SIEMENS

Learn-/Training Document

Siemens Automation Cooperates with Education (SCE) | As of Version V9 SP1

PA Module P01-05
SIMATIC PCS 7 – Functional Safety

siemens.com/sce

SIEMENS
Global Industry Partner of WorldSkills International

worldskills

Matching SCE Trainer Packages for this Learn-/Training Document

- **SIMATIC PCS 7 Software Package V9.0 (set of 3)**
  Order No.: 6ES7650-0XX58-0YS5
- **SIMATIC PCS 7 Software Package V9.0 (set of 6)**
  Order No.: 6ES7650-0XX58-2YS5
- **SIMATIC PCS 7 Software Upgrade Packages (set of 3)**
  Order No.: 6ES7650-0XX58-0YE5 (V8.x→ V9.0)
- **SIMIT Simulation Platform with Dongle V10**
  (contains SIMIT S & CTE, FLOWNET, CONTEC libraries) – 2500 simulation tags
  Order No.: 6DL8913-0AK00-0AS5
- **Upgrade SIMIT Simulation Platform V10**
  (contains SIMIT S & CTE, FLOWNET, CONTEC libraries) from V8.x/V9.x
  Order No.: 6DL8913-0AK00-0AS6
- **Demo Version SIMIT Simulation Platform V10**
  Download
- **SIMATIC PCS 7 AS RTX Box (PROFIBUS) only in combination with ET 200M for RTX –**
  Order No.: 6ES7654-0UE23-0XS1
- **ET 200M for RTX Box (PROFIBUS) only in combination with PCS 7 AS RTX Box –**
  Order No.: 6ES7153-2BA10-4AB1

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

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 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. Send all related requests to scesupportfinder.i-ia@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

Functional safety

1 Goal

After working through this module, the students will know the basic requirements for functional safety. They will become familiar with methods for identifying potential hazards and for evaluating resulting risks. They will know methods and design concepts to safeguard plants by means of process control engineering. They will learn the basic logic operations for interlocking of controls.

2 Prerequisite

This chapter builds on chapter 'Control module functions'. To perform the tasks in this chapter, you can use an existing project from the previous chapter or the archived project 'p01-04-exercise-r1905-en.zip' provided by SCE. The download of the project(s) is stored on the SCE Internet for the respective module.

The simulation for the SIMIT program can be retrieved from the file 'p01-04-plantsim-v10-r1905-en.simarc'. It can be run in demo mode.

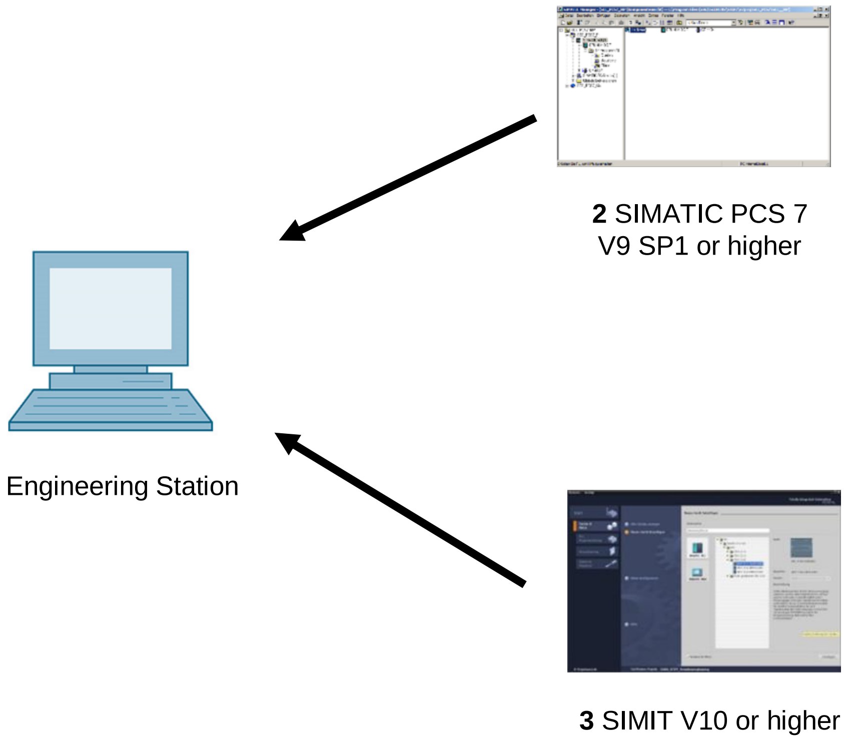
3 Required hardware and software

1 Engineering station: Requirements include hardware and operating system (for further information, see Readme on the PCS 7 installation DVD)

2 SIMATIC PCS 7 software V9 SP1 or higher

- Installed program packages (contained in SIMATIC PCS 7 Software Trainer Package):
  - *Engineering → PCS 7 Engineering*
  - *Engineering → BATCH Engineering*
  - *Runtime → Single Station → OS Single Station*
  - *Runtime → Single Station → BATCH Single Station*
  - *Options → SIMATIC Logon*
  - *Options → S7-PLCSIM V5.4 SP8*

3 Demo Version SIMIT Simulation Platform V10



4 Theory

4.1 Theory in brief

In modern production plants, process variables are used to control and safeguard technical processes. Permissible and impermissible ranges are defined for these variables based on the given technical constraints. The state of the overall plant results from the current values of all process values. The objective of functional safety is to prevent the plant from entering an impermissible state. To this end, appropriate interlocking mechanisms are set up. The objective of interlocks is to prevent signal combinations, signal sequences and characteristics of signals as a function of time that can lead to impermissible fault states.

This can be done by means of process control engineering through so-called safety instrumented systems. They prevent fault states from occurring, or they limit the harm if an impermissible fault state has occurred despite all measures taken. In order to design suitable interlocking mechanisms, a safety concept must be developed for the plant. This task requires exact knowledge of the chemical and process- and plant-related constraints. For this reason, the safety concept is developed by an interdisciplinary team using a HAZOP or PAAG analysis.

The technical implementation of the mechanisms in a process control system should be designed to be as simple, direct-acting and straightforward as possible. For this reason, recurrent standard protective circuits are frequently turned to in practice. They can be grouped into four categories:

Combination circuits are used to generate switching conditions through direct combination of the corresponding process signals. For this purpose, the input signals are combined with the logic operations AND, OR and NOT. The state of the output signal of such a combination circuit can thus be defined at any time through the states of the input signals.

Prioritization circuits permit giving certain signals precedence over other signals. This is often necessary in the case of operating mode selection as well as for start and stop functions. Prioritization circuits are often implemented using combination circuits.

Interlocking circuits prevent the simultaneous setting of differently acting signals. If a certain sequence is required for several control signals, this is referred to as a sequence interlock. Interlocks are implemented using RS flip-flops that are connected to each other.

Circuits with timing behavior allow for delayed switch-on/switch-off, the definition of a minimum or maximum runtime and the implementation of safety functions that require a certain response time. Various pre-assembled timer blocks are available for implementing such functions.

4.2 Process variables

Production plants are used to produce material goods. To that end, they control and monitor material and energy flows, which can be described with physical variables such as volume, mass, temperature and flow. Based on process- and plant-related constraints, the physical variables that are relevant to the technical process and measurable are defined and specified. These variables are referred to as process variables.

Process variables are used to control or safeguard technical processes. The intended ranges for each process variable (the OK range) are specified based on the chemical and process- and plant-related constraints. In addition, ranges outside the OK range in which no safety-related restrictions for further operation exist (permissible fault range) are defined. If a process variable is outside these ranges, undesired events that result directly in bodily injury or environmental damage must be expected (impermissible fault range).

The values of the process variables are acquired and evaluated by means of process control engineering. The current state of the plant is determined from this. Three basic states are differentiated:

– OK state: The values of all process variables are within their respective OK range, and no danger is emanating elsewhere from the plant.

– Permissible fault state: The values of one or more process variables are in the respective permissible fault range. No danger is emanating elsewhere from the plant.

– Impermissible fault state: The values of one or more process variables are in the respective impermissible fault range or a danger is emanating elsewhere from the plant. Impermissible fault states always exist when humans are at risk, the environment will be damaged, technical equipment will be destroyed or the production results will be impaired. In this case, it is enough that the probability of occurrence of any one of these events is sufficiently high. [1].

4.3 Functional safety

Functional safety refers in general to the safeguarding of the process plant against fault states [1]. For many processes and states in process plants, certain events can occur that result in harm. The combination of the frequency of occurrence of a harm and the extent of the harm is called the risk of the corresponding process or state. The goal of functional safety is to take safety measures that decrease the existing risks to the extent that the remaining risk is below an acceptable risk to be defined [2].

The interlocking mechanisms described in chapter 'Control module functions' protect the plant or plant units from device-related fault states. These include all those fault states that are caused by a malfunction of the devices themselves, or by operating the device outside the permissible operating range (for example, overheating of a pump because of an undetected dry run). These fault states are device-specific and can be detected independent of process- and plant-related constraints.

By nature, the examined interlocking mechanisms cannot independently protect against process-related fault states (for example, the running dry of a pump). That is because they depend on process- and plant-related events (for example, dropping below a minimum tank level causes the pump to run dry). For this reason, plants must be safeguarded by implementing suitable process-related interlocks. These often utilize and expand the interlocking mechanisms of the control module functions (see chapter 'Control module functions'). All modes of intended plant operation must to be taken into account for this.

Intended operation refers to the operation for which the plant is intended and designed according to its technical purpose [2]. This usually includes the following operating modes:

– Normal operation

– Startup and shutdown operation

– Commissioning and decommissioning

– Test mode

– Inspection, maintenance and repair activities

A safety concept for the plant is initially developed within an interdisciplinary team for this. The team systematically identifies hazard potentials and faults that can lead to hazards. Established methods for hazard analysis are used, for example, the PAAG method [3].

The risks that result from the identified hazards must then be evaluated. Different methods for the tiered evaluation of the risk to be covered are available, for example, the ALARP method, the LOPA method and the method of the risk graph specified in [2]. If the initial risk of a hazard is greater than the specified acceptable risk, safety measures must be taken that reduce the risk accordingly.

4.4 Functional safety by means of process control engineering

In general, it is preferable to use safety devices that are not based on process control engineering means for functional safety. Often, however, due to the size or complexity of the plant, use of such devices is either inadequate or technically infeasible. The corresponding solution may also be economically infeasible. In this case, safety functions are implemented by means of process control engineering. For this reason, we distinguish between two types of process control systems:

Basic process control systems (BPCS) implement the automation functions required for production and thus also serve to operate the plant as intended within its OK range [2]. Process control monitoring devices react if one or more process variables leave the OK range. They signal permissible fault states or automatically takes steps to return the process variables back to the OK range. From the perspective of functional safety, no requirements are placed on basic process control systems.

Safety instrumented systems (SIS) serve to reduce the risk of identified hazard potentials. They do this by either preventing an event or acting in a way that limits the harm. The goal of safety instrumented systems is to prevent an impermissible fault state from occurring in the plant in the first place. They reduce the probability of occurrence of an undesired event, thereby reducing the risk connected with this event. Damage-limiting safety instrumented systems, on the other hand, aim to lessen the extent of harm of an undesired event after it occurs and thus to reduce the associated risk. Such control systems are only used very rarely.

Figure 1 shows the basic mode of functioning of process control systems within the framework of functional safety. Curve 1 shows a process variable that cannot reach the impermissible fault range for process-related reasons. For this reason, a process control monitoring device is sufficient here. In Curve 2, on the other hand, it is possible for the limit of the impermissible fault range to be exceeded. But because a non-process control safety device is present, a process control monitoring device is also sufficient in this case. In Curve 3, there is no such safeguarding of the plant. Therefore, a safety instrumented system is used to prevent the process variable from reaching the impermissible fault range.

For the process control systems of a plant, it must be clearly defined whether they implement a basic process control function or a safety instrumented function. This differentiation facilitates planning, setup and operation, but also the subsequent change of process control systems.

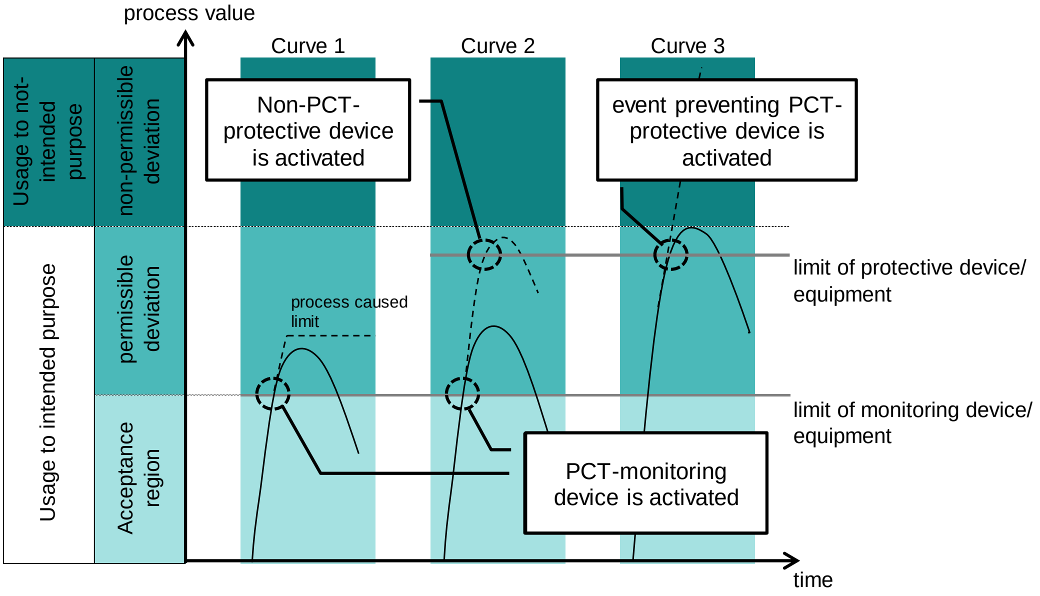


Figure 1: Schematic representation of mode of functioning of process control systems according to [2]

Because functions of safety instrumented systems are required only very rarely, their components may also sometimes be used by basic process control systems for economic reasons. In this case, signals for triggering the safety instrumented function must always take precedence over signals of the basic process control system.

Measures that are as simple, straightforward and direct-acting as possible must be used to implement the safety instrumented functions. It should be possible to directly acquire the utilized process variables with simple and proven methods. It follows that the control design itself is characterized by relatively low complexity.

4.5 Standard protective circuits for functional safety

The objective of safety instrumented systems with the resources of process control engineering is usually to control certain signal combinations, signal sequences, signal characteristics as a function of time or priorities of signals in a way that impermissible process states are prevented. The user realizes these functions with recurrent standard protective circuits. The most important standard protective circuits are presented below.

Combination circuits

In many cases, certain control signals are permissible only if the process is in a certain state. This state can be described as a combination of the corresponding process signals. To combine individual signals to form a switching condition, simple combination circuits can be used. They have the ability to determine the state of an output signal at any time through the state of a set of input signals. To this end, the input signals are logically combined with the logical operations AND, OR and NOT. The combination circuits themselves are stateless, which means they have no storage properties.

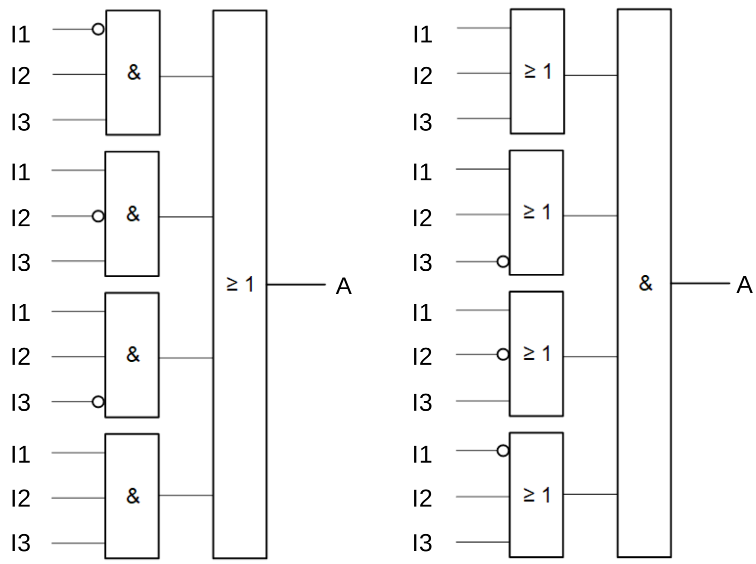
The relationship between input and output signals can be described completely with a function table. The corresponding logic function is always be represented in (at least) two standardized forms.

Disjunctive normal form (DNF) In this representation, the user first defines all combinations of the inputs for which the output signal is to be set (all lines of the function table for which A = 1). These combinations are represented as AND operations of the input signals. The outputs of these AND operations are then connected to each other by means of an OR operation. As a result, the output is set as soon as one of the located combinations occurs.

Conjunctive normal form (KNF): In this representation, the user first defines all combinations of the inputs for which the output signal is not to be set (all lines of the function table for which A = 0). These combinations are inverted and represented as OR operations of the input signals. The user connects the outputs of these OR operations to each other by means of an AND operation. Inverting the located combinations has the effect that the output is set only if none of these combinations occur.

Figure 2 shows an example of a function table with three input signals and the corresponding combination circuits in the disjunctive and conjunctive normal forms.

I1	I2	I3	O
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1



Functions table

Disjunctive normal form (DNF)

Conjunctive normal form (CNF)

Figure 2: Structure of basic combination logic circuits

Prioritization circuits

Safety functions must always take precedence over operator control and monitoring functions. In this case, multiple control signals control the behavior of an actuating signal. For this reason, the control signals have to be prioritized accordingly. In most cases, prioritization is static, and is implemented using a combination circuit.

Latching circuits

It is not always possible to represent the conditions for an output state by the current state of the inputs alone. If, for example, output signal Q is to be set by the input signal I1 and reset by another input signal I2, this can no longer be represented combinational. Q must also remain set when I1 is reset. Only when I2 is set is O to be reset. This makes the effect of I2 dependent on whether I1 was previously set, and thus on the current Out state of the system. This state must be stored in the circuit. These latching circuits also called sequential circuits. Storing the Out state can be implemented using a reset-set flip-flop (RS flip-flop).

As shown in Figure 3, such a circuit has two inputs: one input for setting (S) and one input for resetting (R) the output. It is important here to define how the output is to be switched when both inputs are set. Depending on the implementation of the RS flip-flop, either setting or resetting is dominant (refer to Figure 3).

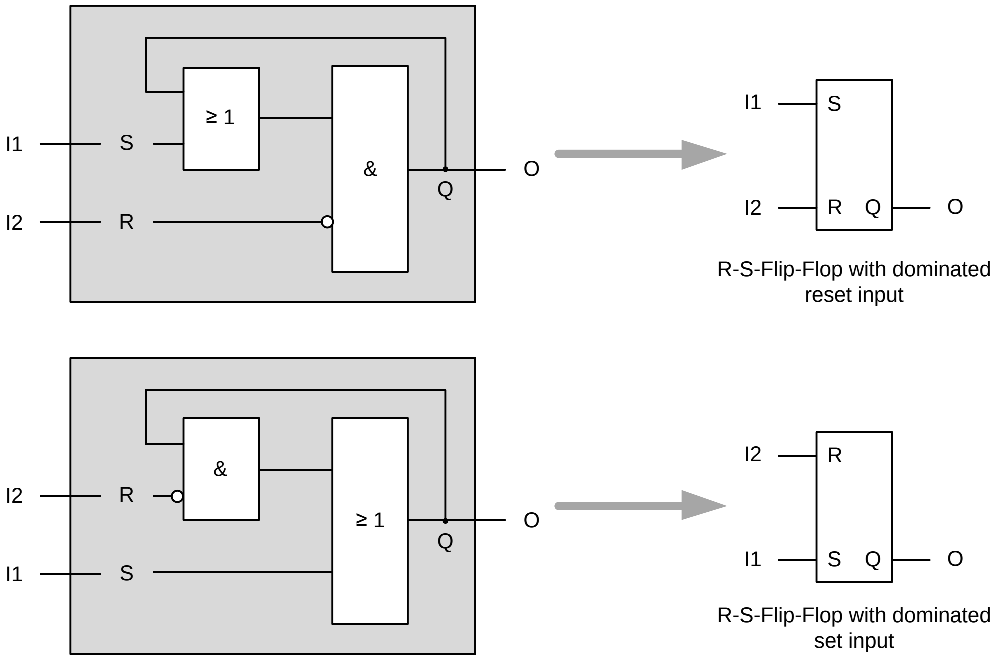


Figure 3: Design and function icons of RS flip-flops

Interlocking circuits

Often, attention must be paid that certain control signals are not set simultaneously. For example, an electric motor with two directions of rotation must not be simultaneously switched to run forward and in reverse. The two signals F (forward) and R (reverse) must be mutually interlocked.

An interlock is realized using two interconnected RS flip-flops. Two interconnection possibilities exist. The interlock takes place via either the set inputs or the reset inputs. Both variants are shown in Figure 4. Note that the interlocking via the reset input only works if the reset input is dominant.

I1	O1 set
I2	O1 reset
I3	O2 set
I4	O2 reset

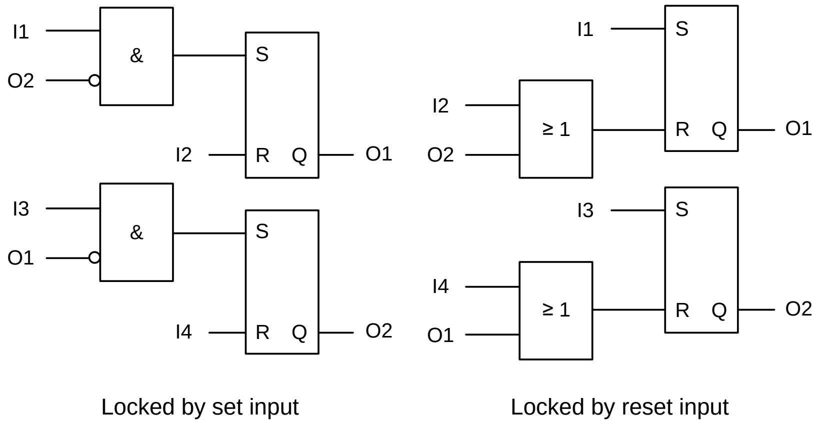


Figure 4: Mutual interlocking of two output signals

In some cases, the sequence in which specific control signals can be set must also be specified. A sequence interlock is implemented in this case. This can also be implemented by a stinging together of flip-flops. You need as many RS flip-flops as steps that are to be coordinated. Figure 5 shows a sequence interlock for two signals.

I 1	O1 set
I 2	O1 reset
I 3	O2 set
I 4	O2 reset
O2 only, if O1	

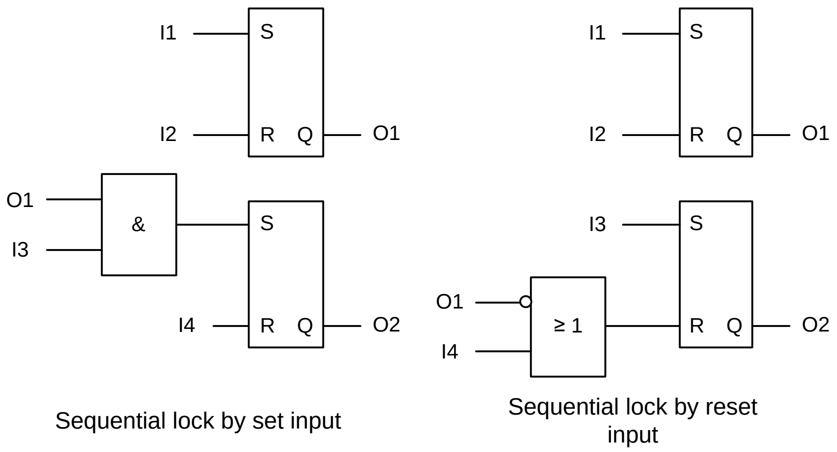


Figure 5: Sequence interlock of two output signals

Note here that only activation sequences are implemented with these circuits and not signal sequences. Setting O2 does not cause O1 to be reset. In case of an interlock via the reset input, O2 is also reset automatically when O1 is reset.

Circuits with timing behavior

Circuits with timing behavior also take into account the time since the occurrence of one or more events. This principle is explained below using the ***two-hand interlock*** as an example. The purpose of this interlock is to prevent workers from being injured when operating a machine, such as a press. It can only be released by simultaneous actuation of two pushbuttons, which prevents the worker from still having one hand in the danger zone of the machine. This task can also be solved by using a combination circuit. To prevent one pushbutton from being permanently fixed with adhesive tape, however, it must also be ensured that both buttons are pressed within a fixed time span. To this end, pulse elements are used that set the output signal for a specified time and then reset it automatically, regardless of the time duration of the set input signal. Only a state change of the input (from reset to set) generates a reset of the output signal. Figure 6 shows the function icon and the switching behavior of a pulse element.

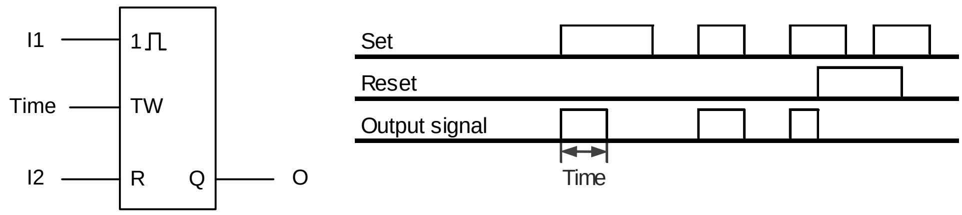


Figure 6: Function icon and switching behavior of a pulse element

The corresponding circuit for a two-hand interlock is shown in Figure 7. If one of the pushbuttons is actuated, output Out of the pulse element is set for duration T. If the second button is then actuated while Out is set, all conditions of the AND element are met, and output O is set. The pulse element is bypassed through the OR operation with output O.

I 1	Button left hand
I 2	Button right hand
T	Actuating time

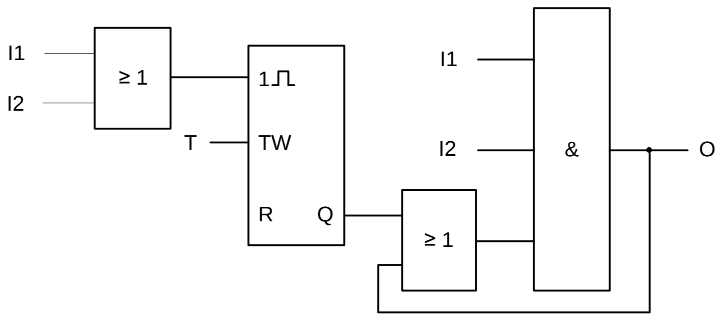


Figure 7: Two-hand interlock when using a pulse element

Timing elements are also used for many other safety functions. For example, they are used for safety gate controls in which open gates close automatically after a specified time and for motor startup controls in which, after a futile start attempt, a rest period for drive recovery is forced.

4.6 References

[1] Strohrmann, G. (1983): Anlagensicherung mit Mitteln der MSR-Technik, Oldenbourg Verlag

[2] VDI 2180 (Edition 2018-02): Functional safety in the process industry – Planning, erection and operation of safety instrumented systems

[3] DIN EN 61511 (Edition 2019-02): Functional safety – Safety instrumented systems for the process industry sector.

5 Task

According to the requirements in chapter 'Process description', manual operation is to be added for the pump motor =SCE.A1.T2-P001 in the CFCs from chapter 'Control module functions'. The following interlock conditions must be observed:

- The pump motor may only be switched on when the main switch of the plant is switched on and the Emergency Off switch is unlocked.
- The pump must not take in air. The minimum level (here: 50 ml) in reactor =SCE.A1.T2-R002 is numerically known and will be evaluated using the measured level.
- The pump must not pump liquid against a closed valve. When the pump is switched on, valve=SCE.A1.T3-V001, valve=SCE.A1.T2-V007 or valve =SCE.A1.T4-V003 must be open.

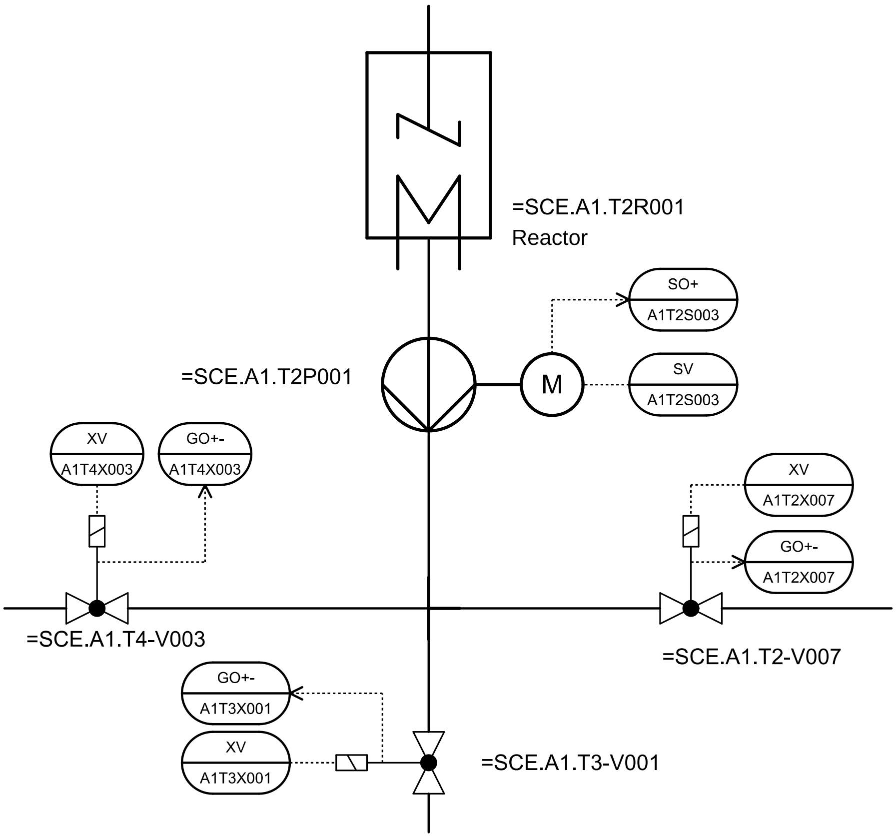


Figure 8: Section of the P&ID flow chart of the reactor pump with its connections

Note:

- *For the approach to a solution, please note the details regarding latching circuits in the theory section.*

6 Planning

The manual control A1T2H011 (see Figure 10) for emptying reactor R001 consists of three parts:

– A digital input for the Start command: 'A1.T2.A1T2H011.HS+.START' / I7.2

– A digital input for the Stop command: 'A1.T2.A1T2H011.HS-.STOP' / I7.3

– A digital output for the status feedback: 'A1.T2.A1T2H011.HO+-.0+' / Q4.2

These are already contained in the symbol table and only still have to be combined.

The manual control influences both pump A1T2S003 and valve A1T3X001 (see also Figure 9), which must be expanded accordingly.

The interlocks mentioned in the task description can all be implemented with previously created sensors and actuators.

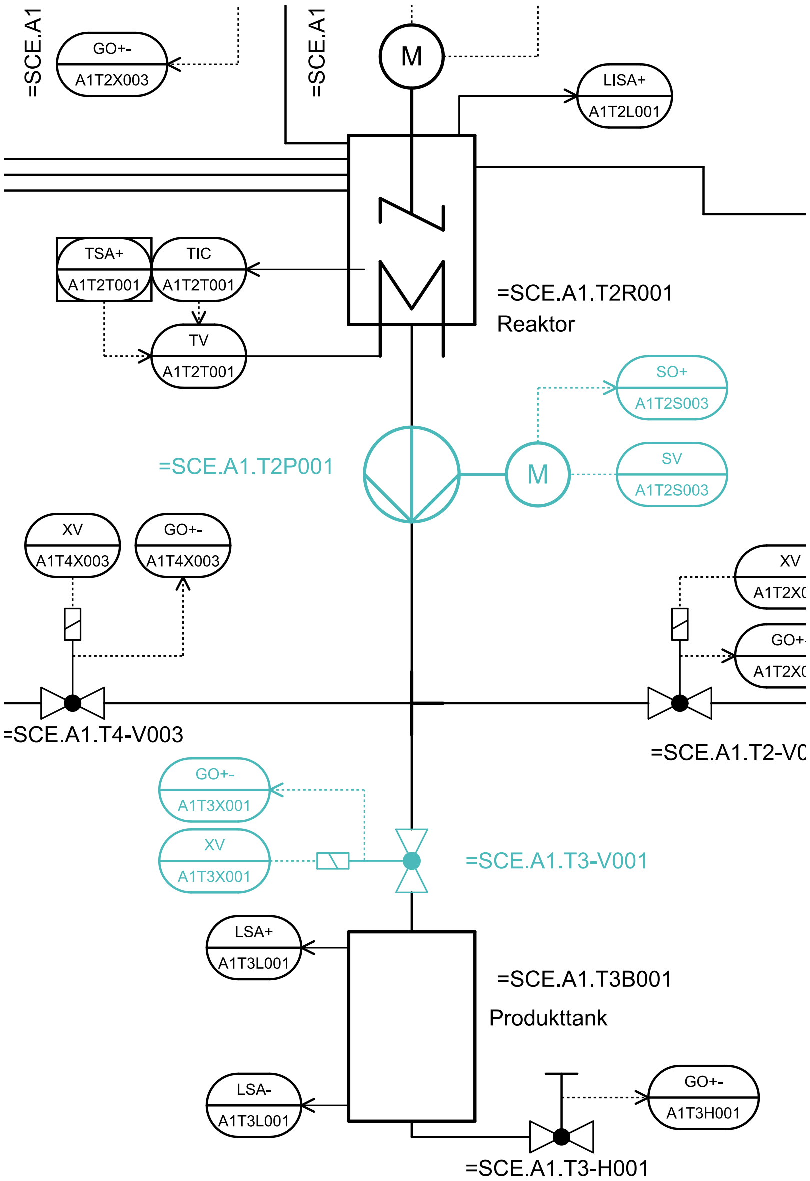


Figure 9: Portion of the P&ID flow chart to be programmed

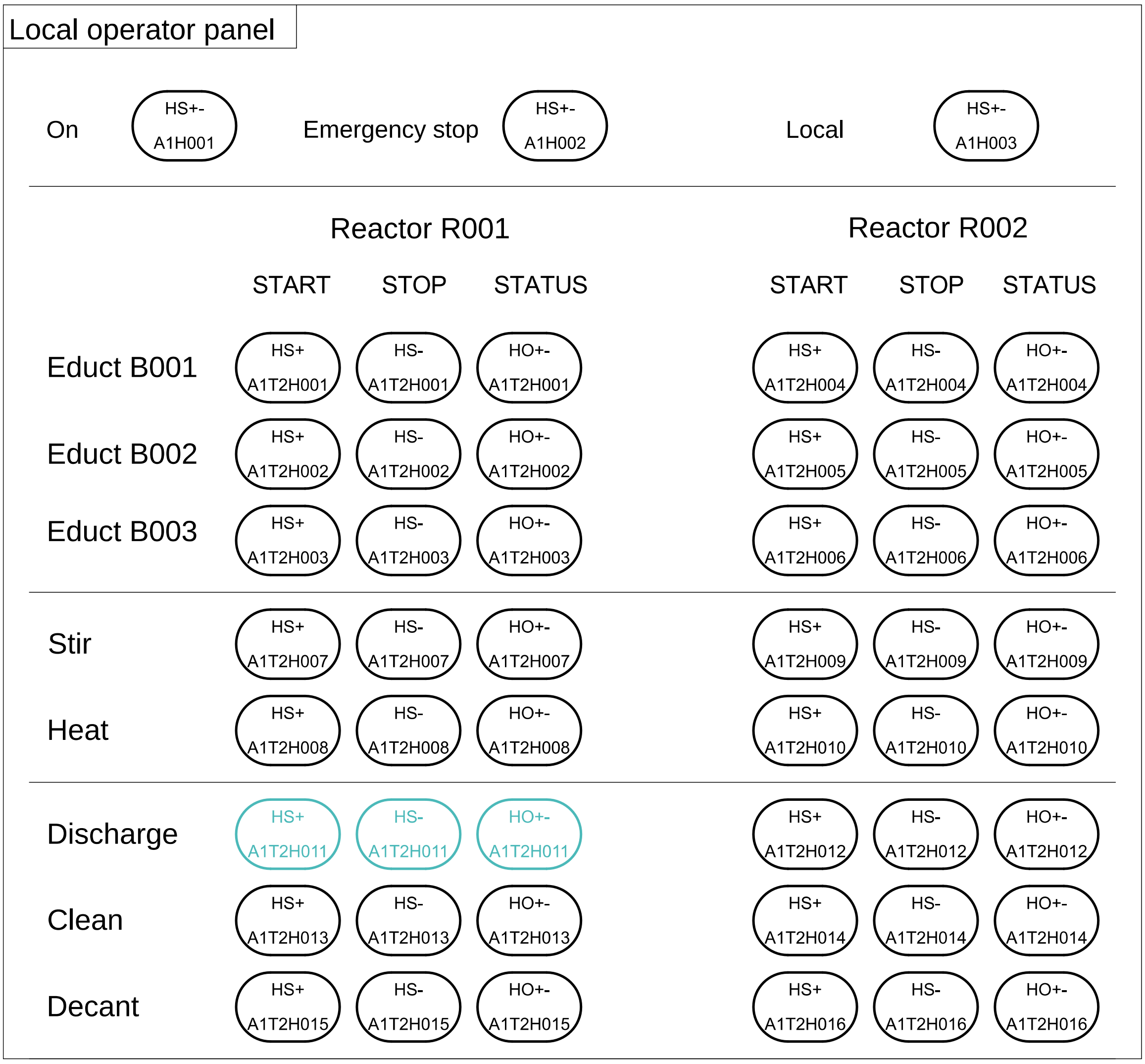


Figure 10: Local operator station

7 Learning objective

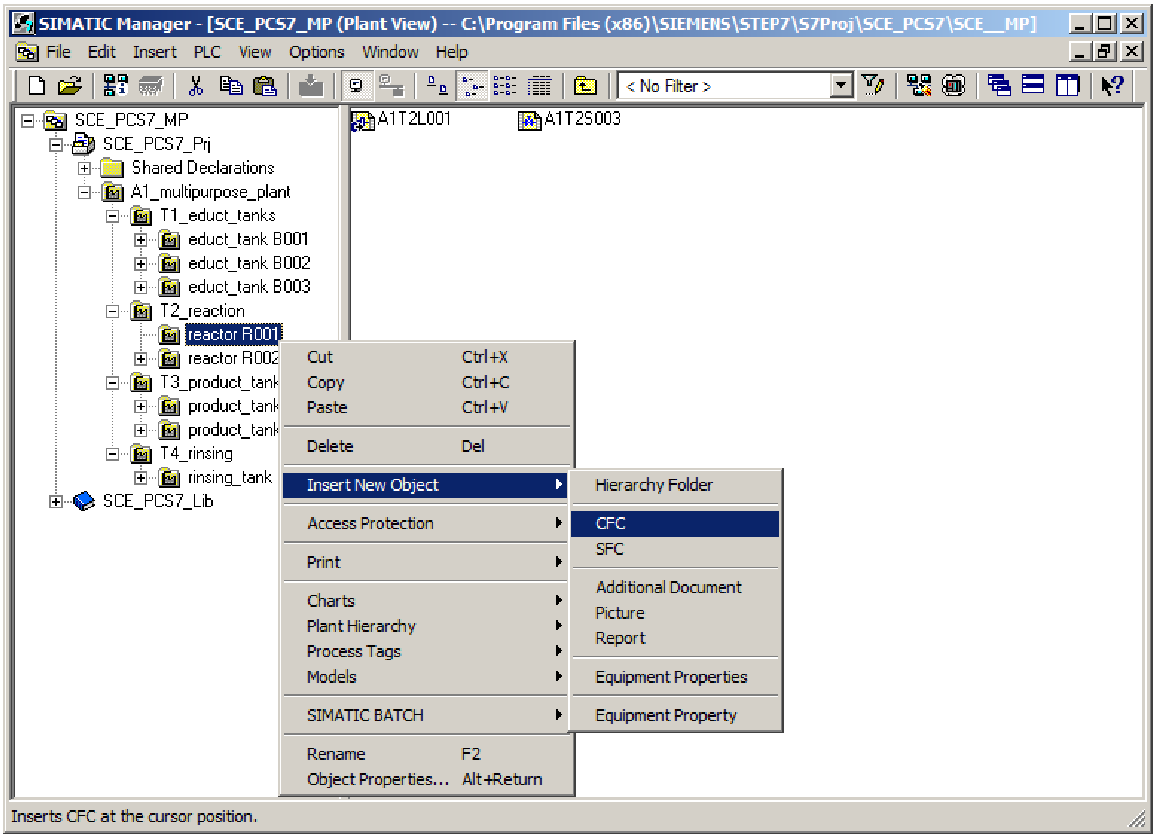
In this chapter, students learn the following:

- Implementation of advanced constraints and the manual operation
- Creation of interconnections between CFCs
- Additional possibilities for programming with CFCs
- Use of additional sheets in the CFCs
- Testing the program using the control functions in the CFC

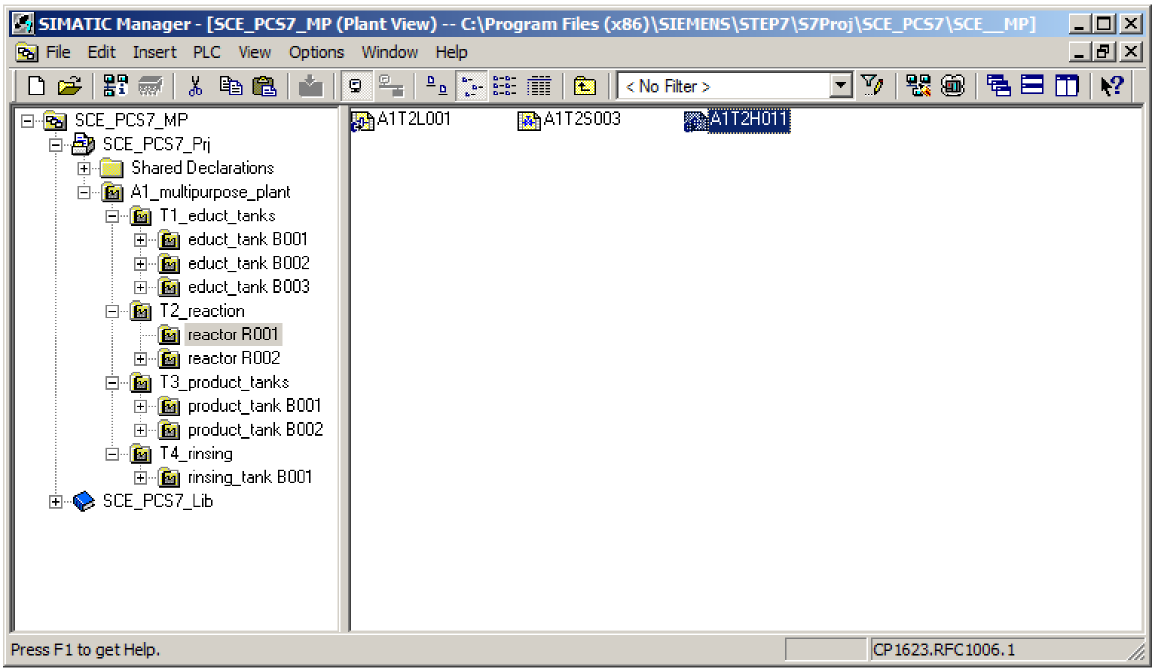
8 Structured step-by-step instructions

8.1 Creating the manual operation A1T2H011

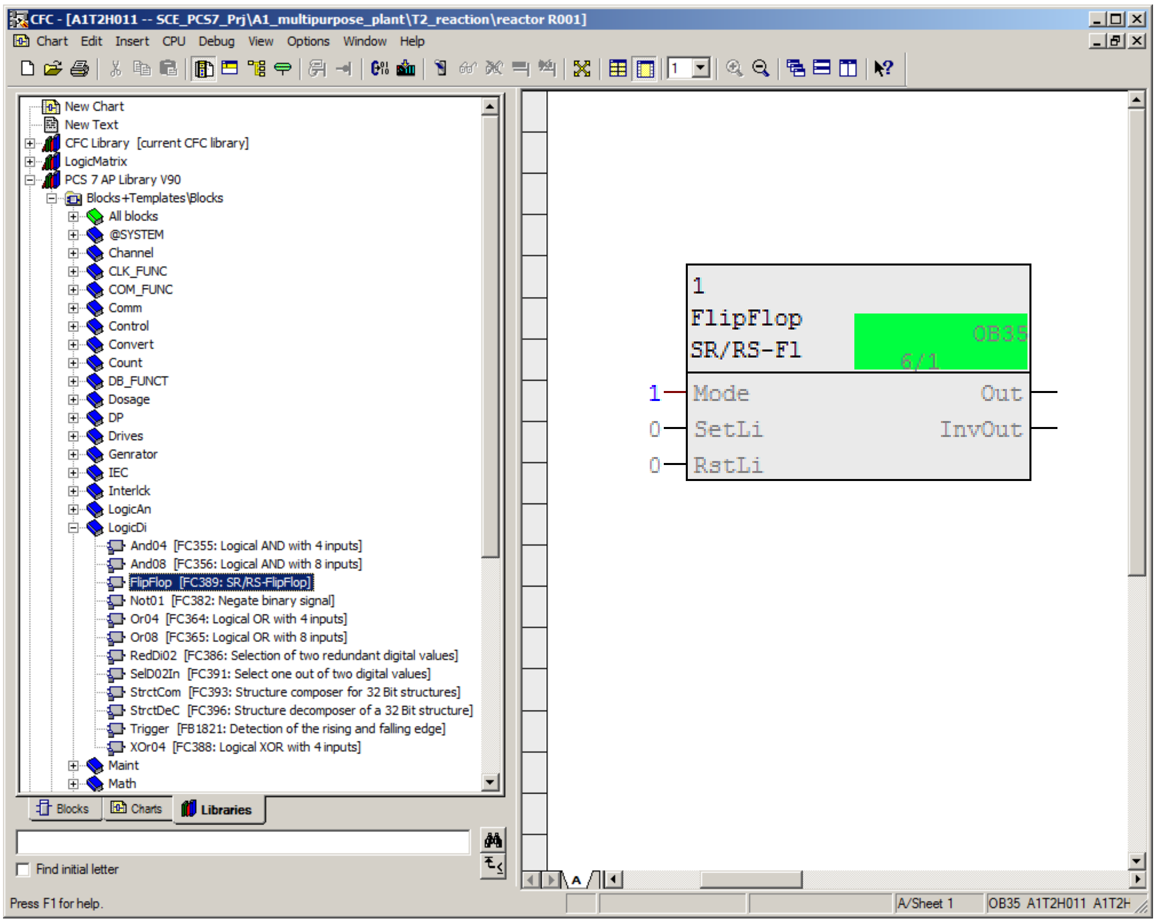
1. Rename the chart to 'A1T2H011' and open it with a double-click. (→ A1T2H011) To program the manual operation for draining reactor R001, create a new CFC in the plant view of SIMATIC Manager in the Reactor R001 folder of the T2_reaction unit. (→ SIMATIC Manager → View → Plant View → reactor R001 → Insert New Object → CFC)



2. Rename the chart to 'A1T2H011' and open it with a double-click. (→ A1T2H011)



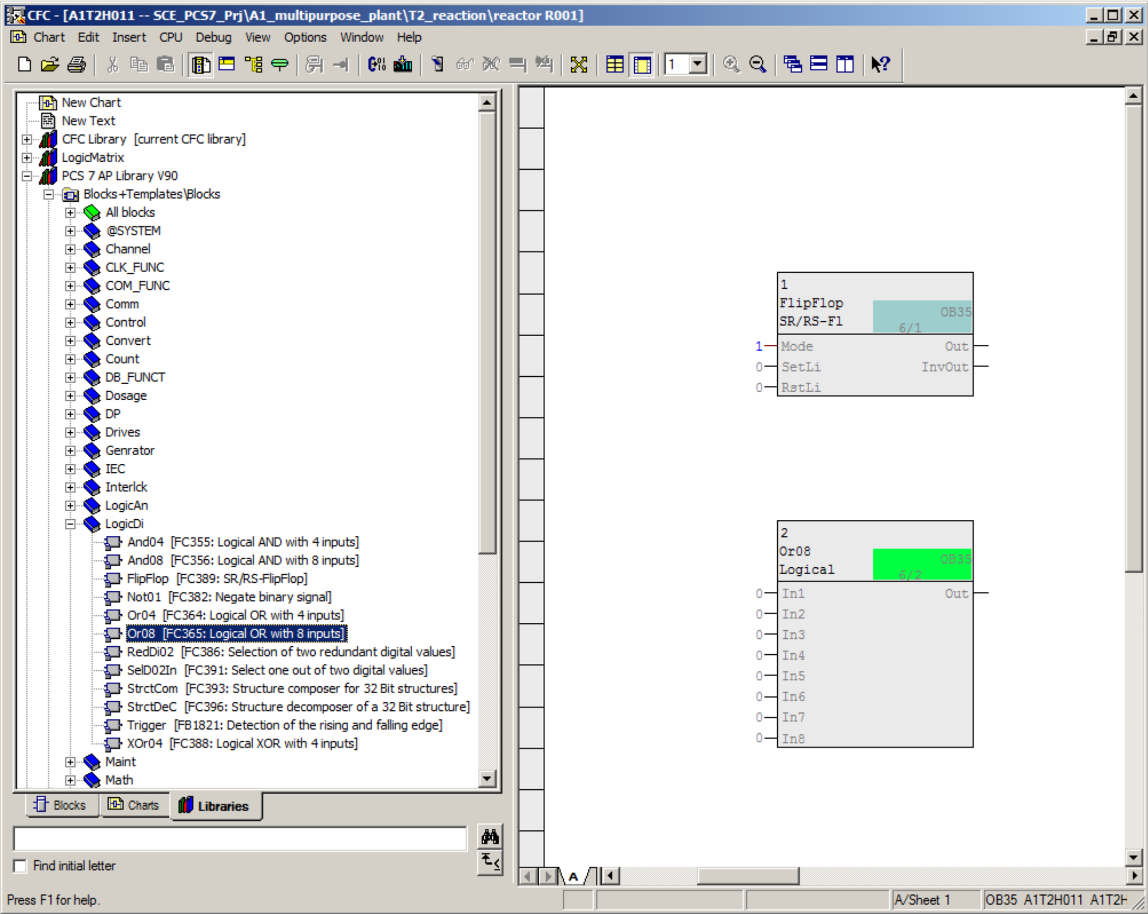
3. In the CFC Editor, drag the 'FlipFlop' block from the 'LogicDi' folder in the 'Blocks' catalog to the first sheet of the chart. You now have a storage element. For the reset or switch-off to be dominant, you must set the mode to '1'. (→ Libraries → PCS 7 AP Library V90 → Blocks+Templates\Blocks → LogicDI → FlipFlop → Mode → Value: 1)



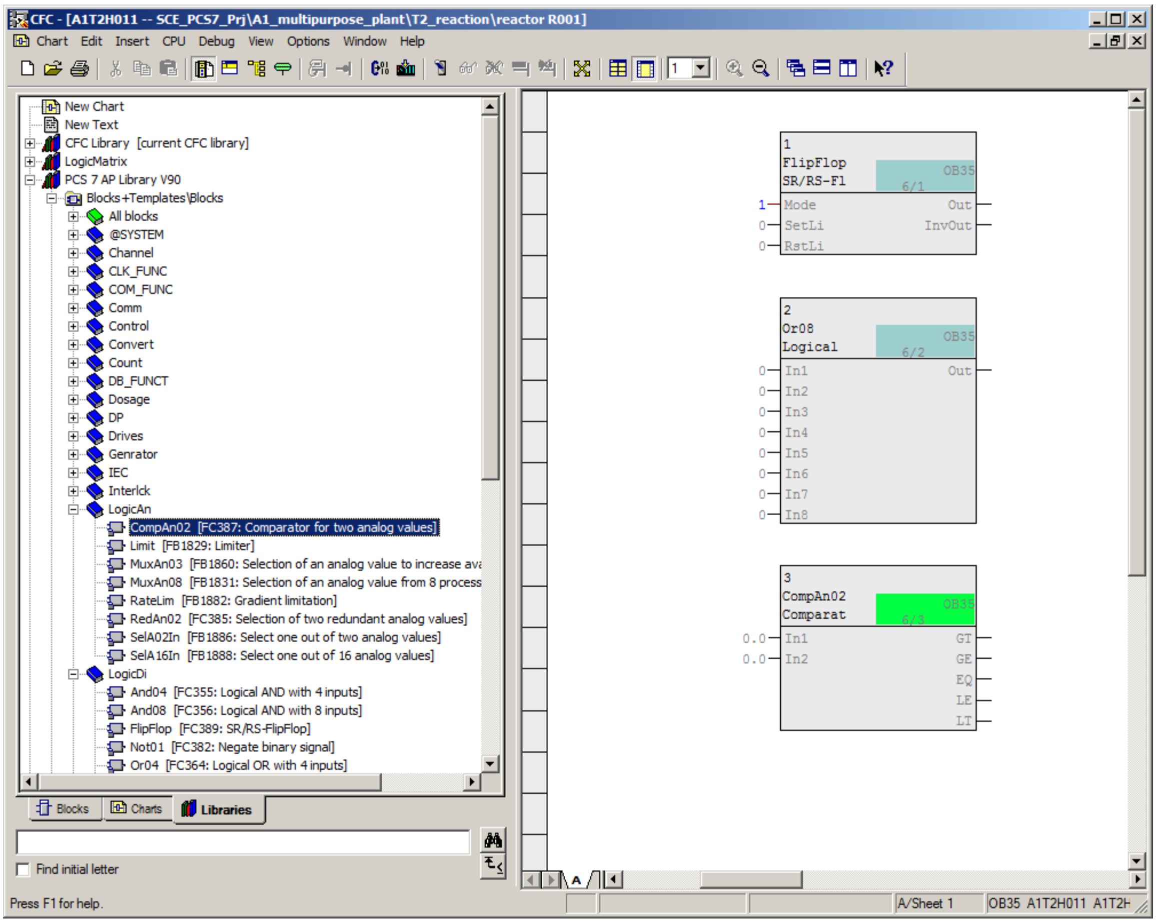
Note:

– *Additional information about the utilized blocks is provided in the detailed online help. To access the help, select the corresponding block and press 'F1' on the keyboard.*

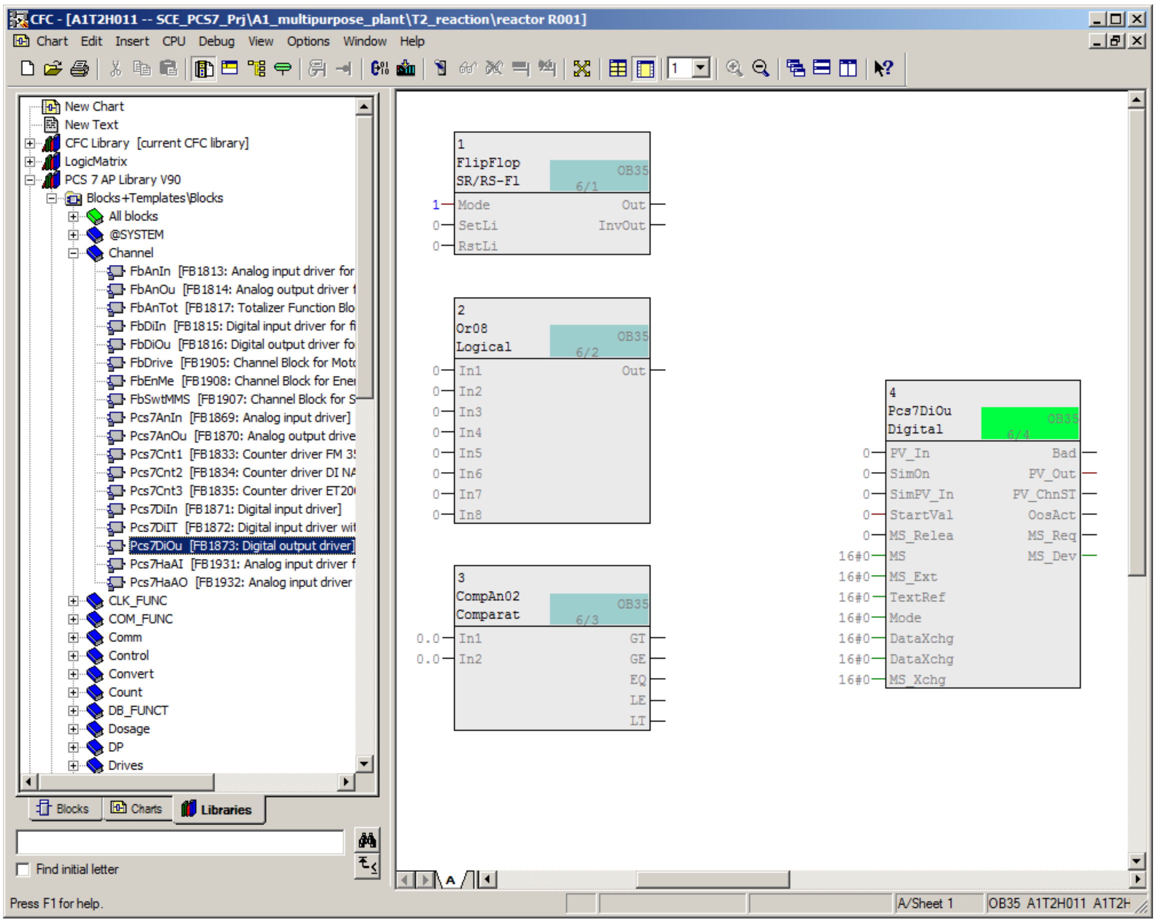
4. Next, drag the 'Or08' block from the 'LogicDi' folder to the chart. (→ Libraries → PCS 7 AP Library V90 → Blocks+Templates\Blocks → LogicDI → Or08)



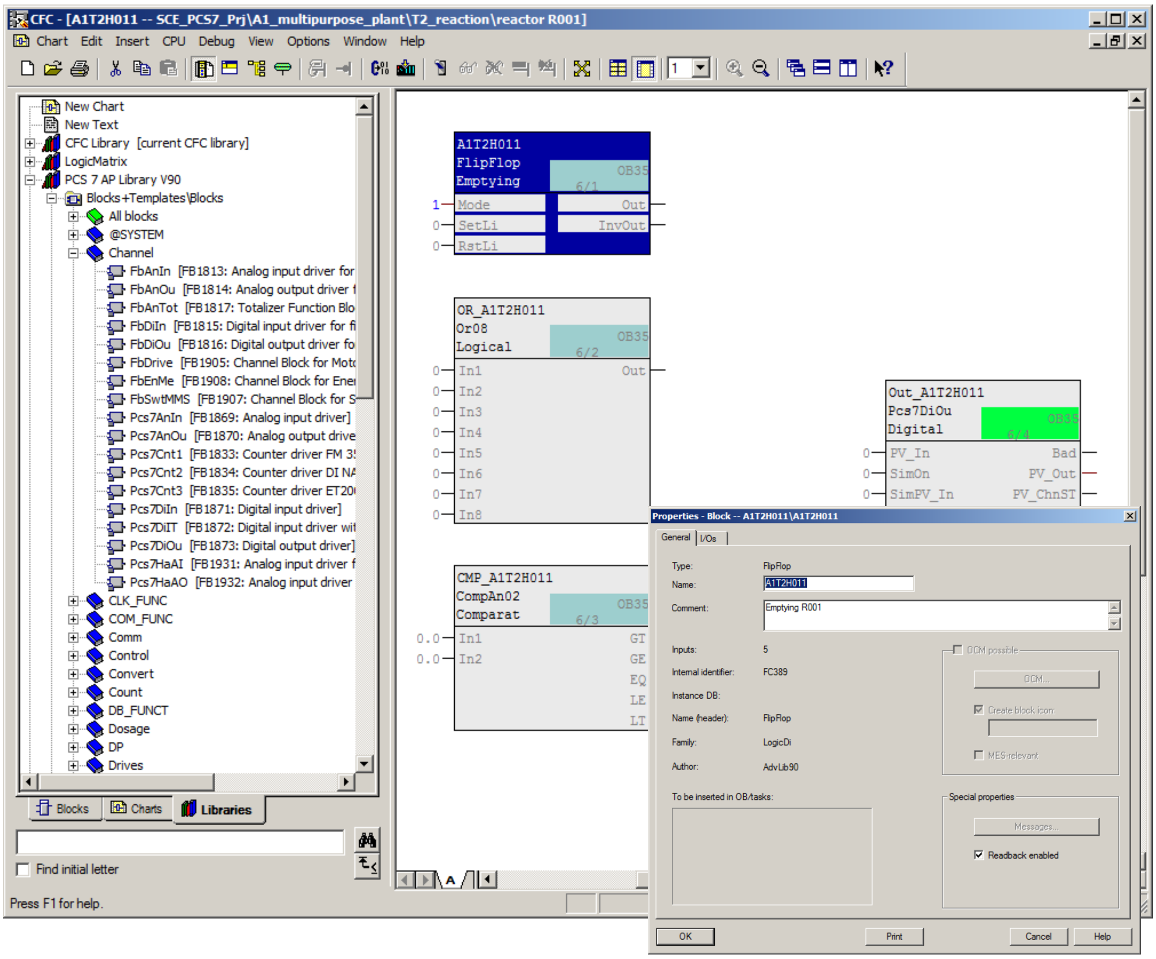
5. Now, drag the 'CompAn02' block from the 'LogicAn' folder in the PCS 7 AP Library V90 in the 'Libraries' catalog to the chart. You will need it in order to take into account the numerical value of the level of reactor R001 for the interlock. (→ Libraries → PCS 7 AP Library V90 → Blocks+Templates\Blocks → LogicAn → CompAn02)



6. Then, drag the driver block for the digital output signal 'Pcs7DiOu' to the chart. (→ Libraries → PCS 7 AP Library V90 → Blocks+Templates\Blocks → Channel →PCS7DiOu)

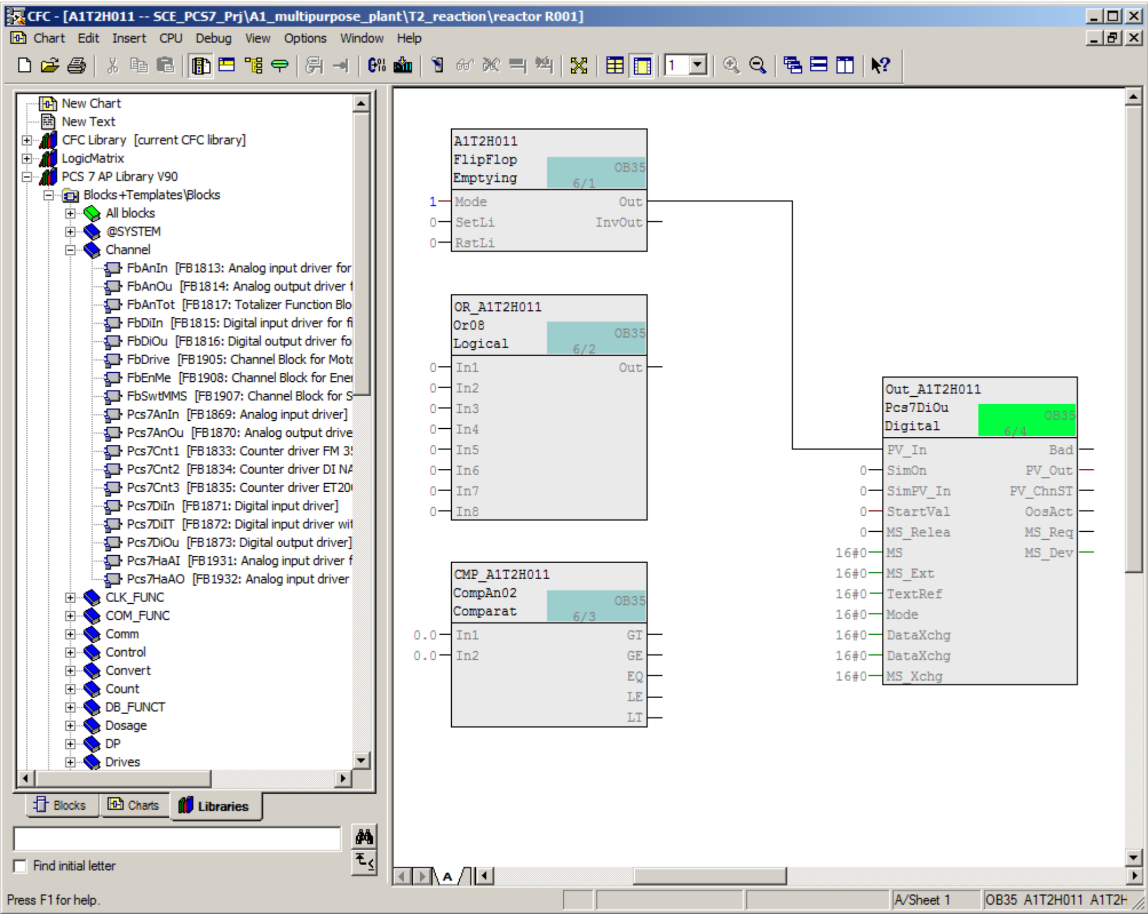


7. Assign names to the blocks as shown. (→ Block → Object properties → Change name → OK)

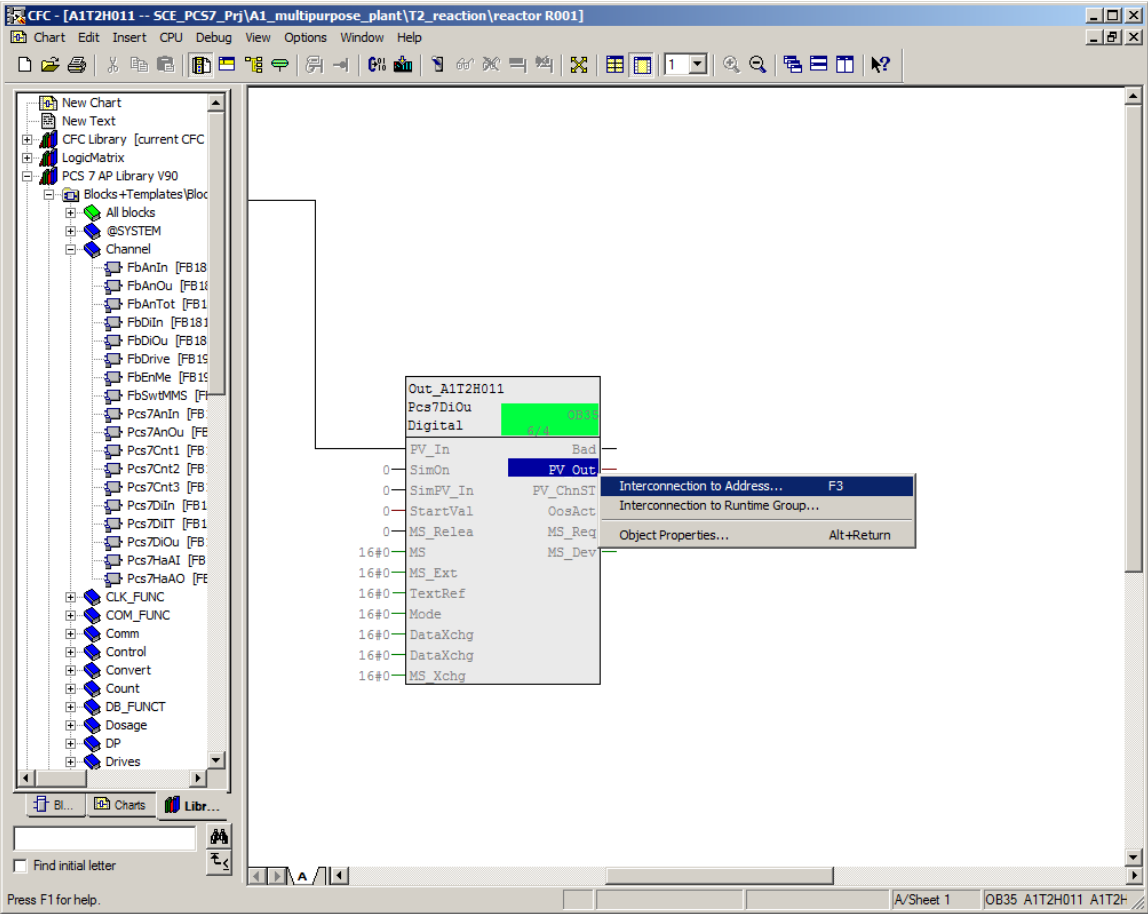


Block	Name
FlipFlop	A1T2H011
Or08	OR_A1T2H011
CompAn02	CMP_A1T2H011
Pcs7DiOu	Out_A1T2H011

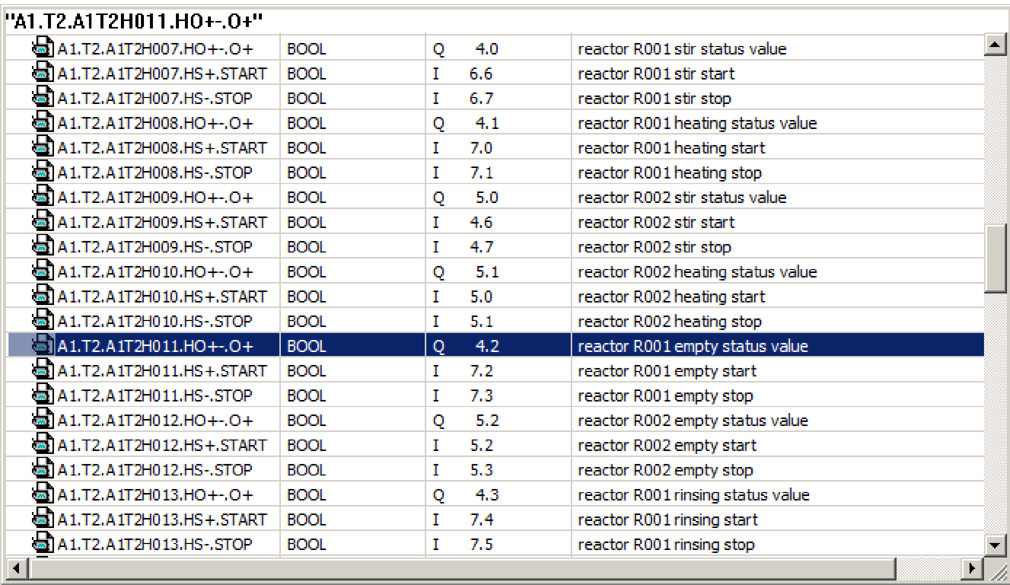
8. Now, you start interconnecting the blocks with one another. To do so, click the 'Out' output of the 'FlipFlop' block and then the 'PV_In' input of the 'Pcs7DiOu' block. The layout of the line of this interconnection is created automatically and cannot be changed in the CFC Editor. (→ FlipFlop → Out → Pcs7DiOu → PV_In)



9. To allow the status of the operator prompt to be displayed, interconnect the 'PV_Out' output of the 'Pcs7DiOu' block with the corresponding address from the symbol table. (→ Pcs7DiOu → PV_Out → Interconnection to Address)



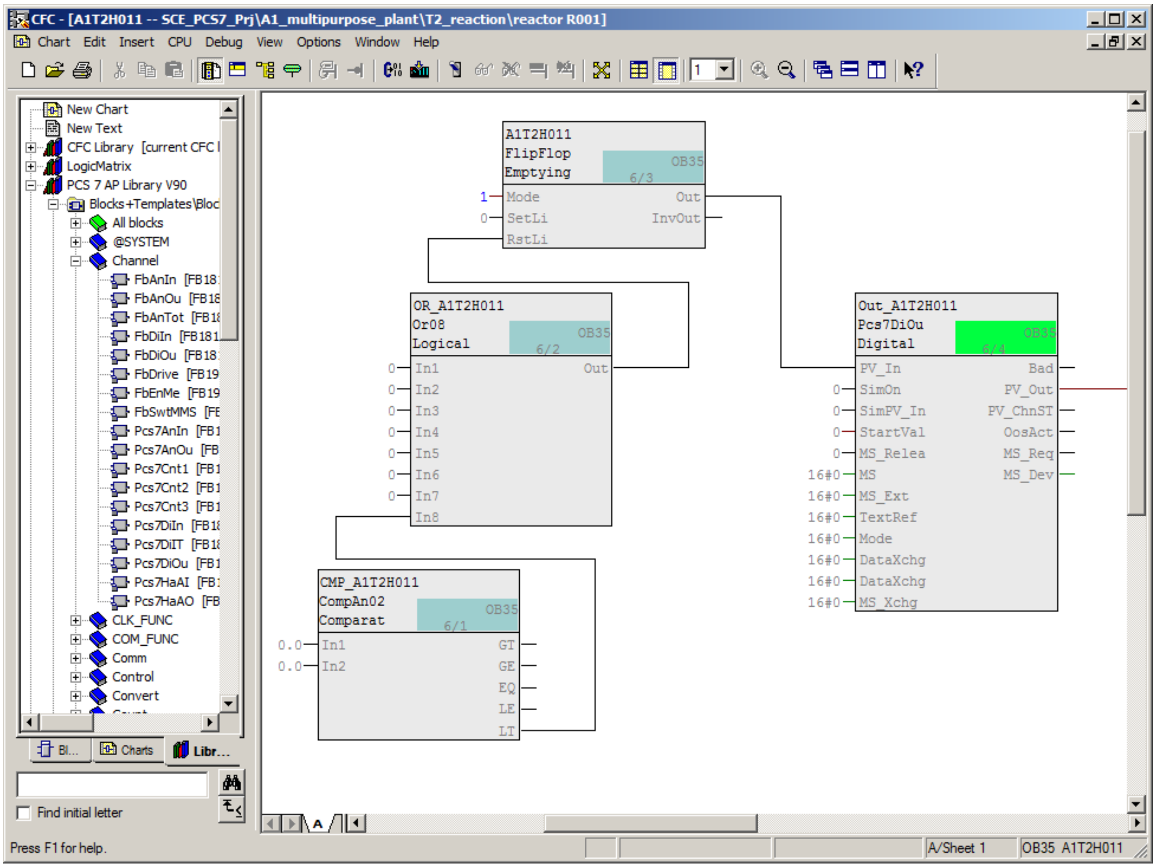
10. From the symbol table that is then displayed, select output Q 4.2 "A1.T2.A1T2H011.HO+-.O+" for the status display of the operator prompt. (→ A1.T2.A1T2H 011.HO+-.O+)



"A1.T2.A1T2H011.HO+-.O+"

A1.T2.A1T2H007.HO+-.O+	BOOL	Q	4.0	reactor R001 stir status value
A1.T2.A1T2H007.HS+.START	BOOL	I	6.6	reactor R001 stir start
A1.T2.A1T2H007.HS-.STOP	BOOL	I	6.7	reactor R001 stir stop
A1.T2.A1T2H008.HO+-.O+	BOOL	Q	4.1	reactor R001 heating status value
A1.T2.A1T2H008.HS+.START	BOOL	I	7.0	reactor R001 heating start
A1.T2.A1T2H008.HS-.STOP	BOOL	I	7.1	reactor R001 heating stop
A1.T2.A1T2H009.HO+-.O+	BOOL	Q	5.0	reactor R002 stir status value
A1.T2.A1T2H009.HS+.START	BOOL	I	4.6	reactor R002 stir start
A1.T2.A1T2H009.HS-.STOP	BOOL	I	4.7	reactor R002 stir stop
A1.T2.A1T2H010.HO+-.O+	BOOL	Q	5.1	reactor R002 heating status value
A1.T2.A1T2H010.HS+.START	BOOL	I	5.0	reactor R002 heating start
A1.T2.A1T2H010.HS-.STOP	BOOL	I	5.1	reactor R002 heating stop
A1.T2.A1T2H011.HO+-.O+	BOOL	Q	4.2	reactor R001 empty status value
A1.T2.A1T2H011.HS+.START	BOOL	I	7.2	reactor R001 empty start
A1.T2.A1T2H011.HS-.STOP	BOOL	I	7.3	reactor R001 empty stop
A1.T2.A1T2H012.HO+-.O+	BOOL	Q	5.2	reactor R002 empty status value
A1.T2.A1T2H012.HS+.START	BOOL	I	5.2	reactor R002 empty start
A1.T2.A1T2H012.HS-.STOP	BOOL	I	5.3	reactor R002 empty stop
A1.T2.A1T2H013.HO+-.O+	BOOL	Q	4.3	reactor R001 rinsing status value
A1.T2.A1T2H013.HS+.START	BOOL	I	7.4	reactor R001 rinsing start
A1.T2.A1T2H013.HS-.STOP	BOOL	I	7.5	reactor R001 rinsing stop

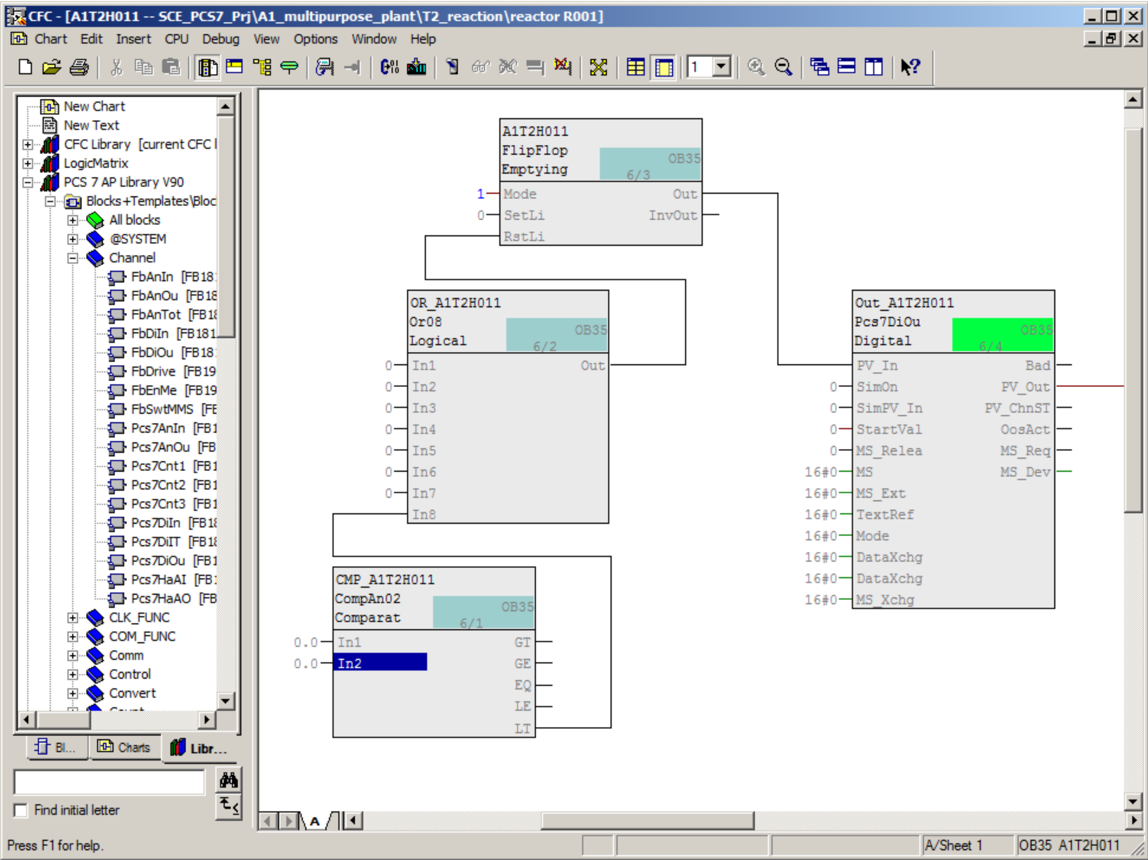
11. Continue interconnecting the blocks with one another. Connect the output of the 'Or08' block to the 'RstLi' input of the 'FlipFlop' block. Then, connect the 'LT' output of the 'CompAn02' block to an input of the 'Or08' block. (→ Or08.Out → FlipFlop.RstLi → CompAn02.LT → Or08.In8)



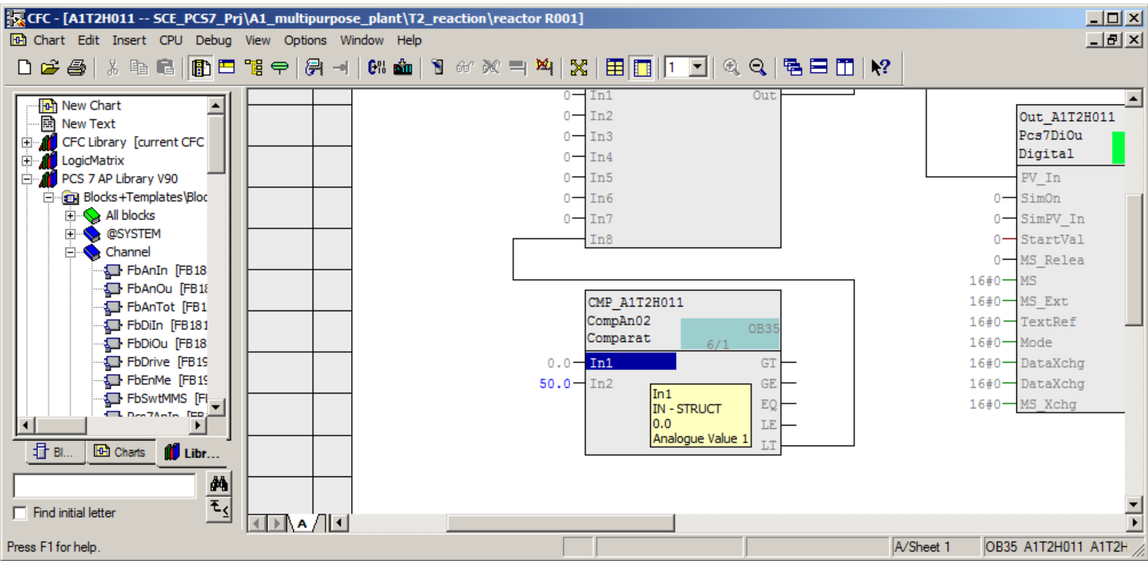
Note:

– *The 'LT' output of the 'CompAn02' block has the state 1 if 'In1' is less than 'In2'.*

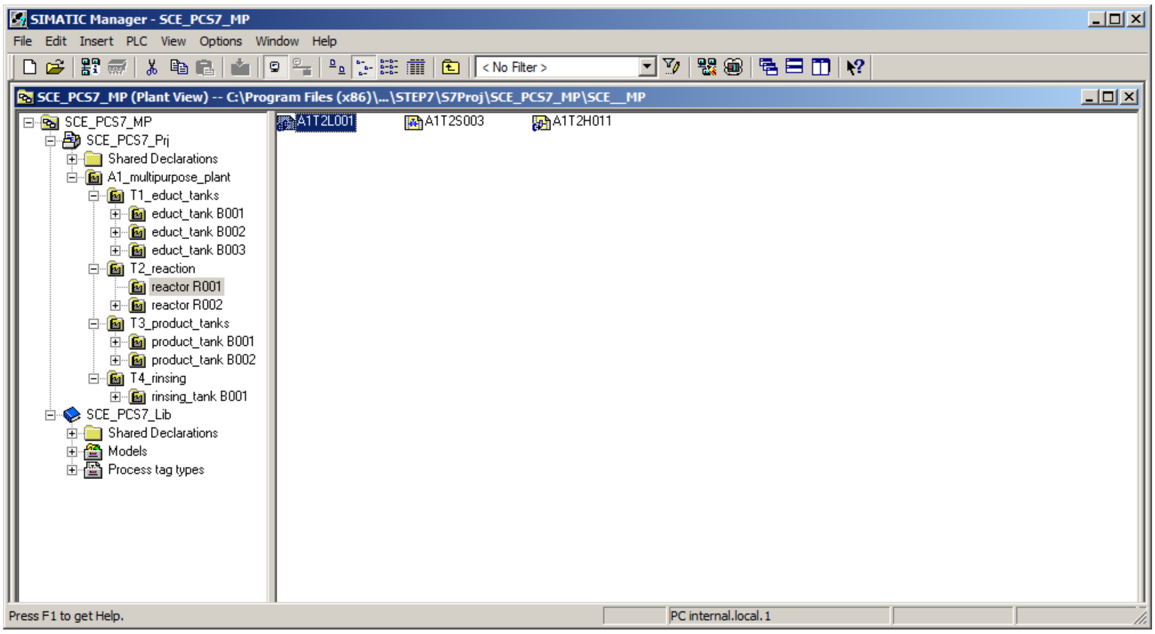
12. The comparison value is set at input 'In2' by opening the properties with a double click. Enter 50.0 as the value and apply this change with OK. (→ CompAn02 → In2 → Value → Value: 50.0 → OK → Close)



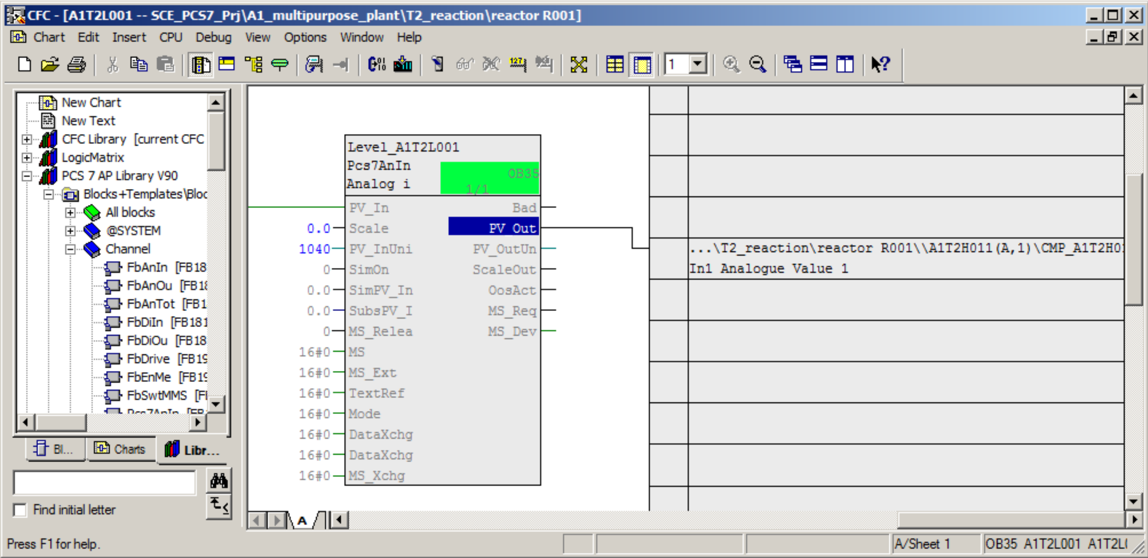
13. Next, create a cross-chart interconnection of input 'In1' with the measured level of reactor =SCE.A1.T2.R001. To do this, select 'In1' on the 'CompAn02' block. (→ CompAn02 → In1)

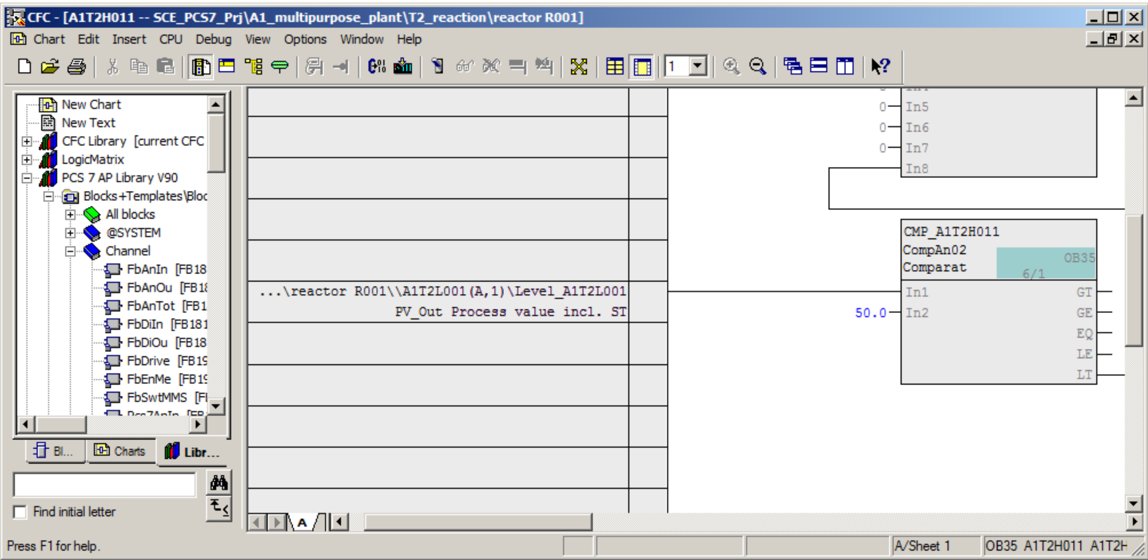


14. Then, open the CFC 'A1T2L001' in the plant view with a double-click. (→ SIMATIC Manager → Plant View → A1T2L001)

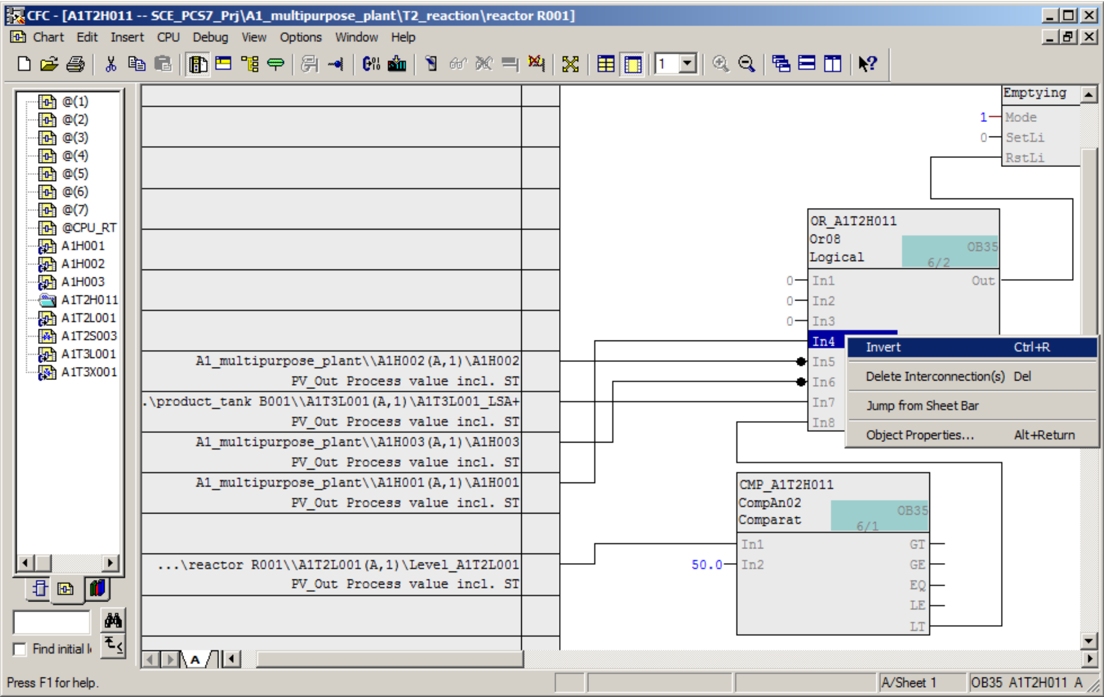


15. In the open chart 'A1T2L001', click the 'PV_Out' output of the 'Pcs7AnIn' block. The cross-chart link is created and displayed for both charts on the sheet bar. In the chart 'A1T2L001', the destination of the interconnection is shown on the right. In chart 'A1T2H011' the source of the interconnection is shown on the left. (→ A1T2L001 → Pcs7AnIn → PV_Out)

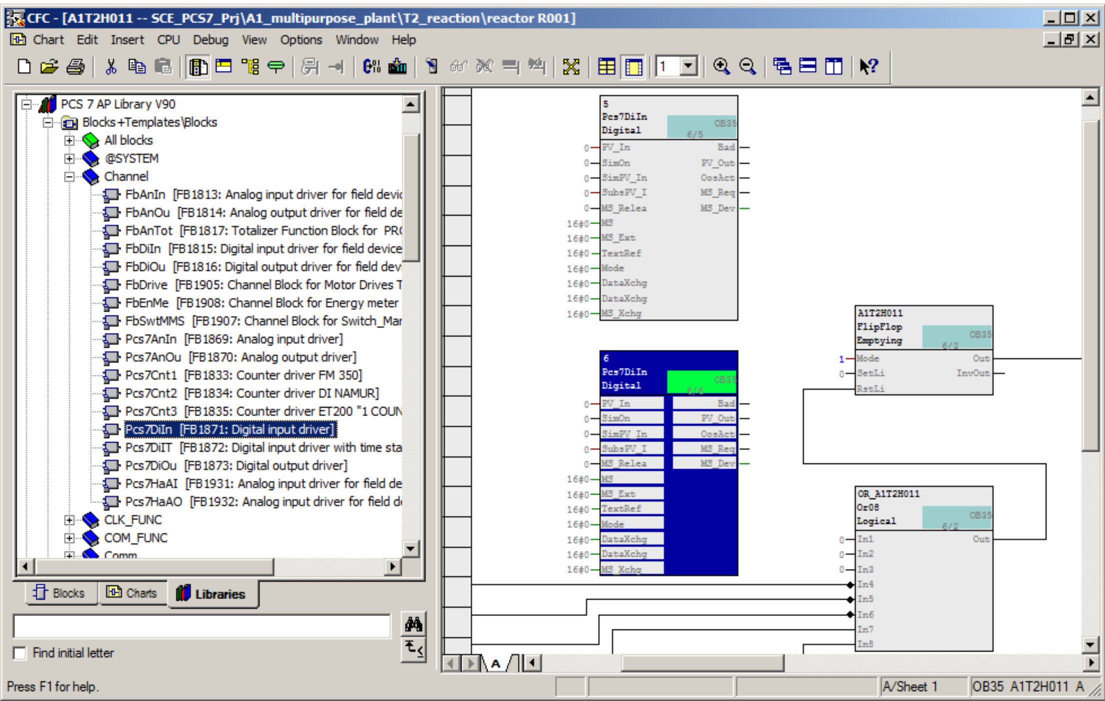




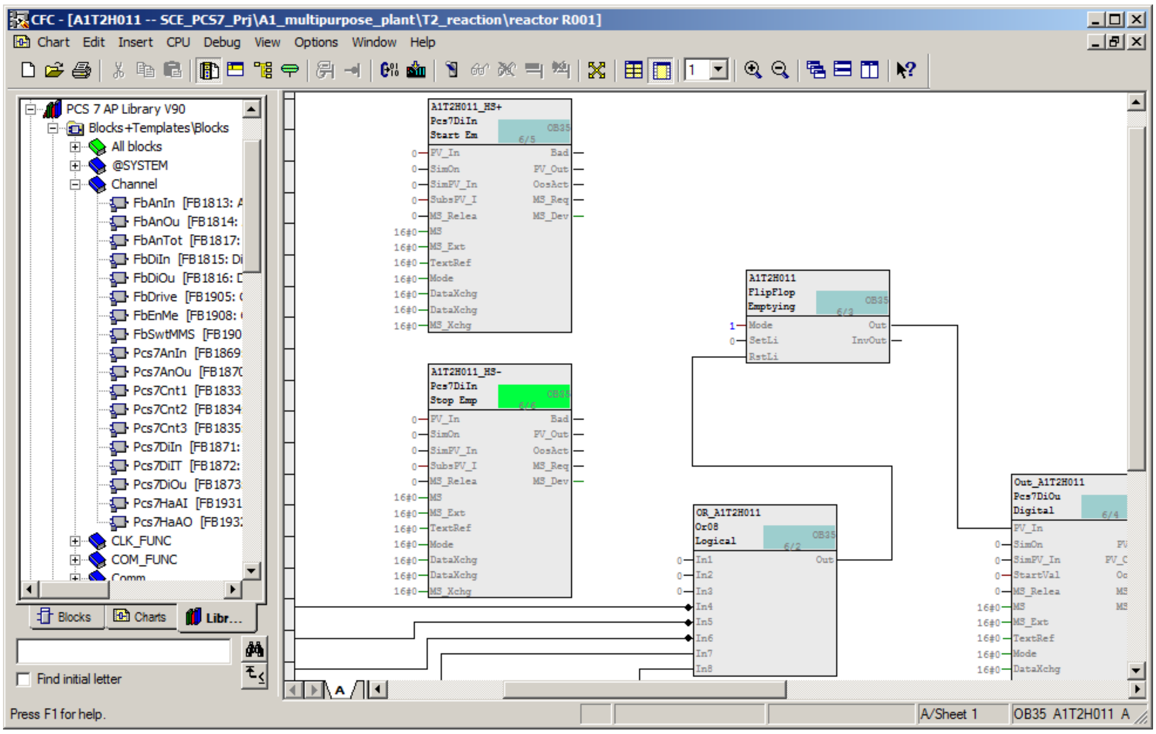
16. The signals that require the reset must be connected to the 'Or08' block. These signals are shown below and are also listed in Table 1 at the end of this section. Please note that some signals are connected inverted. Right-click on the I/O to open the shortcut menu and select Invert.



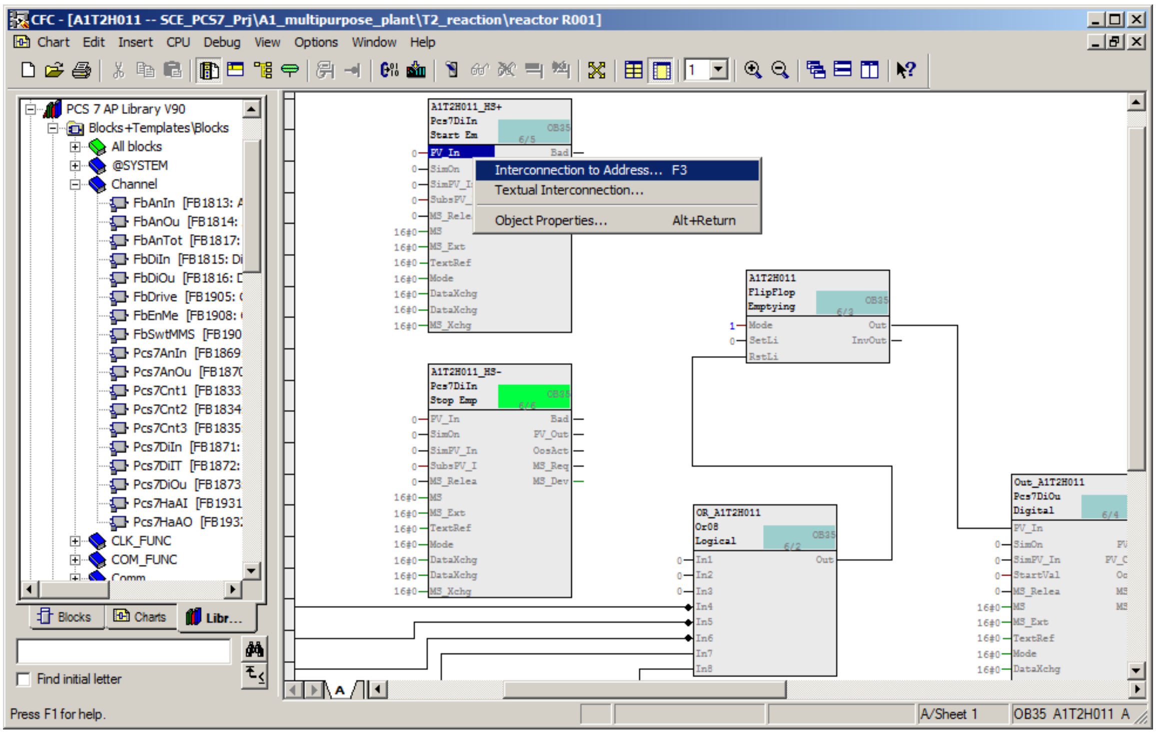
17. To complete the chart, the two signals that start and stop the manual operation A1T2H011 are needed. To read in these signals, insert two driver blocks for a digital input signal. (→ Libraries → PCS 7 AP Library V90 → Blocks+Templates\Blocks → Channel → PCS7DiIn → Pcs7DiIn)



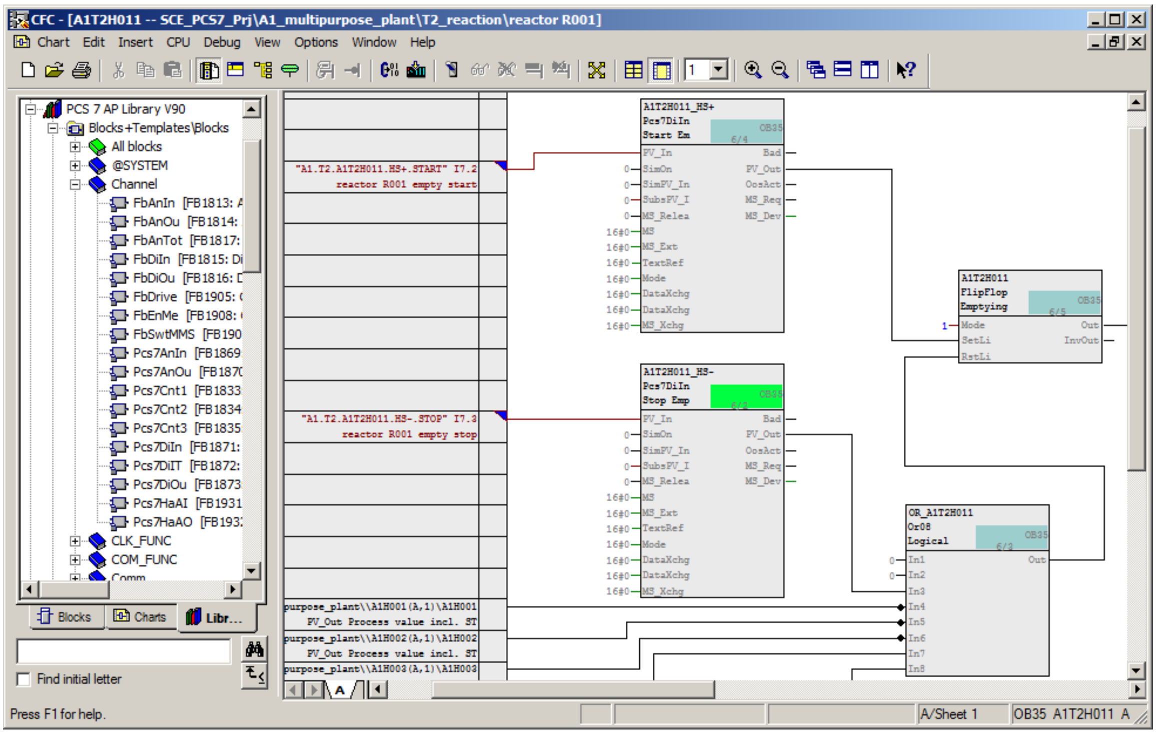
18. To differentiate the blocks, change the name and add a comment. In the result, the blocks can now be easily distinguished.
(→ Pcs7DiIn → Object properties → Name: A1T2H011_HS+ → Comment: empty start → Name: A1T2H011_HS+ → Comment: empty stop)



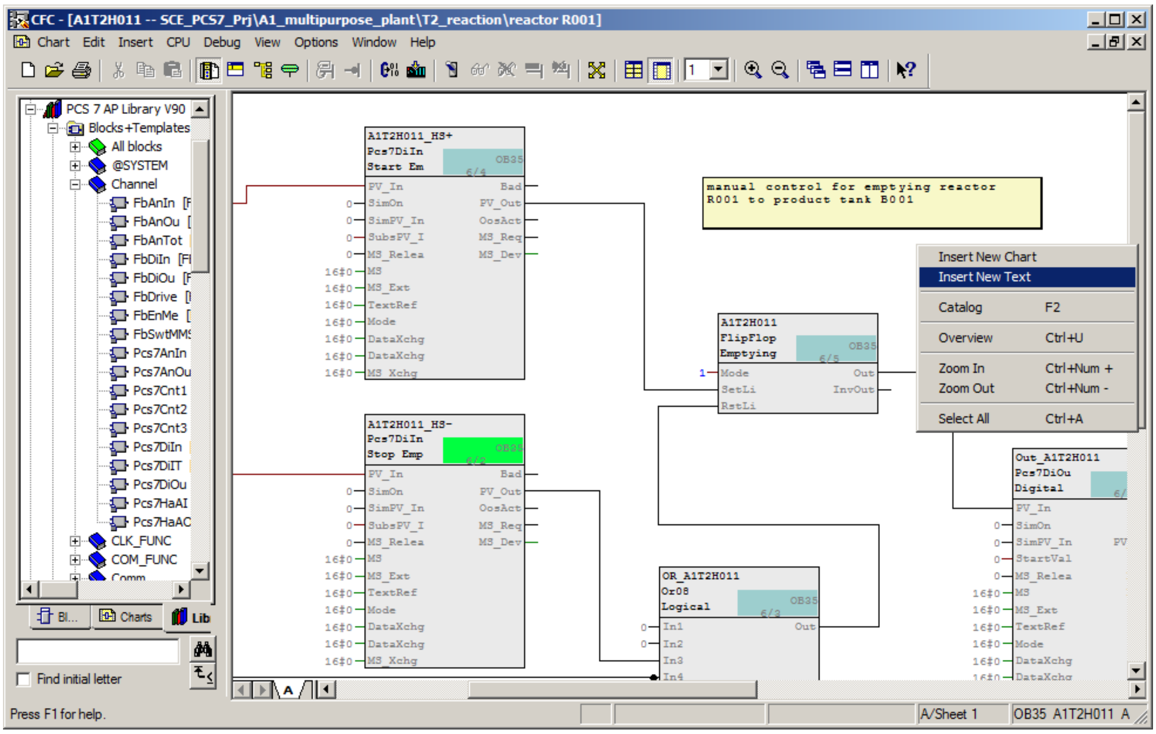
19. Then, interconnect both driver blocks with the respective signal. (→ Pcs7DiIn → PV_In → Interconnection to Address → A1.T2.A1T2H011.HS+.START / E7.2 and A1.T2.A1T2H011.HS-.STOP / E7.3)



20. Finally, the 'PV_Out' output of the block for starting must be connected to the 'SetLi' input of the flipflop, and the 'PV_Out' output of the block for stopping to an input of the 'Or08' block. (→ Pcs7DiIn (A1T2H011_HS+) → PV_Out → FlipFlop → SetLi →Pcs7DiIn (A1T2H011_HS-) → PV_Out → Or08 → In3)



21. The final step is to insert a text field for the description. The inserted text field can be edited with a double-click. (→ Right click → Insert New Text → "Manual control for emptying reactor R001 to product tank B001")



22. Using the tables below, check the interconnections you have just created for A1T2H011.

Note:

– *The cross-sheet interconnections are all structured the same. Their structure will be explained in the following example.*

'A1T2L001(A,1) / Level_A1T2L001 PV_Out Process value incl. ST' stands for:

– *Chart A1T2L001*

– *Subchart A, Sheet 1*

– *Block Level_A1T2L001*

– *I/O PV_out Process value including ST (STRUCT consisting of value and ST)*

Input:	**Interconnection to:**	**Inverted**
Pcs7DiIn.HS+.PV_In	'A1.T2.A1T2H011.HS+.START' / I7.2 / reactor R001 empty start	No
Pcs7DiIn.HS-.PV_In	'A1.T2.A1T2H011.HS-.STOP' / I7.3 / reactor R001 empty stop	No
Or08.In4	A1H001(A,1) / A1H001 PV_Out Process value incl. ST	Yes
Or08.In5	A1H002(A,1) / A1H002 PV_Out Process value incl. ST	Yes
Or08.In6	A1H003(A,1) / A1H003 PV_Out Process value incl. ST	Yes
Or08.In7	A1T3L001(A,1) / A1T3L001_LSA+ PV_Out Process value incl. ST	No
CompAn02.In1	A1T2L001(A,1) / Level_A1T2L001 PV_Out Process value incl. ST	No
CompAn02.In2	50.0	
FlipFlop.Mode	1	

Table 1: Input interconnections in chart 'A1T2H011/Sheet1'

Input:	Output:	Inverted
FlipFlop.SetLi	Pcs7DiIn.HS+.PV_Out	No
FlipFlop.RstLi	Or08.Out	No
Or08.In3	Pcs7DiIn.HS-.PV_Out	No
Or08.In8	CompAn02.LT	No
Pcs7DiOu.PV_In	FlipFlop.Out	No

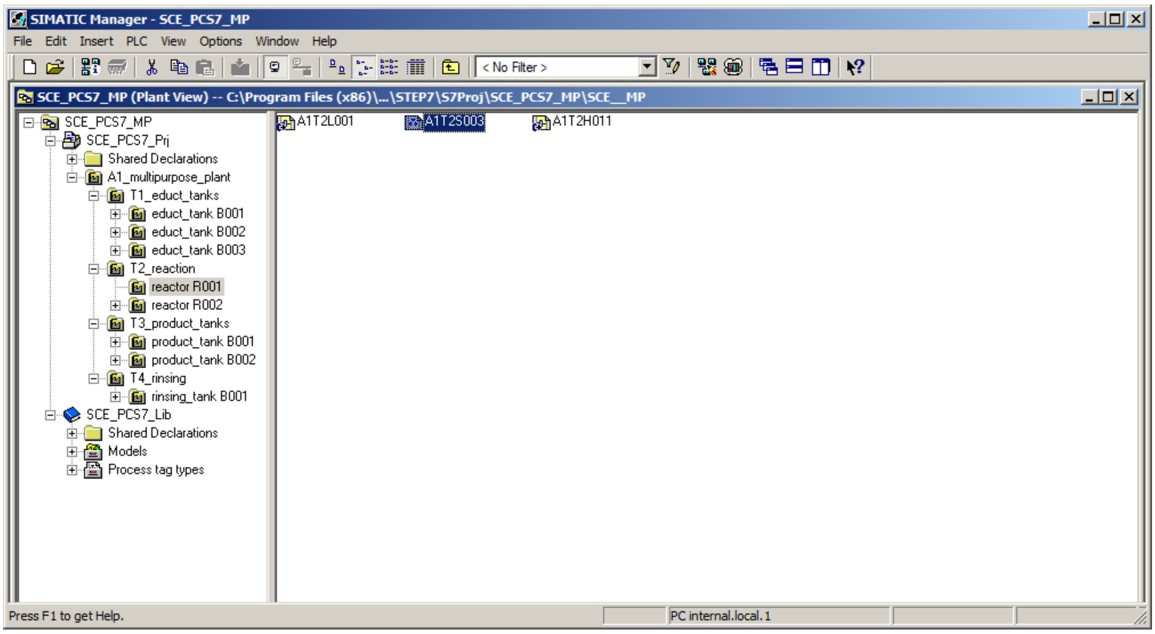
Table 2: Block interconnections in chart 'A1T2H011/Sheet1'

Output:	Interconnection to:	Inverted
Pcs7DiOu.PV_Out	'A1.T2.A1T2H011.HO+-.0+' / Q4.2 / reactor R001 empty status value	No

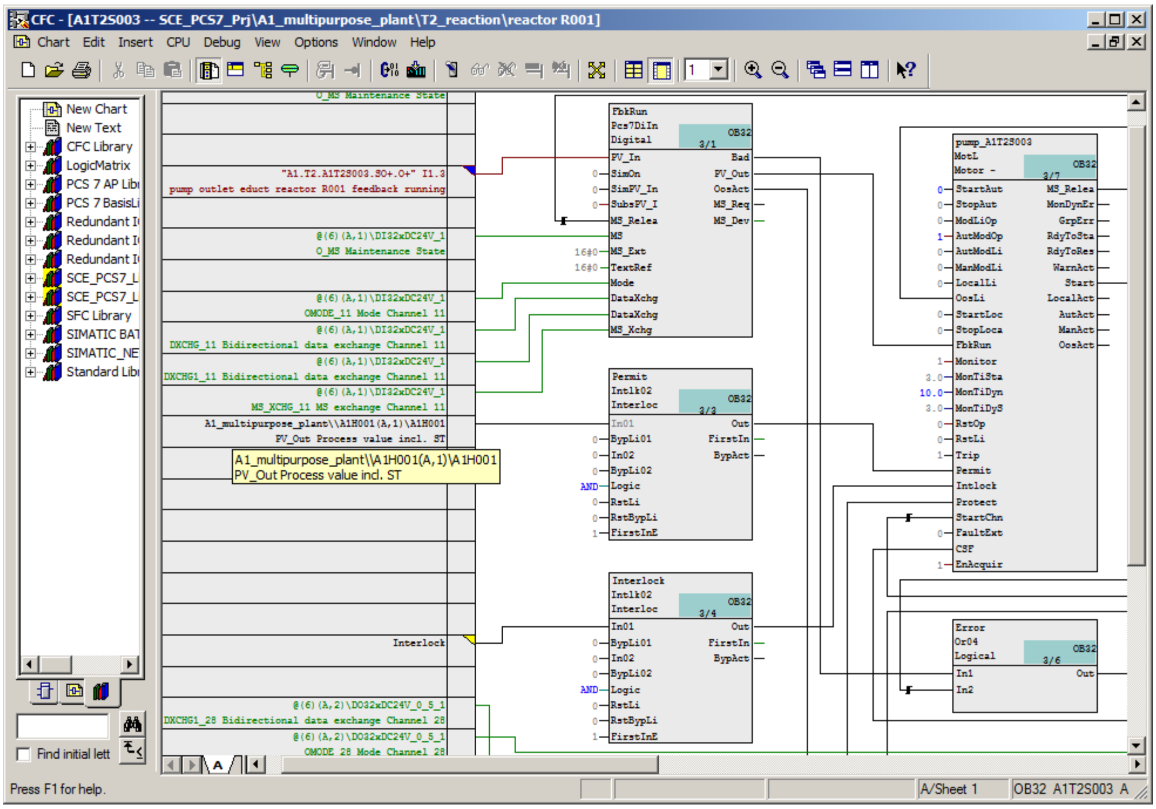
Table 3: Output interconnections in chart 'A1T2H011/Sheet1'

8.2 Interlocks of pump A1T2S003

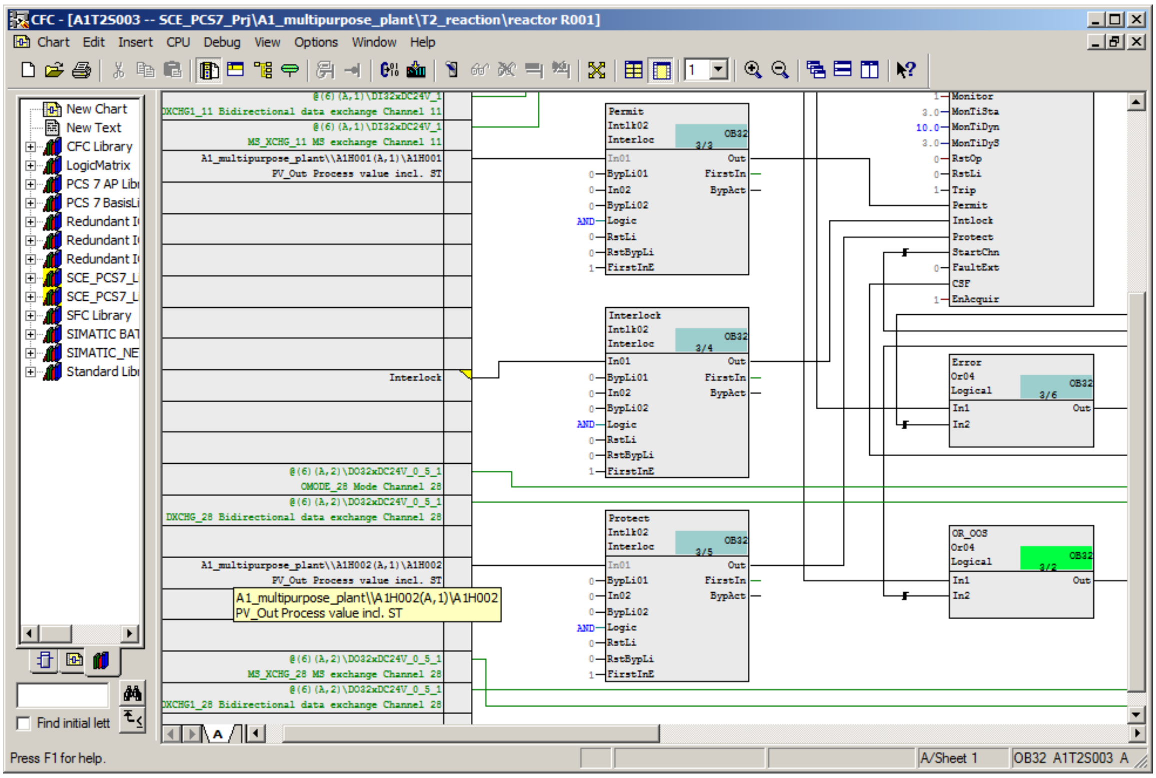
1. The first step is to create the interlock conditions for the pump 'Drainage Reactor R001'. To do this, open the CFC 'A1T2S003' in the plant view with a double-click. (→ SIMATIC Manager → Plant View → A1T2S003)



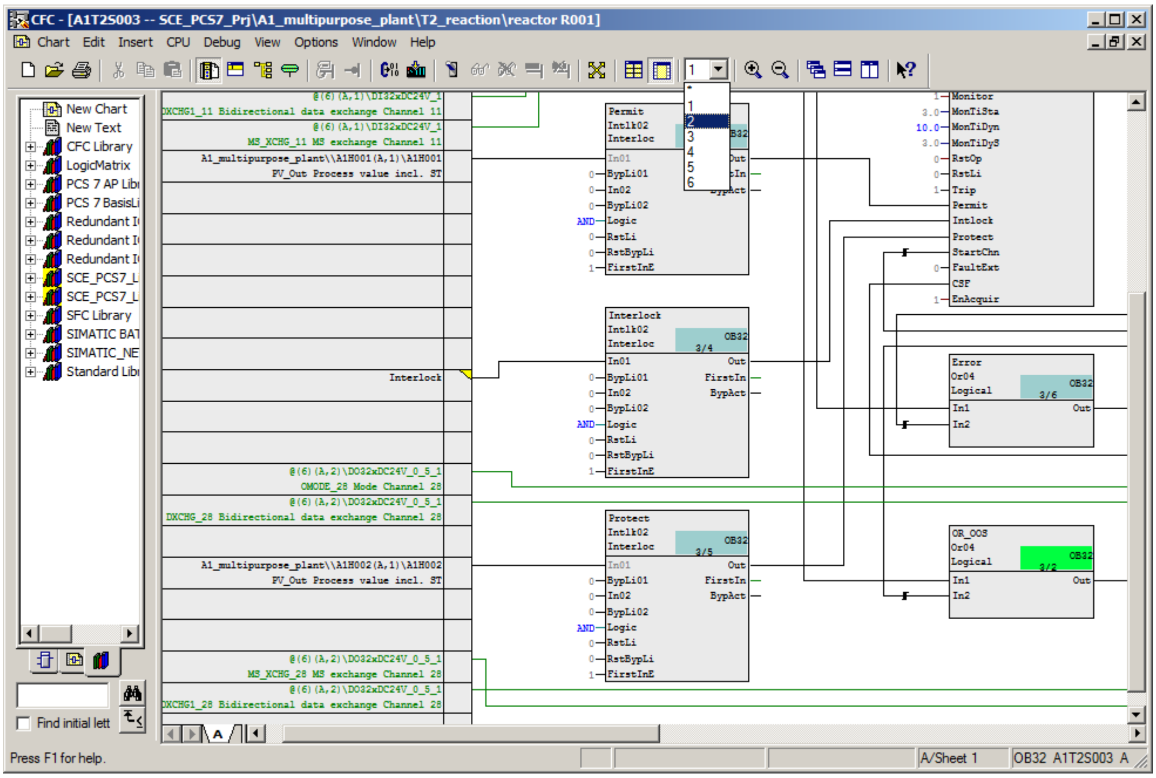
2. The Motor_Lean template contains other blocks besides the control module function for the 'MotL' motor. There are three 'Intlk02' blocks for the interlock of 'MotL'. The first is called 'Permit' and permits control of the motor only when the conditions are met. Here, you connect the signal of the main switch of plant A1H001. For this, you first delete the connection to the 'Permit' placeholder on the left margin. Next, you create a cross-chart connection. The result is shown here and in Table 4 at the end of the section.



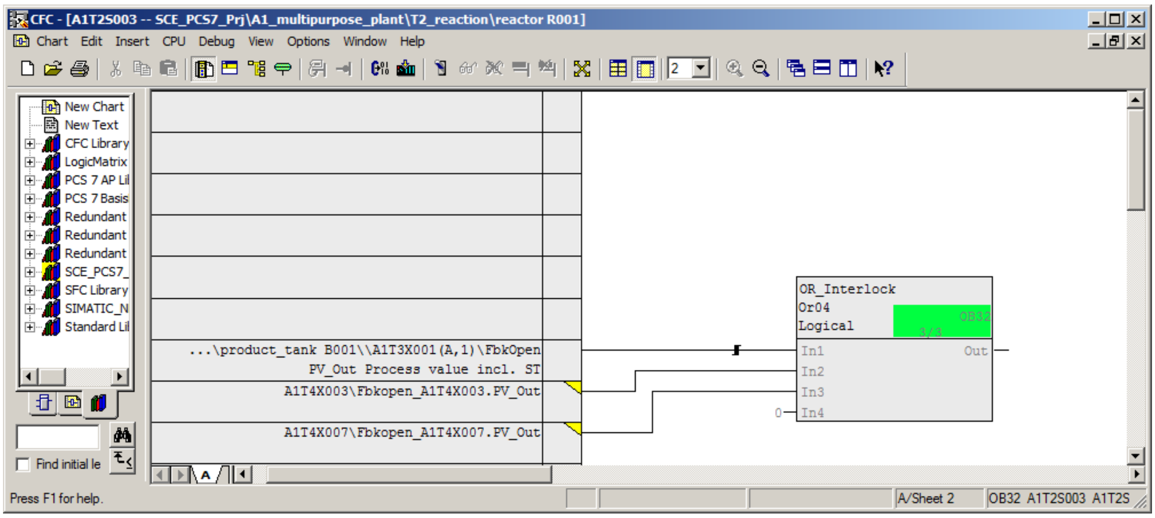
3. Now, do the same for the 'Protect' block. The 'Protect' block is used for connecting interlocks that require an acknowledgment for the motor to be enabled again. Here you connect EMERGENCY OFF. The connection is shown in Table 4 at the end of the section.



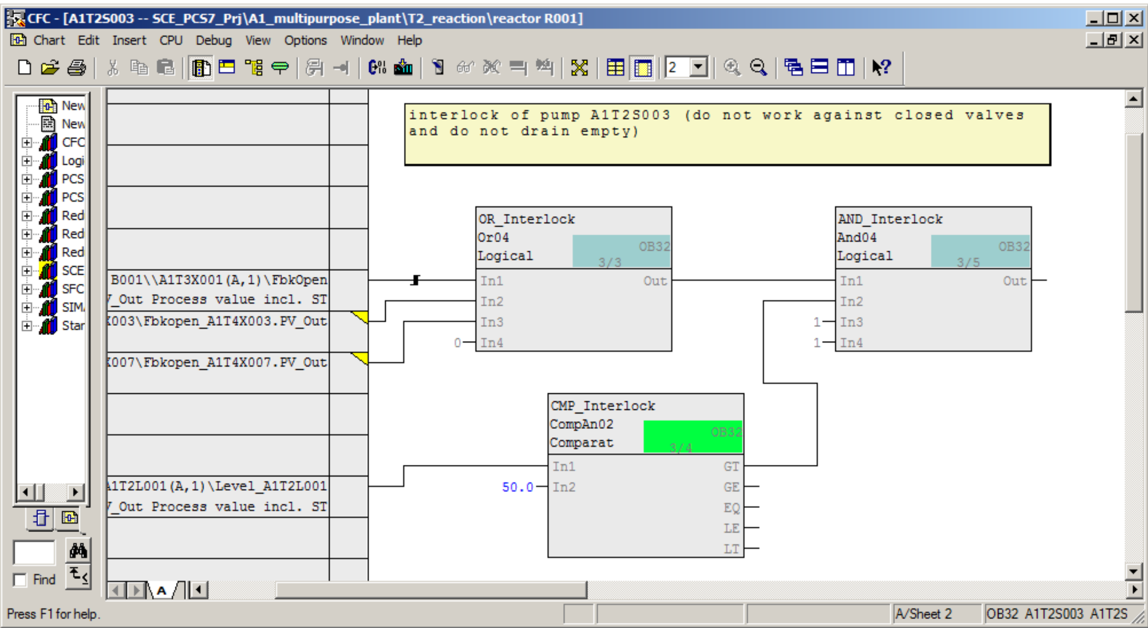
4. The 'Interlock' block is intended for general interlock conditions. Here, you will implement the conditions from the task description (for example, at least one valve open). Because more than two conditions exist in this example, you must group them first before they can be logically combined. For this, you first change to sheet 2 of the CFC.



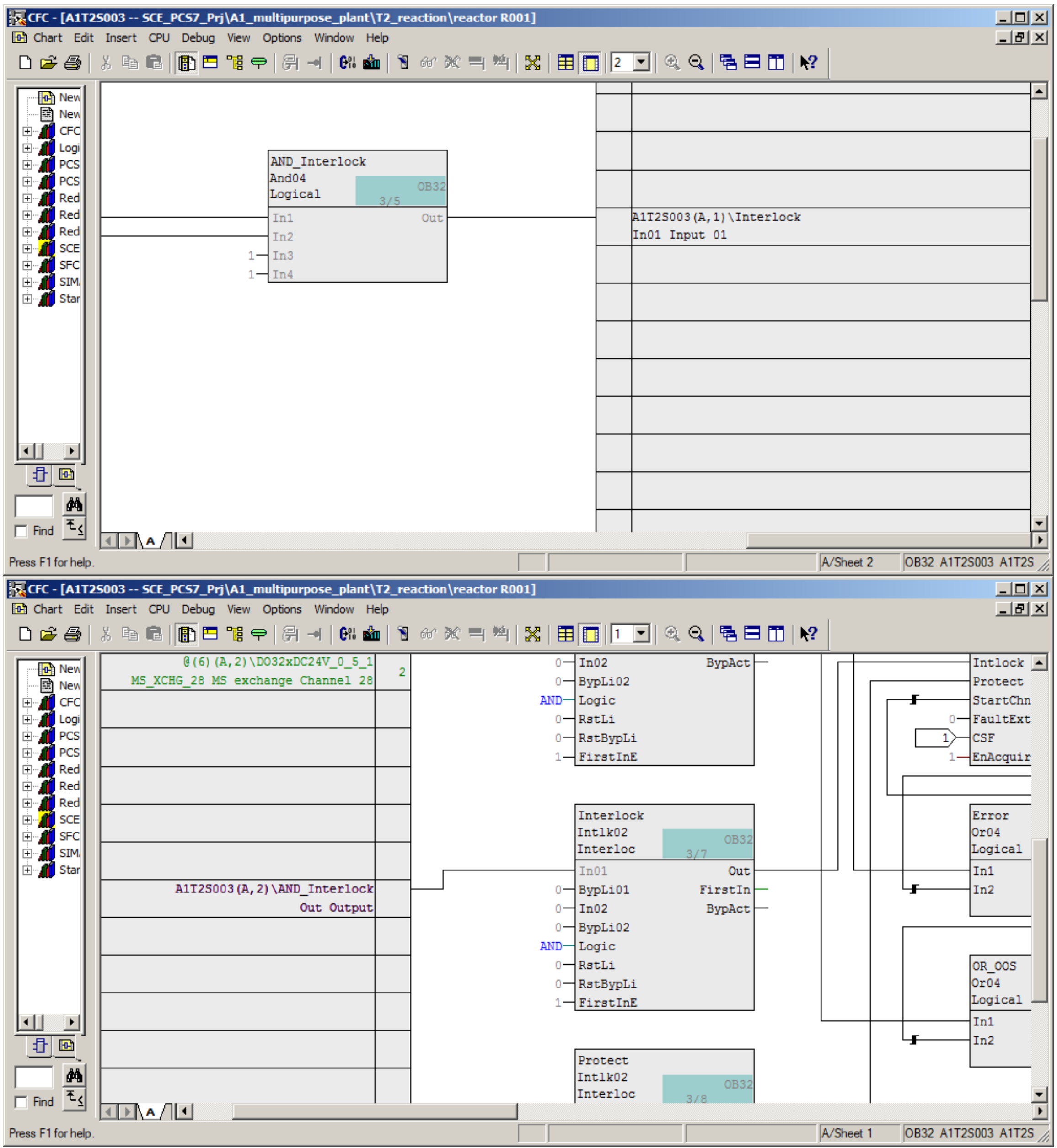
5. On the new sheet, insert an 'Or04' block from the library. The feedback signals (FbkOpen) of the valves must now be connected to this block. Because only 1 of the 3 valves has been created so far, you can create placeholders for these signals using textual interconnections. These produce warnings when the program is compiled, but the program still functions. You can find an overview of the connections created here in Table 5 and Table 6 at the end of the section.



6. Next, the minimum level must be queried using the 'CompAn02' block, and the conditions can be combined with an 'And04'. The connections are shown in Table 5, Table 6 and Table 7 at the end of this section. The result looks like this.



7. The next step is to connect the 'Out' output of the 'And04' block on sheet 2 to the 'In01' input of the 'Interlock' block on sheet 1. For this, you must delete the textual interconnection on the Interlock block beforehand. The new interconnection is shown in Table 8.



8. Below you are provided a further overview of all the new interconnections in the 'A1T2S003' chart.

Input:	Interconnection to:	Inverted
Permit.In01	A1H001(A,1) / A1H001 PV_Out Process value incl. ST	No
Protect.In01	A1H002(A,1) / A1H002 PV_Out Process value incl. ST	No

Table 4: Input interconnections in chart 'A1T2S003/Sheet1'

Block:	Catalog/Folder:
Or04 / OR function with 4 inputs	Libraries/PCS7 APL V90/ Blocks+ Templates \Blocks/LogicDi
And04 / AND function with 4 inputs	Libraries/PCS7 APL V90/ Blocks+ Templates \Blocks/LogicDi
CompAn02 / Comparison of analog values	Blocks / LogicAn

Table 5: New blocks in chart 'A1T2S003/Sheet2'

Input:	Interconnection to:	Inverted
Or04.In1	A1T3X001(A,1) / FbkOpen_A1T3X001 PV_Out Process value including ST	No
Or04.In2	A1T4X003\FbkOpen.PV_Out (textual interconnection)	No
Or04.In3	A1T2X007\FbkOpen.PV_Out (textual interconnection)	No
CompAn02.In1	A1T2L001(A,1) / Level_A1T2L001 PV_Out Process value including ST	
CompAn02.In2	Value: Value=50.0 Comment=Minimum level	

Table 6: Input interconnections in chart 'A1T2S003/Sheet2'

Input:	Output:	Inverted
And04.In1	Or04.Out	No
And04.In2	CompAn02.GT	No

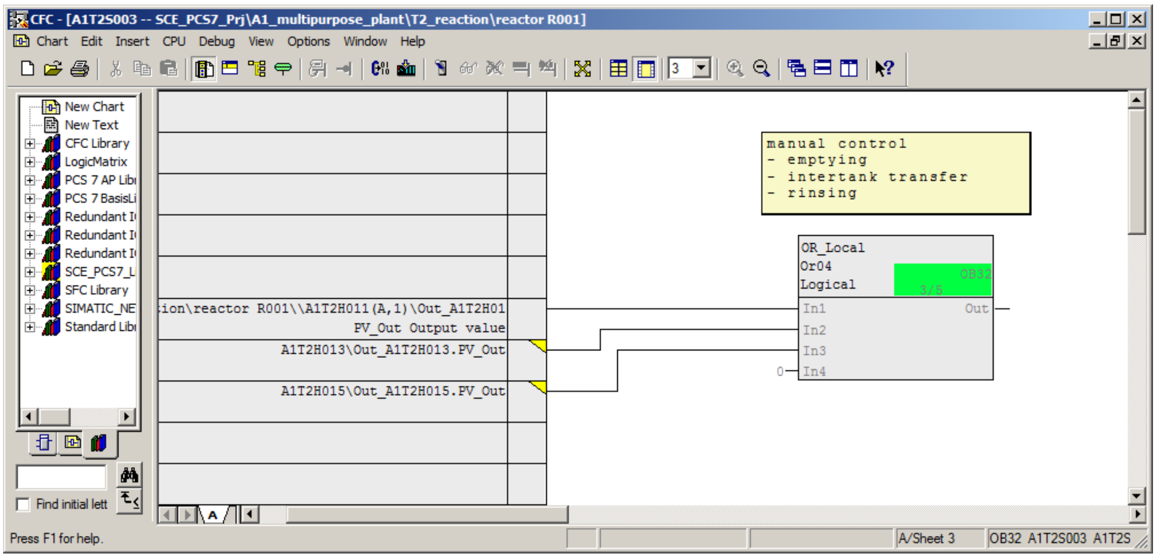
Table 7: Block interconnections in chart 'A1T2S003/Sheet2'

Input:	Output:	Inverted
Interlock.In02	And04.Out	No

Table 8: Block interconnections between chart 'A1T2S003/Sheet1' and 'A1T2S003/Sheet2'

8.3 Manual operation of pump A1T2S003 and valve A1T3X001

1. In sheet 1 of chart 'A1T2S003', you now make the interconnections for manual operation with A1T2H011 (for emptying reactor R001). Because other manual operations can also access the pump, an 'Or04' is created on sheet 3.



Block:	Catalog/Folder:
Or04 / OR function with 4 inputs	Blocks/LogicDi

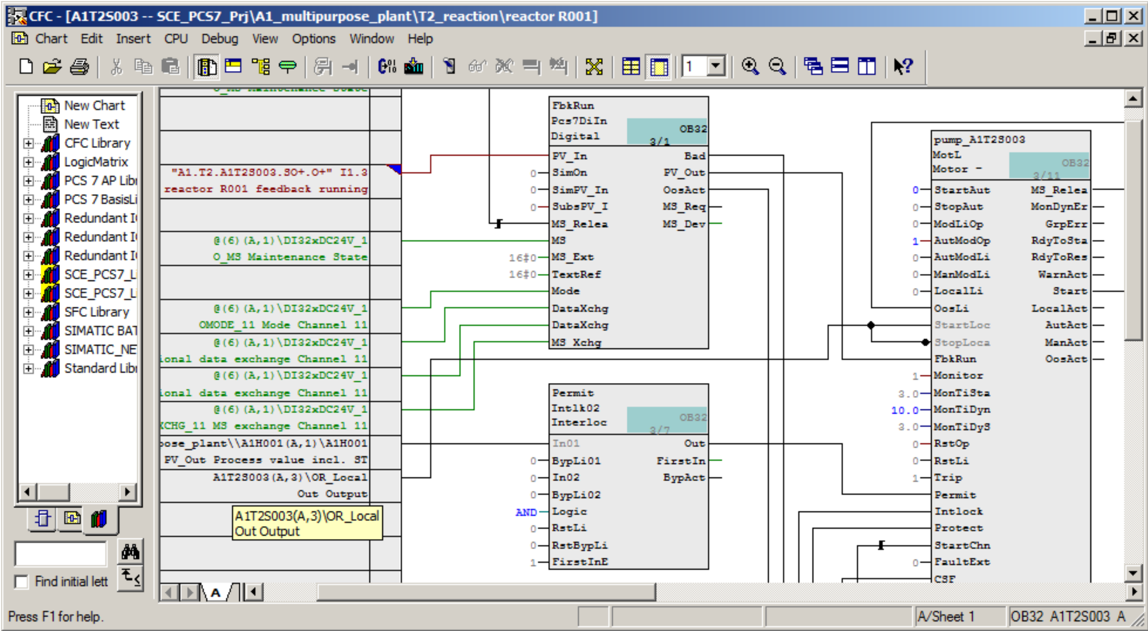
Table 9: New blocks in chart 'A1T2S003/Sheet3'

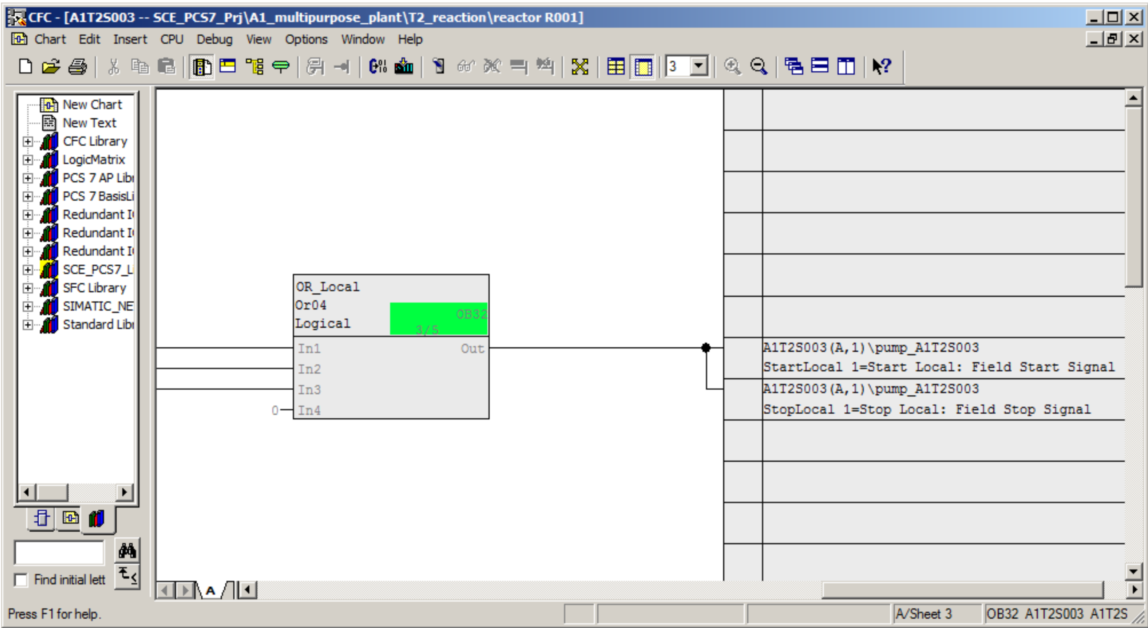
Input:	Interconnection to:	Inverted
Or_Local.In1	A1T2H011(A,1) / Out_A1T2H011 PV_Out Output value	No
Or_Local.In2	A1T2H013\Out_A1T2H013.PV_Out (textual interconnection)	No
Or_Local.In3	A1T2H015\Out_A1T2H015.PV_Out (textual interconnection)	No

Table 10: Input interconnections in chart 'A1T2S003/Sheet3'

Input:	Output:	Inverted
MotL.Pump_A1T2S003.StartLocal	Or04.Or_Local.Out	No
MotL.Pump_A1T2S003.StopLocal	Or04.Or_Local.Out	Yes

Table 11: Block interconnections between the 'A1T2S003/Sheet1' and 'A1T2S003/Sheet3' charts





2. Switchover to local operation is required for the local start and stop of the manual operation to work.

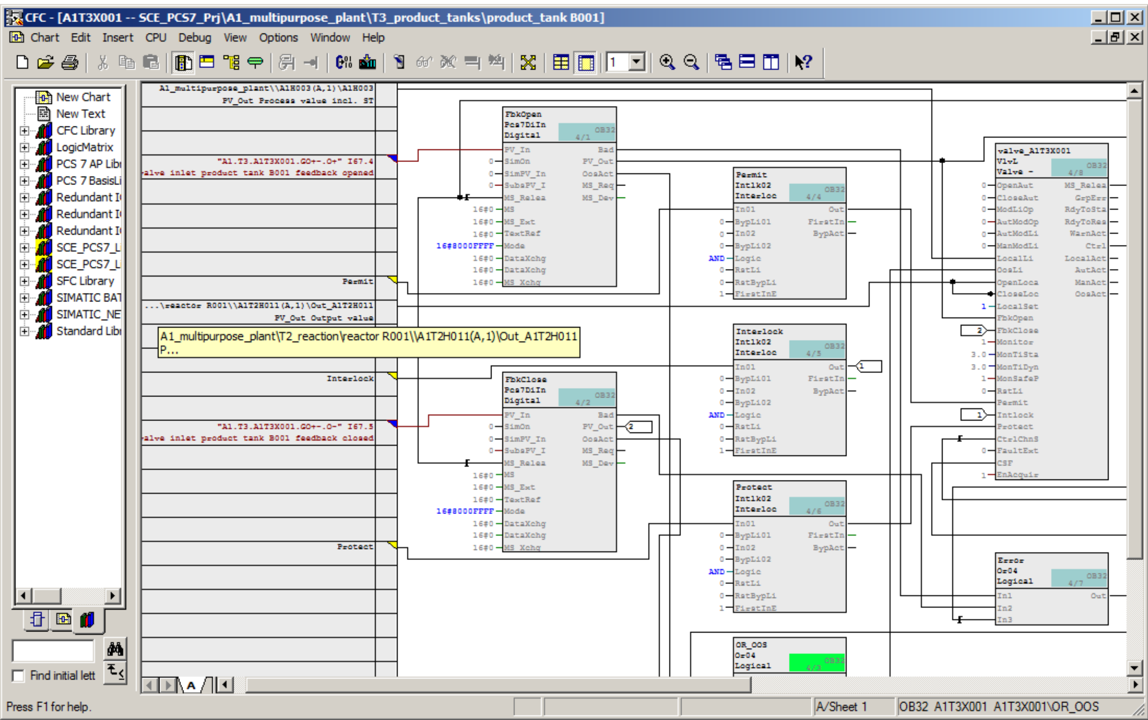
Input:	Interconnection to:	Inverted
MotL.Pumpe_A1T2S003 .LocalLi	A1H003(A,1) / A1H003 PV_Out Process value incl. ST	No
MotL.Pumpe_A1T2S003 .LocalSetting	1 (→ Connection invisible → Double-click on block and open I/Os → Change value)	

Table 12: Additional input interconnections in chart 'A1T2S003/Sheet1'

3. For the manual operation A1T2H011, valve A1T3X001 is needed in addition to pump A1T2S003; the valve was implemented in the exercise in the previous chapter. The interconnections additionally created for the manual operation in chart A1T3X001 are shown below.

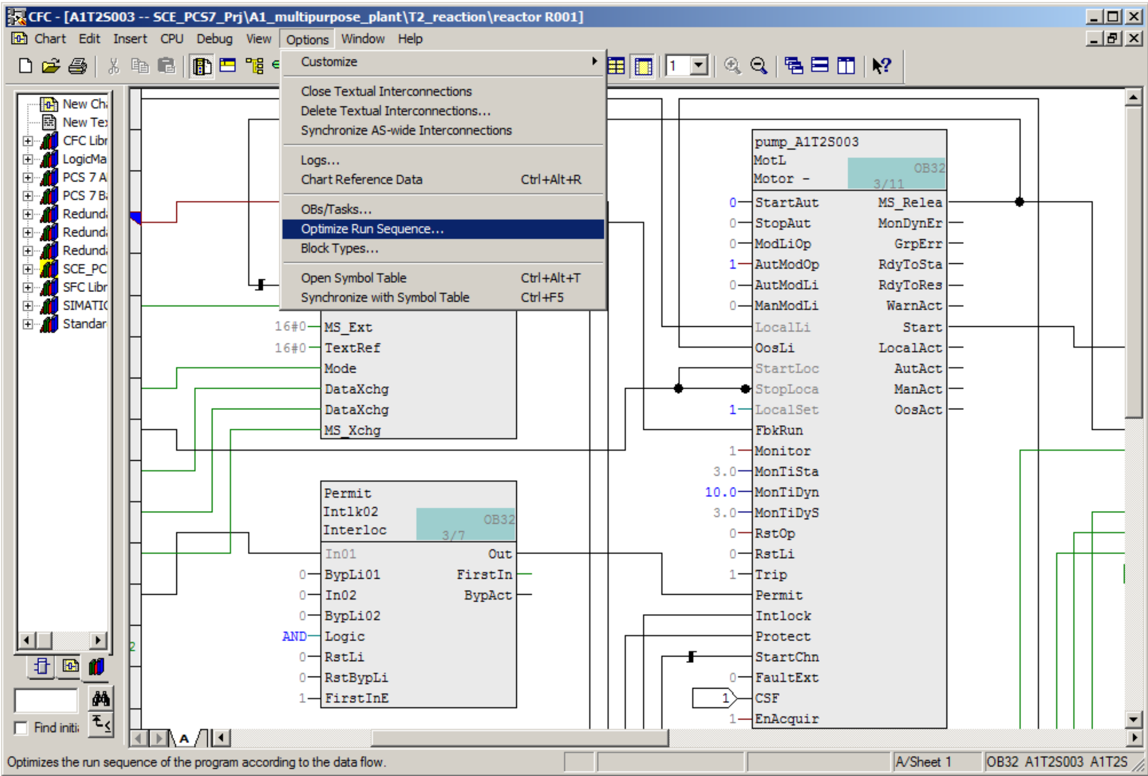
Input:	Interconnection to:	Inverted
VlvL.Ventil_A1T3X001 .LocalLi	A1H003(A,1) / A1H003 PV_Out Process value including ST	No
VlvL.Ventil_A1T3X001 .OpenLocal	A1T2H011(A,1) / Out_A1T2H011 PV_Out Process value including ST	No
VlvL.Ventil_A1T3X001 .CloseLocal	A1T2H011(A,1) / Out_A1T2H011 PV_Out Process value including ST	Yes
VlvL.Ventil_A1T3X001 .LocalSetting	1	

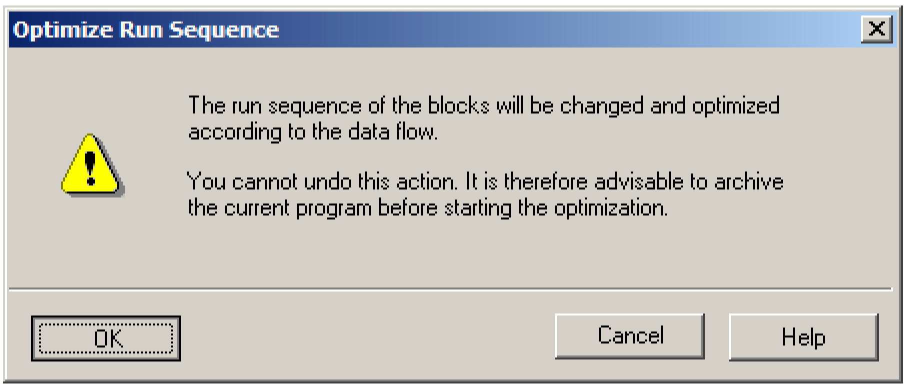
Table 13: Input interconnections in chart 'A1T3X001/Sheet1'



8.4 Optimizing the run sequence

All blocks used in the charts are integrated in the run sequence of the overall program when inserted. Clicking on the symbol  displays the run sequence. To improve the data flow within the overall program, it is recommended that the run sequence be optimized after creating the program. To do so, select "Optimize Run Sequence' in the "Options" menu of the CFC editor (irrespective of which view you are in). (→ Options → Optimize Run Sequence → OK)





8.5 Checklist – step-by-step instruction

The following checklist helps students to independently check whether all steps of the step-by-step instruction have been carefully completed and enables them to successfully complete the module on their own.

No.	Description	Checked
1	Manual control A1T2H011 created and completely interconnected	
2	Interlocks combined in motor A1T2S003	
3	Manual control combined in motor A1T2S003	
4	Manual control combined in valve A1T3X001	
5	Run sequence optimized	
6	Interlocks and manual control successfully tested (optional)	
7	Project successfully archived	

Table 14: Checklist for step-by-step instructions

9 Exercises

In the exercises, you apply what you learned in the theory section and in the step-by-step instructions. The existing multiproject from the step-by-step instructions (p01-05-project-r1905-en.zip) is to be used and expanded for this. The download of the project is stored as zip file "Projects" on the SCE Internet for the respective module.

In the following exercises, you can design and implement additional interlocks on your own. Keep in mind that for the valve interlock, only the EMERGENCY OFF switch, the main switch and the level(s) of the respective tank(s) are needed.

9.1 Tasks

The following tasks are geared to the step-by-step instructions. The corresponding steps of the instructions can be used to assist with each task.

1. Complete the interlocks for the existing valve:
   - A1T3X001
2. Create the CFC for the level of educt tank B001:
   - A1T1L001
3. Create the CFCs of the following valves, including interlocks:
   - A1T1X004
   - A1T2X001
4. Create the CFC for the following pump including interlocks:
   - A1T1S001
5. Create the CFC for the following manual operation:
   - A1T2H001
6. Test the implementation with the simulation! You should now be able to pump from educt tank B001 to reactor R001 and then to product tank B001 with the manual controls.

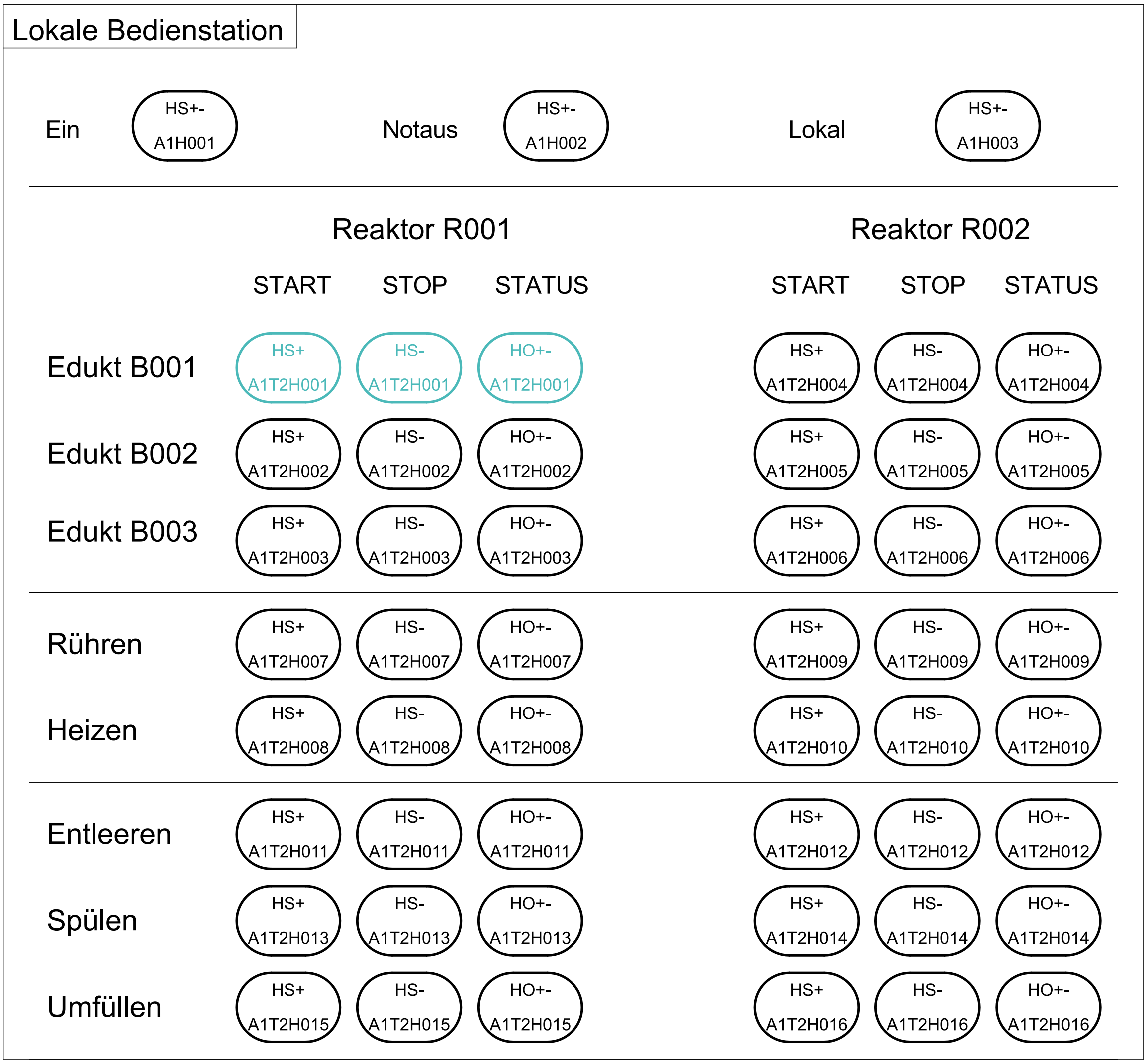


Figure 11: Excerpt from the local operator station

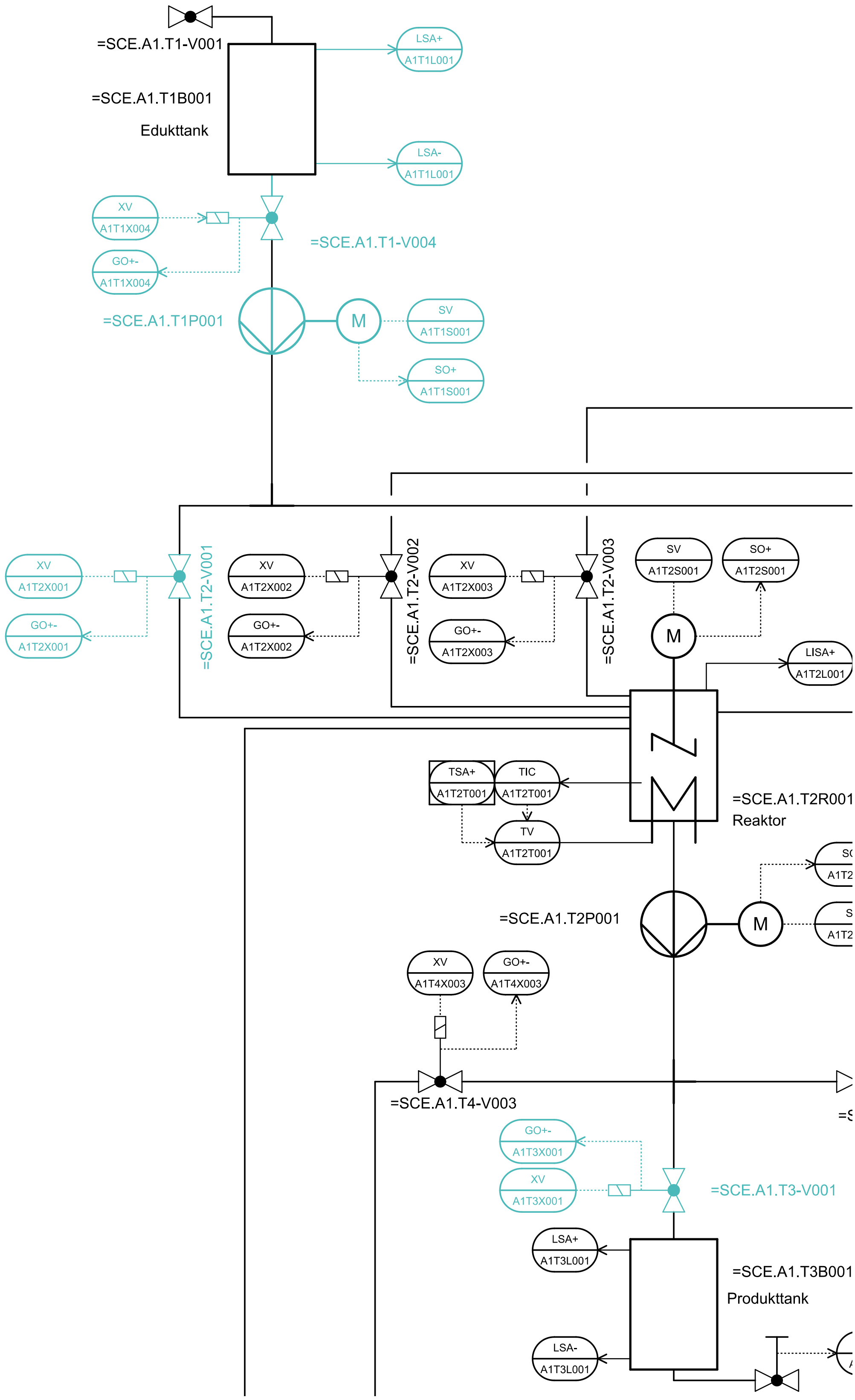


Figure 12: Excerpt from P&ID flowchart

9.2 Checklist – exercise

The following checklist helps students to independently check whether all steps of the exercise have been carefully completed and enables them to successfully complete the module on their own.

No.	Description	Checked
1	Interlocks combined in motor A1T3X001	
2	Level 'Educt tank B001\A1T1L001' created and configured	
3	Valve 'Educt tank B001\A1T1X004' created, configured and interlocked	
4	Valve 'Reactor R001\A1T2X001' created, configured and interlocked	
5	Pump 'Educt tank B001\A1T1S001' created, configured and interlocked	
6	Manual control 'Reactor R001\A1T2H001' created and configured	
7	Manual control A1T2H001 combined with pump A1T1S001 and valve A1T2X001	
8	New elements successfully tested (optional)	
9	Project successfully archived	

Table 15: Checklist for exercises

10 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:

siemens.com/sce/pcs7

Preview "Additional information"

Getting Started, Videos, Tutorials, Apps, Manuals, Trial-SW/Firmware

- SIMATIC PCS 7 Overview
- SIMATIC PCS 7 Videos
- Getting Started
- Application Examples
- Download Software/Firmware
- SIMATIC PCS 7 Website
- SIMATIC S7-400 Website

Further Information

Siemens Automation Cooperates with Education
siemens.com/sce

Siemens SIMATIC PCS 7
siemens.com/pcs7

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
Digital Industries, FA
P.O. Box 4848
90026 Nuremberg
Germany