나→ 🖥
Retrieve		2 🕣 = Initial_Call Bool Initial call of this OB	
Manage multiuser server projects			-{=]
Tard Reader/USB memory Memory card file))	a >=1 (
Start basic integrity check		Network 1: Control conveyor motor forwards in manual mode	
Upgrade		Comment	
Print	Ctrl+P		✓ Basic instructions Name Descri
Print preview		%FC1	✓ Basic instructions Name Descri ▶ General 000000000000000000000000000000000000
C:\Users\mde\D\031-100_FC_Program	ming	"MOTOR_MANUAL"	General General General General General
C:\Users\mde\Docume\011-101_CPU1	1214C	& EN	
C:\Users\mde\Documen\011-100_CPL	J1200	%40.1 %40.2 Manual_	O Timer operations H Counter operations Comparator operations C A Comparator operations M Math functions
Exit	Alt+F4	"-K0" — "-50" — mode_active	Comparator operations
опше овекара		%0.5 *-B1*Pushbutton	Math functions
✓ Details view		741.4 manualConveyor	Move operations
		motor_	Conversion operations
		-54 -945 Enable_OK manual_ %Q0.0 %40.0 Safety mode	Program control operati
Name		*A1 — @ shutoff_active ENO	Word logic operations
Add new device			Shift and rotate
Devices & networks			1
CPU_1214C		Network 2:	
H Ungrouped devices		Comment	< III >
Common data			Extended instructions
Documentation settings		< III > 100% •	> Technology
🐻 Languages & resources		Properties Linfo Diagnostics	> Communication
		General Cross-references Compile Energy Suite Syntax	> Optional packages
🖣 Portal view 🔛 Overvie	ew 🗧		nnection to CPU_1214C terminated.

7.13 Checklist

No.	Description	Completed
1	Compiling successful and without error message	
2	Download successful and without error message	
3	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated MANUAL mode (-S0 = 0) Activate manual mode conveyor forwards (-S3 = 1) Conveyor motor forwards fixed speed (-Q1 = 1)	
4	Same as 3 but activate EMERGENCY OFF $(-A1 = 0)$ ® $-Q1 = 0$	
5	Same as 3 but AUTO mode $(-S0 = 1)$ \circledast $-Q1 = 0$	
6	Same as 3 but switch off station (-K0 = 0) \circledast -Q1 = 0	
7	Same as 3 but cylinder not retracted $(-B1 = 0)$ (B) $-Q1 = 0$	
8	Same as 8 but also activate manual mode conveyor backwards $(-S4 = 1) $ \circledast -Q1 = 0	
9	Project successfully archived	

8 Exercise

8.1 Task – Exercise

The following functions of the sorting station process description will be planned, programmed and tested in this chapter:

- Manual mode - Control of conveyor tracking backwards in manual/jog mode

8.2 Technology diagram

Here, you see the technology diagram for the task.

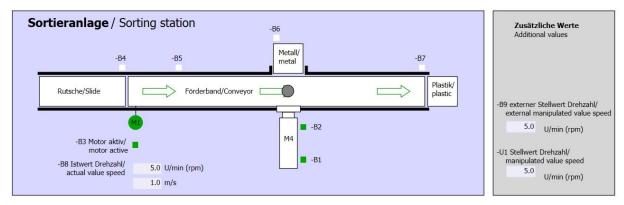


Figure 10: Technology diagram

-P4 aktiviert/active	-S2 Stopp/stop	-S3 Tippbetrieb -M1 vorwärts/ Manual -M1 forwards -S4 Tippbetrieb -M1 rückwärts/ Manual -M1 backwards -S5 Zylinder -M4 ausfahren/ cylinder -M4 ausfahren/ cylinder -M4 einfahren/ cylinder -M4 einfahren/
----------------------	----------------	--

Figure 11: Control panel

8.3 Reference list

DI	Туре	Identifier	Function	NC/NO
I 0.0	BOOL	-A1	Return signal emergency stop ok	NC
I 0.1	BOOL	-K0	Main switch "ON"	NO
1 0.2	BOOL	-S0	Mode selector manual (0)/ automatic (1)	Manual = 0 Auto = 1
I 0.5	BOOL	-B1	Sensor cylinder M4 retracted	NO
I 1.4	BOOL	-S3	Pushbutton manual mode conveyor M1 forward	NO
l 1.5	BOOL	-S4	Pushbutton manual mode conveyor M1 reverse	NO

The following signals are needed as operands for this task.

DO	Туре	Identifier	Function	
Q 0.1	BOOL	-Q2	Conveyor motor M1 backwards fixed speed	

Legend for reference list

DI	Digital Input	DO	Digital Output
AI	Analog Input	AO	Analog Output
I	Input	Q	Output
NC	Normally Closed		

8.4 Planning

NO

Plan the implementation of the task on your own.

Normally Open

8.5 Checklist – Exercise

No.	Description	Completed
1	Compiling successful and without error message	
2	Download successful and without error message	
3	Switch on station (-K0 = 1) Cylinder retracted / Feedback activated (-B1 = 1) EMERGENCY OFF (-A1 = 1) not activated MANUAL mode (-S0 = 0) Activate manual mode conveyor backwards (-S4 = 1) Conveyor motor backwards fixed speed (-Q2 = 1)	
4	Same as 8 but activate EMERGENCY OFF (-A1 = 0) \textcircled{B} -Q2 = 0	
5	Same as 8 but AUTO mode (-S0 = 1) $\ensuremath{\mathbb{B}}$ -Q2 = 0	
6	Same as 8 but switch off station $(-K0 = 0)$ ® $-Q2 = 0$	
7	Same as 8 but cylinder not retracted (-B1 = 0) \textcircled{B} -Q2 = 0	
8	Same as 8 but also activate manual mode conveyor forwards $(-S3 = 1)$ \circledast -Q1 = 0 and -Q2 = 0	
9	Project successfully archived	

9 Additional information

More information for further practice and consolidation is available as orientation, for example: Getting Started, videos, tutorials, apps, manuals, programming guidelines and trial software / firmware, under the following link:

www.siemens.com/sce/s7-1200

Preview "Additional information"

- Getting Started, Videos, Tutorials, Apps, Manuals, Trial-SW/Firmware
 - ↗ TIA Portal Videos
 - ↗ TIA Portal Tutorial Center
 - > Getting Started
 - ↗ Programming Guideline
 - Easy Entry in SIMATIC S7-1200
 - > Download Trial Software/Firmware
 - ↗ Technical Documentation SIMATIC Controller
 - ↗ Industry Online Support App
 - TIA Portal, SIMATIC S7-1200/1500 Overview
 - ↗ TIA Portal Website
 - ↗ SIMATIC S7-1200 Website
 - ↗ SIMATIC S7-1500 Website

Further Information

Siemens Automation Cooperates with Education siemens.com/sce

SCE Learn-/Training Documents siemens.com/sce/documents

SCE Trainer Packages siemens.com/sce/tp

SCE Contact Partners siemens.com/sce/contact

Digital Enterprise siemens.com/digital-enterprise

Industrie 4.0 siemens.com/future-of-manufacturing

Totally Integrated Automation (TIA) siemens.com/tia

TIA Portal siemens.com/tia-portal

SIMATIC Controller siemens.com/controller

SIMATIC Technical Documentation siemens.com/simatic-docu

Industry Online Support support.industry.siemens.com

Product catalogue and online ordering system Industry Mall **mall.industry.siemens.com**

Siemens AG Digital Factory P.O. Box 4848 90026 Nuremberg Germany

Subject to change and errors © Siemens AG 2018

siemens.com/sce