SCE Training Curriculum

Siemens Automation Cooperates with Education (SCE) | 09/2015

PA Module P01-05

SIMATIC PCS 7 – Functional Safety

Matching SCE Trainer Packages for these curriculum

- **SIMATIC PCS 7 Software block of 3 packages**
  Order No. 6ES7650-0XX18-0YS5
- **SIMATIC PCS 7 Software block of 6 packages**
  Order No. 6ES7650-0XX18-2YS5
- **SIMATIC PCS 7 Software Upgrade block of 3 packages**
  Order No. 6ES7650-0XX18-0YE5 (V8.0 → V8.1) or 6ES7650-0XX08-0YE5 (V7.1 → V8.0)
- **SIMATIC PCS 7 Hardware Set including RTX Box**
  Order No. 6ES7654-0UE13-0XS0

Please note that these trainer packages may be replaced with subsequent packages.
An overview of the available SCE packages is provided at: siemens.com/sce/tp

Continuing education

For regional Siemens SCE continuing education, contact your regional SCE contact partner.
siemens.com/sce/contact

Additional information relating to SIMATIC PCS 7 and SIMIT

In particular, Getting Started, videos, tutorials, manuals and programming guide.
siemens.com/sce/pcs7

Additional information relating to SCE

siemens.com/sce

Note on Usage

We would like to thank the Technical University Dresden, particularly Prof. Dr. Leon Urbas and Annett Krause, MS, as well as the Michael Dziallas Engineering Corporation and those who provided support in preparing this SCE training document.

FUNCTIONAL SAFETY

TRAINING OBJECTIVE

After working through this module, the students are familiar with the basic requirements for functional safety. They will learn methods to identify potential danger as well as to evaluate the risks resulting from it. They will know methods and design concepts to safeguard plants with the means of process control engineering. They will learn the basic connections for interlocking controls.

THEORY IN BRIEF

In modern production plants, process variables are used to control and safeguard technical processes. Based on the given technical general conditions, permissible and impermissible ranges are defined for these variables. The state of the entire 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, different interlocking mechanisms are set up. The objective of interlocks is to prevent combinations, sequences, and time characteristics of signals that can lead to impermissible fault states.

This can be done, among other things, with the means of process engineering through so-called safety set-ups. They prevent faults from occurring, or they limit the damage if an impermissible fault state occurred despite all measures taken. In order to design suitable interlock mechanisms, a protective concept has to be developed for the plant. This task requires exact knowledge of the chemical, process and system boundary conditions. For this reason, the protective concept is developed by an interdisciplinary team using the HAZOP or PAAG analysis.

The technical implementation of the mechanisms in a process control system should be designed as simple as possible, have a direct effect and be manageable. For this reason, in practice recurring standard protective circuits are resorted to. They can be arranged in four categories.

Combinatorial circuits are used to generate switching conditions by directly combining the corresponding process signals. To this end, the input signals are connected through the logic operations AND, OR and NOT. The state of the output signal of such a combinatorial circuit can thus be determined anytime 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 with combinatorial circuits.
Locking circuits prevent that signals having opposite effects can be set simultaneously. If a certain sequence for several control signals is required beyond this, it is referred to as a sequence lock. Interlocks are implemented using R-S flip-flops that are connected to each other.
Circuits with timing behavior allow for delayed switch-on/switch-off, the definition of a minimum as well as maximum execution time, and the implementation of protective functions that require a certain reaction time. To implement such circuits, different pre-assembled time blocks are available.

THEORY

PROCESS VARIABLES

Production plants are used to produce material goods. They control and monitor the flow of material and energy that can be described with physical variables such as volume, mass, temperature or flow. Based on process and system engineering boundary conditions, those physical variables that are relevant to the technical process and that can be measured are defined and specified. These variables are called process variables.

Process variables are used to control or safeguard technical processes. For each process variable, ranges are specified based on chemical, process or system engineering boundary conditions for which this process variable is intended (OK range), as well as ranges outside the OK range where no safety related restrictions exist for further operation (permissible fault range). If the process variable is outside these ranges, undesirable events that result directly in bodily injury and environmental damage are to be reckoned with (impermissible fault range).

The values of the process variables are recorded and evaluated with the means of process control engineering. From this data, the current state of the system is determined. Three basic states are differentiated:

OK state: The values of all process variables are in the respective OK range and no danger is emanating from the system elsewhere.
Permissible fault: The values of one or several process variables are in the respective permissible fault range. Otherwise, no danger is emanating from the system
Impermissible fault state: The values of one or several process variables are in the respective impermissible fault range and danger emanates from the system elsewhere.

Impermissible fault states always exist when bodily harm can occur or the life of employees is jeopardized, the environment is damaged, technical facilities are destroyed or the product yield is diminished. It suffices in this case that the probability is sufficiently high for these events to occur [1].

FUNCTIONAL SAFETY

Functional safety refers in general to safeguarding the process engineering system against fault states [1]. For many processes states in process systems, certain events can occur that cause damage. The combination of the frequency of the occurrence of damage and its extent is called the risk of the corresponding process or state. It is the objective of functional safety to take protective measures that decrease the existing risks to the extent that the remaining risk is below the acceptable risk that is to be defined, i.e., justifiable [2].

The locking mechanisms described in the chapter ‘Individual Drive 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 a pump because of an undetected dry run). These fault states are device-specific and can be detected regardless of process and system engineering boundary conditions.

The locking mechanisms described by their very nature cannot safeguard against process related fault states (for example, dry run of a pump) by themselves because they depend on process and system engineering events (for example, dropping below a minimum tank level causes the pump to run dry). For this reason, plants have to be made safe by implementing suitable process-related interlocks. These often utilize and expand the locking mechanisms of the individual drive functions (refer to the chapter ‘Individual Drive Functions’). All types of intended plant operations have to be taken into account.

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 mode
– Startup and shutdown operation
– Commissioning and decommissioning
– Test mode
– Inspection, maintenance and repair processes

To this end, an interdisciplinary team first develops a safety concept for the plant. The team systematically identifies danger potential and faults that can lead to danger. Established methods for danger analysis are used, for example, the PAAG procedure [3].

Next, the risks have to be evaluated that result from the dangers that were recognized. Different methods for graduated evaluation of the risk to be covered are available, for example, the ALARP method, the LOPA method or the method of the risk graph specified in [2]. If the initial risk of danger is larger than the specified acceptable risk, protective measures have to be taken that reduce the risk accordingly.

FUNCTIONAL SAFETY BY MEANS OF PROCESS CONTROL ENGINEERING

For functional safety, protective equipment that is not based on the means of process control technology is used. Often, however, this is not possible or sufficient because either of the size or complexity of the plant, or the corresponding solution cannot be efficiently implemented under economical aspects. In this case, protective functions are implemented with the means of process control technology. For this reason, we distinguish between two types of process control systems (PCS) equipment:

PCS operating and monitoring equipment (Basic Process Control System - BPCS) implements the automation functions required for production and is therefore used for the intended operation of the plant in its OK range [2]. PCS monitoring equipment reacts if one or several process variables leave the OK range. It signals permissible fault states or automatically takes steps to take the process variables back to the OK range. From the perspective of functional safety, no requirements are made of PCS operating and monitoring equipment.
PCS safety systems (Safety Instrumented Systems - SIS) are used to reduce the risk of recognized danger potential. They do this by either preventing an event or by limiting the damage. The objective of event-preventing PCS protective equipment is to prevent the impermissible fault to occur in the plant in the first place. It thus reduces the frequency of an undesirable state to occur as well as the risk connected with this event. The objective of damage-limiting PCS equipment is to limit the extent of the resulting damage an undesired event causes, and to decrease the risk associated with it. Such protective equipment is used very rarely.

Figure 1 shows the method of functioning of PCS devices within the scope of functional safety. Curve 1 shows a process variable that–process related–cannot reach the impermissible fault range. For this reason, a PCS monitoring device is sufficient here. In Curve 2, on the other hand, exceeding the limit to the impermissible fault range is possible. But because a non-PCS protective device exists, a PCS monitoring device is also sufficient in this case. In Curve 3, there is no such safeguarding of the plant. Therefore, a (event-preventing) PCS protective device is used to prevent the process variable from reaching the impermissible fault range.

For the process control engineering systems of a plant we have to clearly define whether they implement an operational or a monitoring function, or a protective function This differentiation facilitates planning, setup and operation, but also subsequent changes to the PCS systems.

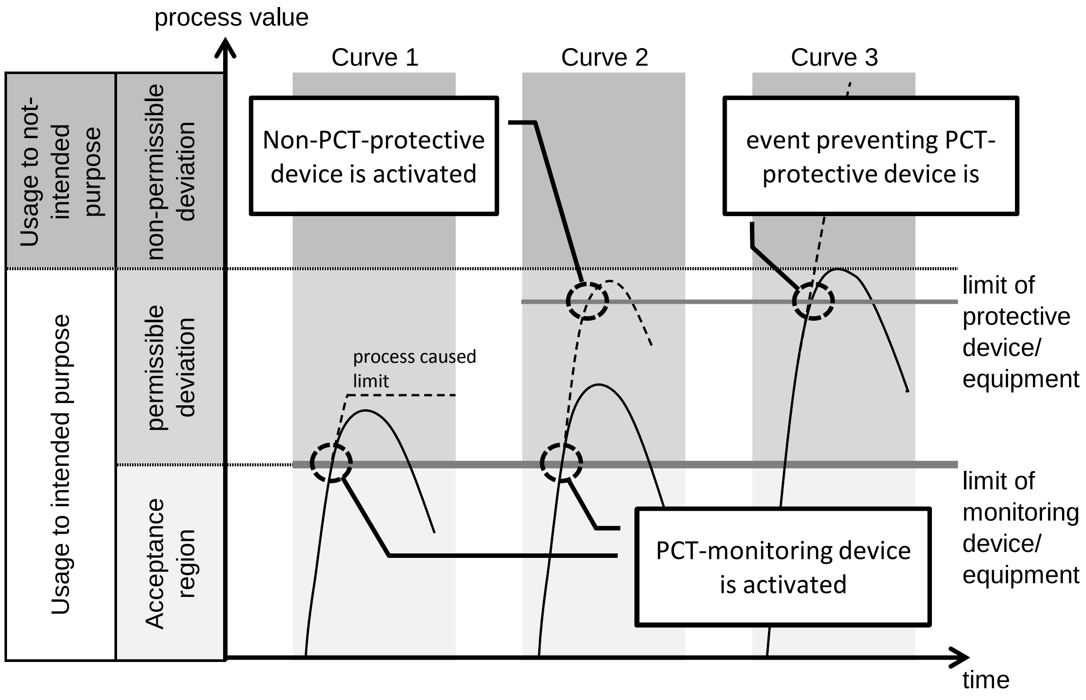

process value
Curve 1
Curve 2
Curve 3
Usage to not-intended purpose
non-permissible deviation
Usage to intended purpose
permissible deviation
Acceptance region
Non-PCT-protective device is activated
event preventing PCT-protective device is
process caused limit
limit of protective device/ equipment
limit of monitoring device/ equipment
PCT-monitoring device is activated
time

Figure 1: Schematic display of mode of action of PCS equipment according to [2]

Because functions of PCS devices are required only very rarely, components of protective equipment are sometimes used also by operational devices for economic reasons. In this case, signals for activating protecive functions must always take precedence over signals of the operational and monitoring equipment.

To implement the protective functions, manageable and immediately effective steps should be taken. The process variables used can be logged directly with simple and tried methods. This means that the complexity of the control design itself is relatively low.

STANDARD CIRCUITS FOR FUNCTIONAL SAFETY

The objective of safety systems using the means of process control engineering is usually this: to control certain combinations, sequences, time characteristics or priorities of signals in a way that impermissible process states are prevented. These functions are implemented with recurring standard circuits. The most important standard circuits are described below.

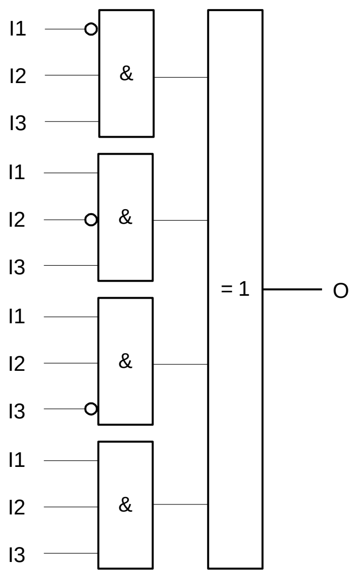
Combinatorial Circuits

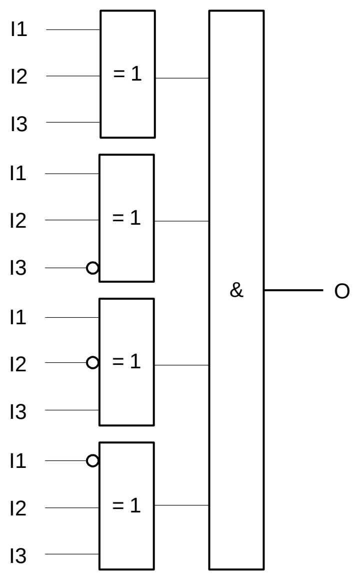
In many cases, certain control signals are permissible only if the process is in a certain state. This state can be described directly as a combination of the corresponding process signals. To link individual signals into a switching condition, simple combinatorial circuits can be used. They have the 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 combined with the logical operations AND, OR NOT. The combinatorial 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 can always be represented in (at least) two standardized forms.

Disjunctive normal form (DNF): In this representation, first all combinations of the inputs are determined for which the output signal is to be set (which means all lines of the function table for which O = 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. This sets the output as soon as one of the located combinations occurs.
Conjunctive normal Form (CNF): In this representation, first all combinations of the inputs are determined for which the output signal is not to be set (which means all lines of the function table for which O = 0). These combinations are inverted and represented as OR operations of the input signals. The outputs of these OR operations are then connected 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 occurs.

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

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





Function table
Disjunctive normal form (DNF)
Conjunctive normal form (CNF)

Figure 2: Structure of basic combinatorial logic circuit

Prioritizing Circuits

Protective functions always take precedence over operational and monitoring functions, which means several 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 can therefore be implemented using a combinatorial 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 output signal O is to be set by the input signal I1, for example, and deleted by another input signal I2, this can no longer be represented combinatorically. O has to remain set when I1 is deleted. Only when I2 is set, is O to be deleted. This makes the effect of I2 dependent on whether I1 was previously set, which means before the current Out state of the system. This state has to be stored in the circuit. Such latching circuits are also called sequential circuits. Storing the Out state can be implemented using a reset-set storage element (R-S 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 R-S flip-flop, either setting or resetting is dominant (refer to Figure 3**).**

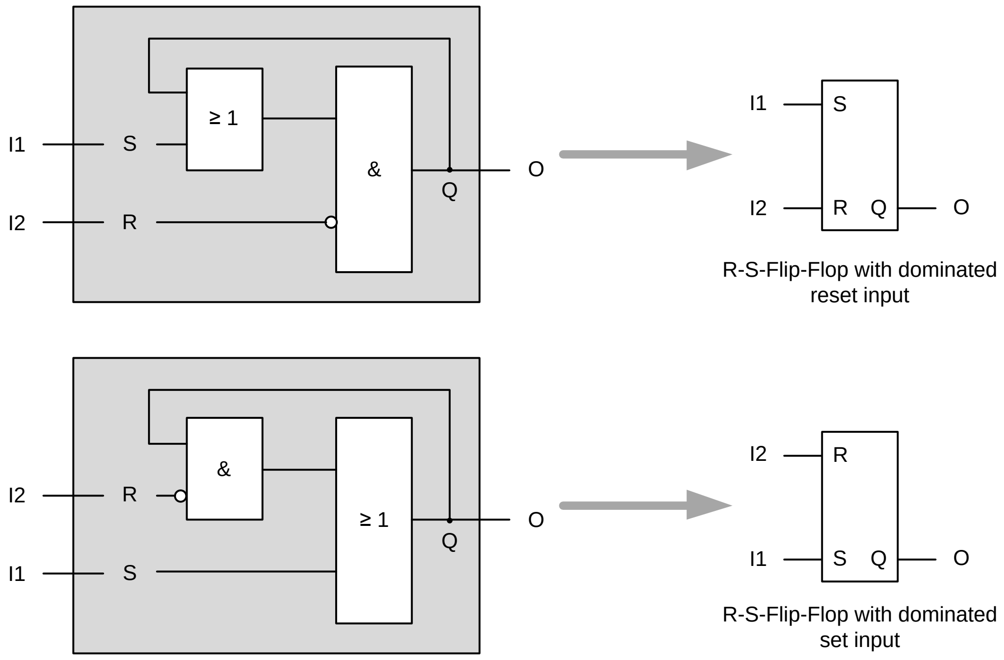

I1
S
≥ 1
&
Q
O
I2
R
I1
S
I2
R Q
O
R-S-Flip-Flop with dominated reset input
&
I2
R
≥ 1
Q
O
I1
S
I2
R
I1
S Q
O
R-S-Flip-Flop with dominated set input

Figure 3: Design and function symbols of R-S flip-flops

Interlocking Circuits

Often, specific control signals must not be set simultaneously. For example, an electrical motor must not be switched in two directions of rotation at the same time, forward movement and reverse movement. The two signals F (forward) and R (reverse) have to lock each other out.

Using two connected R-S flip-flops, a lock can be implemented. Two connections are possible: The lock takes place either by means of set inputs or the reset inputs. Both are shown in Figure 4. It should be noted that locking by means of the reset input works only if the reset input is dominant.

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

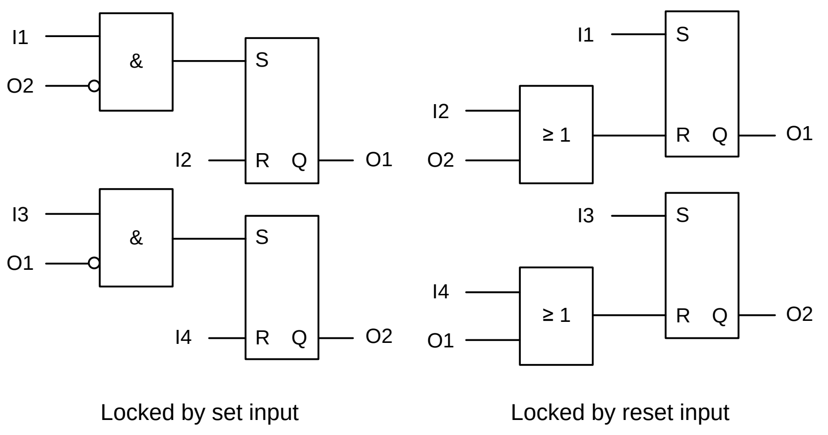


Figure 4: Mutual interlock of two output signals

In some cases the sequence in which specific control signals can be set must also be specified. A sequential lock has to be implemented for this. This also can be implemented by sequencing storage elements. We need as many R-S flip-flops as steps that are to be coordinated. Figure 5 shows a sequential lock for two signals.

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

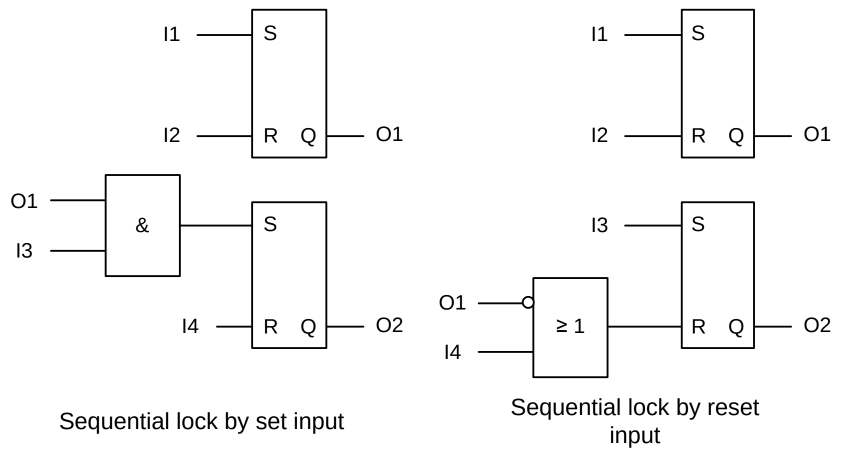


Figure 5: Sequential lock of two output signals

It has to be noted that only activation sequences are implemented with these circuits but no sequences. Setting O2 does not cause O1 to be reset. When locking by means of the reset input, O2 is 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 several events. This principle is explained below using the ***two hand lock*** as an example. It is to prevent that workers get hurt when operating a machine (for example, a press). To prevent that a worker still has one hand in the danger zone of the machine, it can only be activated by operating two buttons at the same time. This task can also be solved with a combinatorial circuit. To prevent that one switch is continuously set with adhesive tape, however, it also has to be ensured that both buttons are pressed within a fixed time span. To this end, **impulse blocks** 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) sets the output signal again. Figure 6 shows the function symbol and the switching performance of an impulse block.

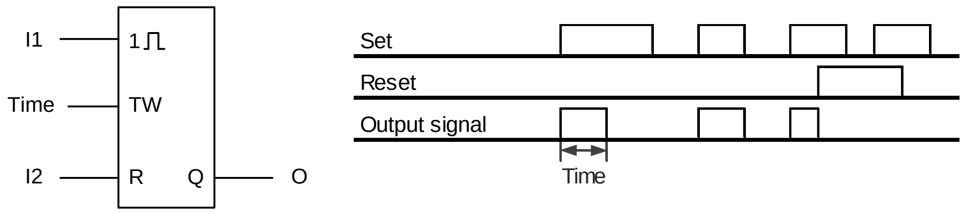


Figure 6: Function symbol and switching performance of an impulse block

The corresponding circuit for two hand locks is shown in Figure 7. If one of the buttons is operated, output OUT of the impulse block is set for the duration T. If the second button is operated while OUT is set, all conditions for the AND element are met and output O is set. Then, the impulse block is jumpered through the OR operation with output O.

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

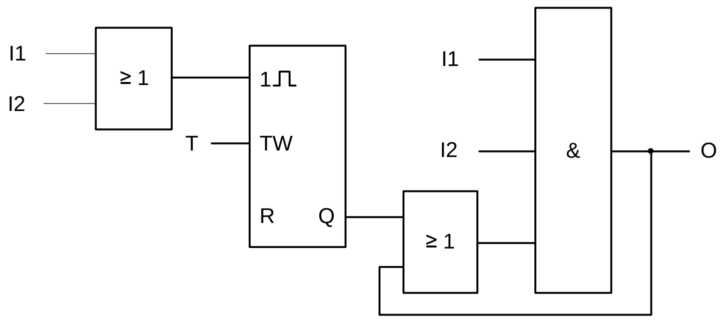


Figure 7: Two-hand lock when using an impulse block

Timing elements are used for a variety of other protective functions; for example, for protective gate controls where open gates close automatically after a specified time, or for motor startup controls where, after a futile start attempt, a rest period for drive recovery is forced.

LITERATURE

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

[2] VDI 2180 (Ed. 2007-04): Safeguarding of industrial process plants by means of process control engineering.

[3] BS EN 61511 (Ed. 2003-03): Functional safety. Safety instrumented systems for the process industry sector.

Step by Step instructions

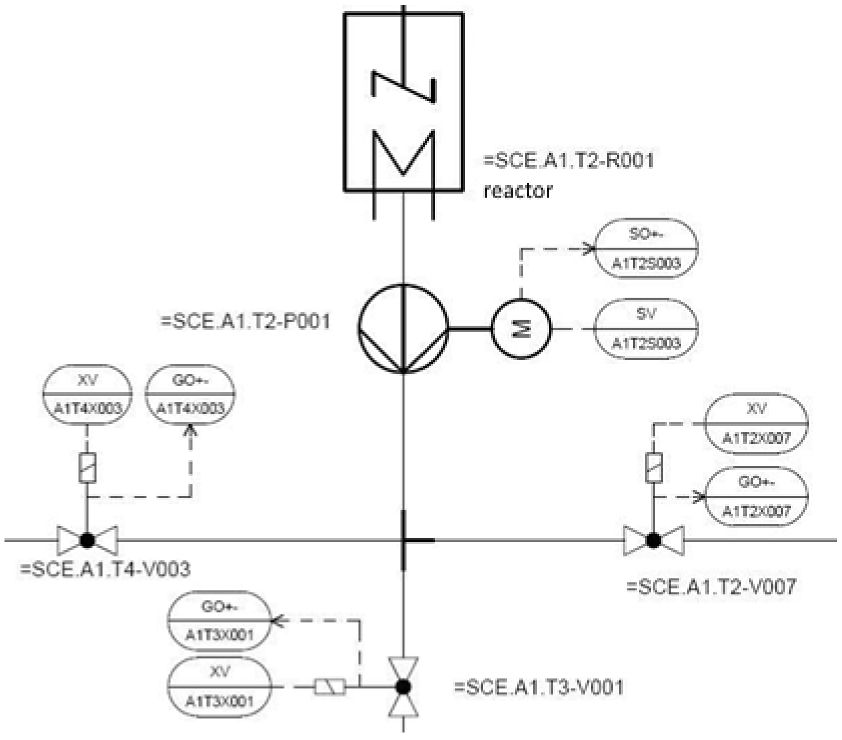
Task

According to the requirements in the chapter ‘Process Description’, the CFCs from the chapter ‘Individual Drive Functions’ will be supplemented with the manual operation of the pump motor =SCE.A1.T2-P001. The following interlocking requirements have to be noted:

– The pump motor must only be switched on if the main switch of the system is switched on and the emergency stop switch is unlatched.

– The pump must not take in air. The minimum level (here: 50ml) in the reactor =SCE.A1.T2-R002 is known numerically and is evaluated by means of the measured level.

– The pump must not press liquid against a closed valve. This means when the pump is switched on, either valve=SCE.A1.T3-V001, valve=SCE.A1.T2-V007 or valve =SCE.A1.T4-V003 must be open.



Note: For the approach to the solution, please note the details regarding latching circuits in the theoretical part.

TRAINING OBJECTIVE

In this chapter, the students will learn the following:

– Implementing expanded boundary conditions and manual operation

– Setting up connections between CFCs

– Additional possibilities for programming with CFCs

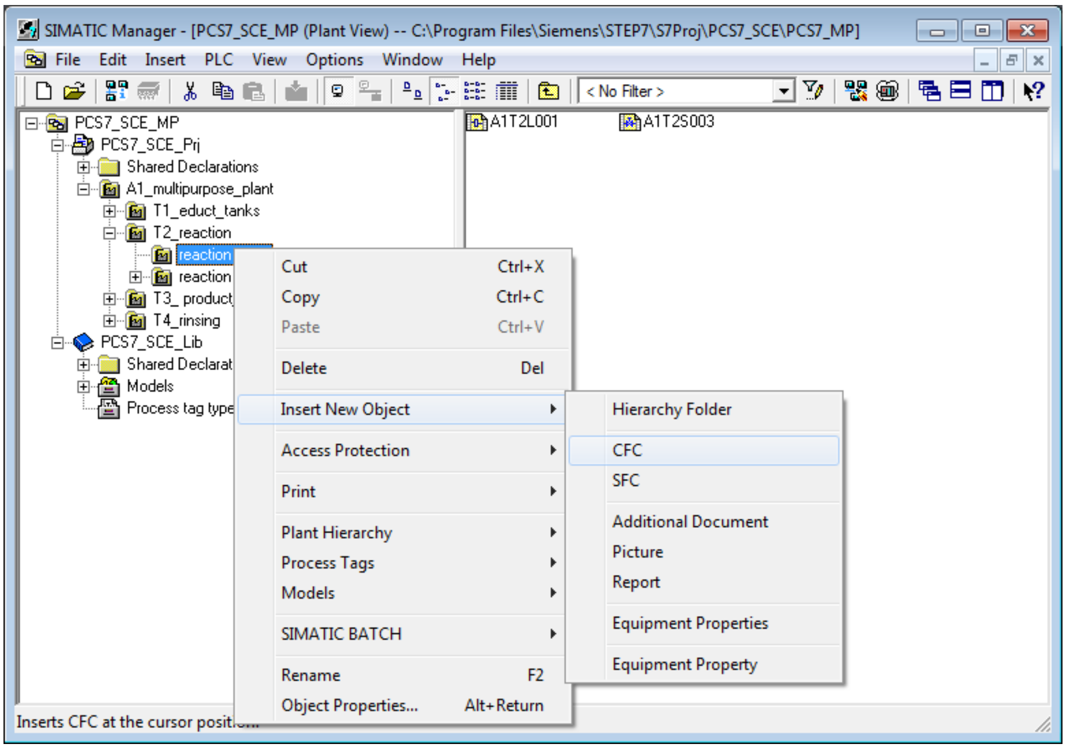
– Utilizing additional sheets in the CFCs

These instructions are based on the project ‘PCS7_SCE_0104_Ueb_R1305_en.zip‘.

PROGRAMMING

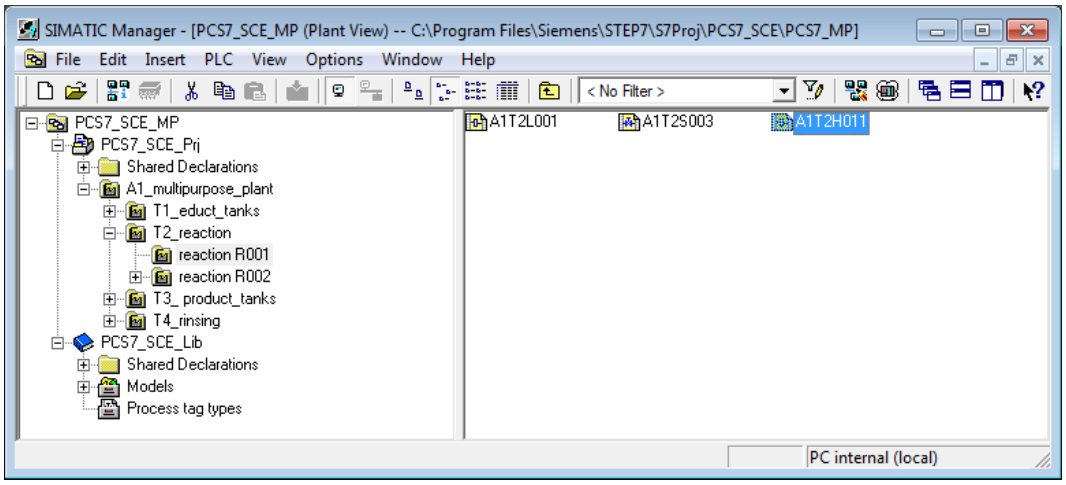
1. To program the manual operation for draining Reactor R001, set up a new CFC in the SIMATIC Manager in the Plant View in the folder Reactor R001 of the subsystem T2_Reaction.

   (→ SIMATIC Manager → View → Plant View → Reactor R001 → Insert New Object → CFC)



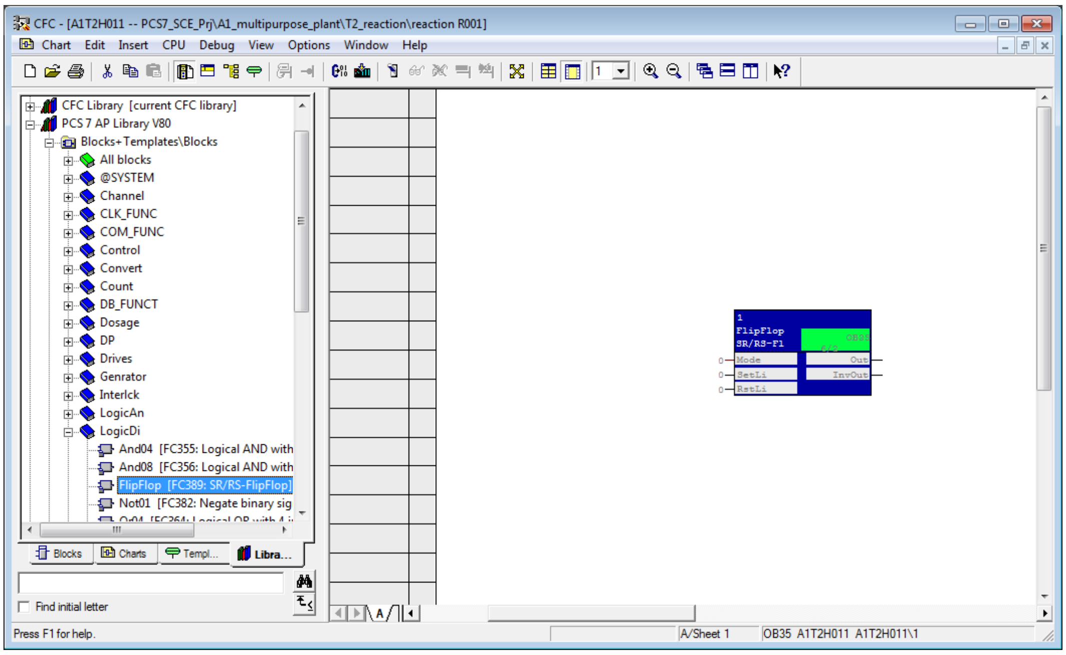
2. The chart is then renamed 'A1T2H011' and opened with a double click.

   (→ A1T2H011)



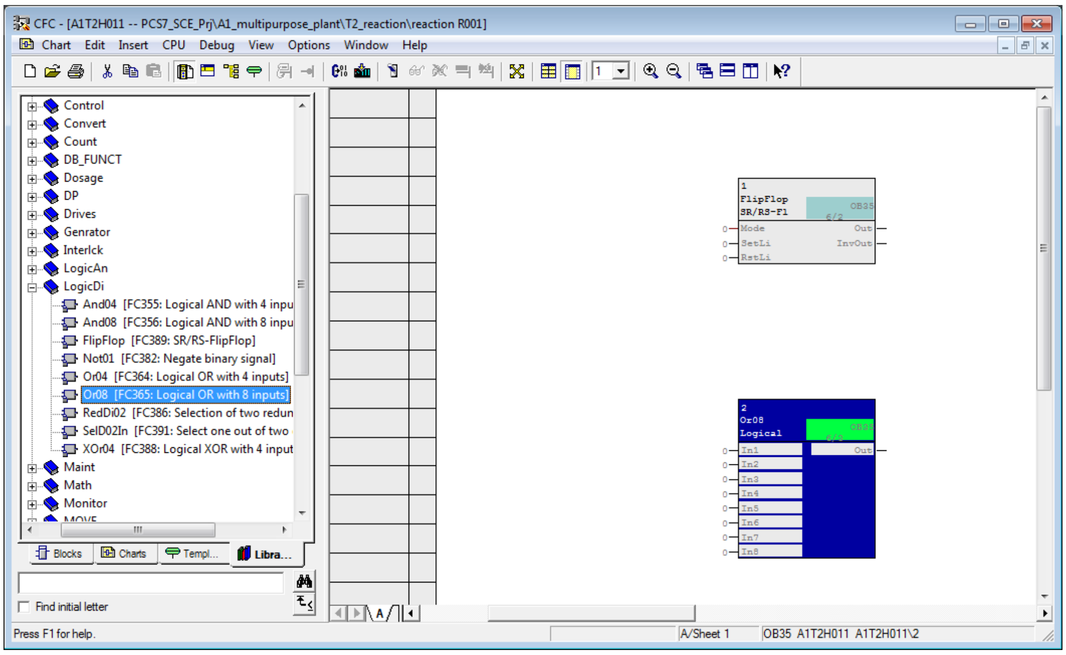
3. In the CFC Editor, drag the 'FlipFlop' block from the 'LogicDi' folder in the ‘Block’ catalog to the first sheet of the chart. You now have a storage element. For the reset or switch-off to be dominant, the mode has to be set to ‘1’.

   (→ Library → PCS 7 AP Library V8.1 → Blocks+Templates\Blocks → LogicDi → FlipFlop → Mode → '1‘)



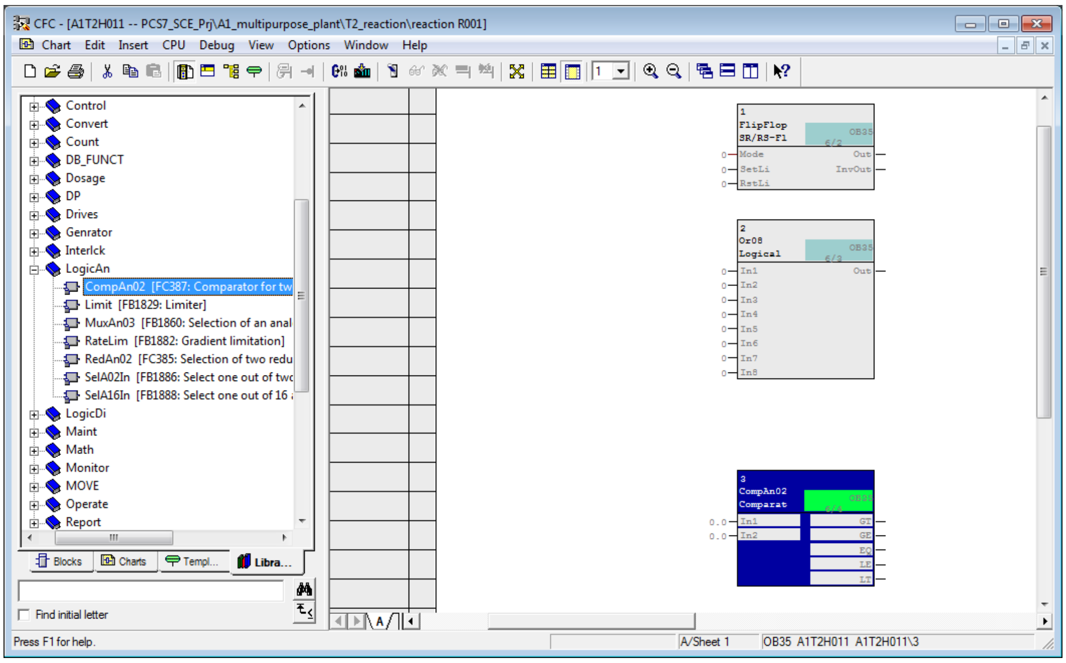
Note: Additional information about the blocks used is provided in the detailed online help. Highlight the corresponding block and press 'F1' on the keyboard.

4. Next, from the 'LogicDi' folder, drag the block 'Or08' to the chart.


CFC - [A1T2H011 -- PCS7_SCE_Prj\A1_multipurpose_plant\T2_reaction\reaction R001]
Chart Edit Insert CPU Debug View Options Window Help
Control
Convert
Count
DB_FUNCT
Dosage
DP
Drives
Genrator
Interlck
LogicAn
LogicDi
And04 [FC355: Logical AND with 4 inpu
And08 [FC356: Logical AND with 8 inpu
FlipFlop [FC389: SR/RS-FlipFlop]
Not01 [FC382: Negate binary signal]
Or04 [FC364: Logical OR with 4 inputs]
Or08 [FC365: Logical OR with 8 inputs]
RedDi02 [FC386: Selection of two redun
SelD02In [FC391: Select one out of two
XOr04 [FC388: Logical XOR with 4 input
Maint
Math
Monitor
Blocks
Charts
Templ...
Libra...
Find initial letter
1 FlipFlop SR/RS-F1 OB35 6/2
Mode SetLi RstLi Out InvOut
2 Or08 Logical
In1 In2 In3 In4 In5 In6 In7 In8 Out
Press F1 for help.
A/Sheet 1
OB35 A1T2H011 A1T2H011\2

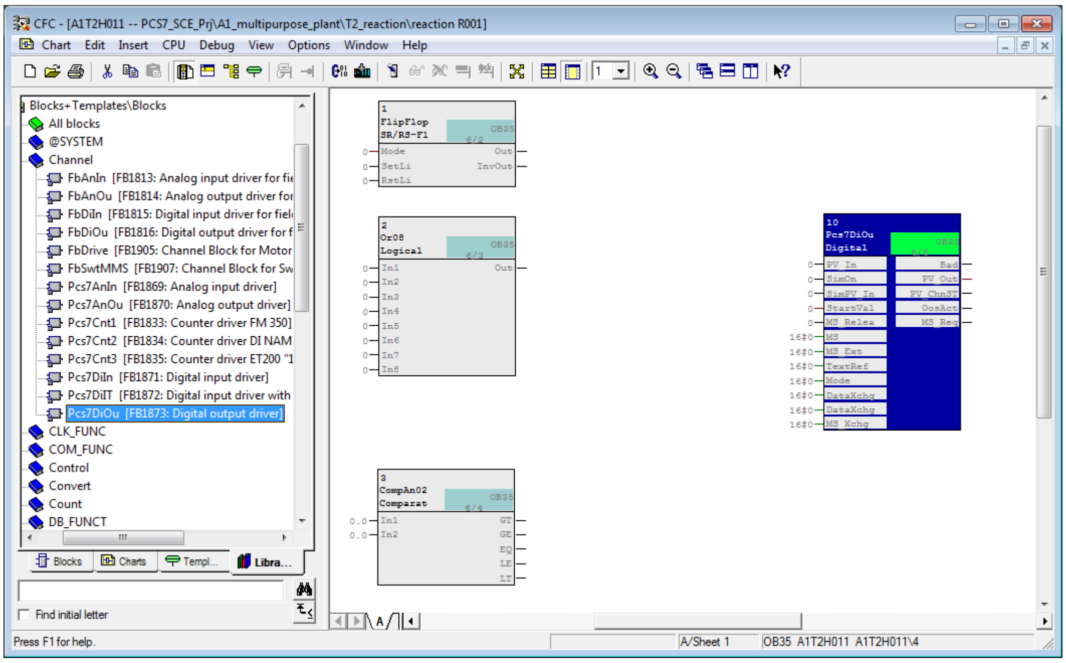
5. Now, from the 'LogicAn' folder in the PCS 7 AP Library V81 in the 'Library' tab, drag the block 'CompAn02' into the chart. You will need it to take into account the level of Reactor R001 present as a numerical value for the interlock.

(→ Libraries → PCS 7 AP Library V8.1 → Blocks+Templates\Blocks → LogicAn → CompAn02)


CFC - [A1T2H011 -- PCS7_SCE_Prj\A1_multipurpose_plant\T2_reaction\reaction R001]
Chart Edit Insert CPU Debug View Options Window Help
Control
Convert
Count
DB_FUNCT
Dosage
DP
Drives
Genrator
Interlck
LogicAn
CompAn02 [FC387: Comparator for tw
Limit [FB1829: Limiter]
MuxAn03 [FB1860: Selection of an anal
RateLim [FB1882: Gradient limitation]
RedAn02 [FC385: Selection of two redu
SelA02In [FB1886: Select one out of two
SelA16In [FB1888: Select one out of 16
LogicDi
Maint
Math
Monitor
MOVE
Operate
Report
Blocks
Charts
Templ...
Libra...
Find initial letter
1 FlipFlop SR/RS-F1 OB35 6/2
2 Or08 Logical OB35 6/3
3 CompAn02 Comparat
In1 In2 GT GE EQ LE LT
Press F1 for help.
A/Sheet 1
OB35 A1T2H011 A1T2H011\3

6. Next, drag the driver block for a digital output signal 'Pcs7DiOu' into the chart.

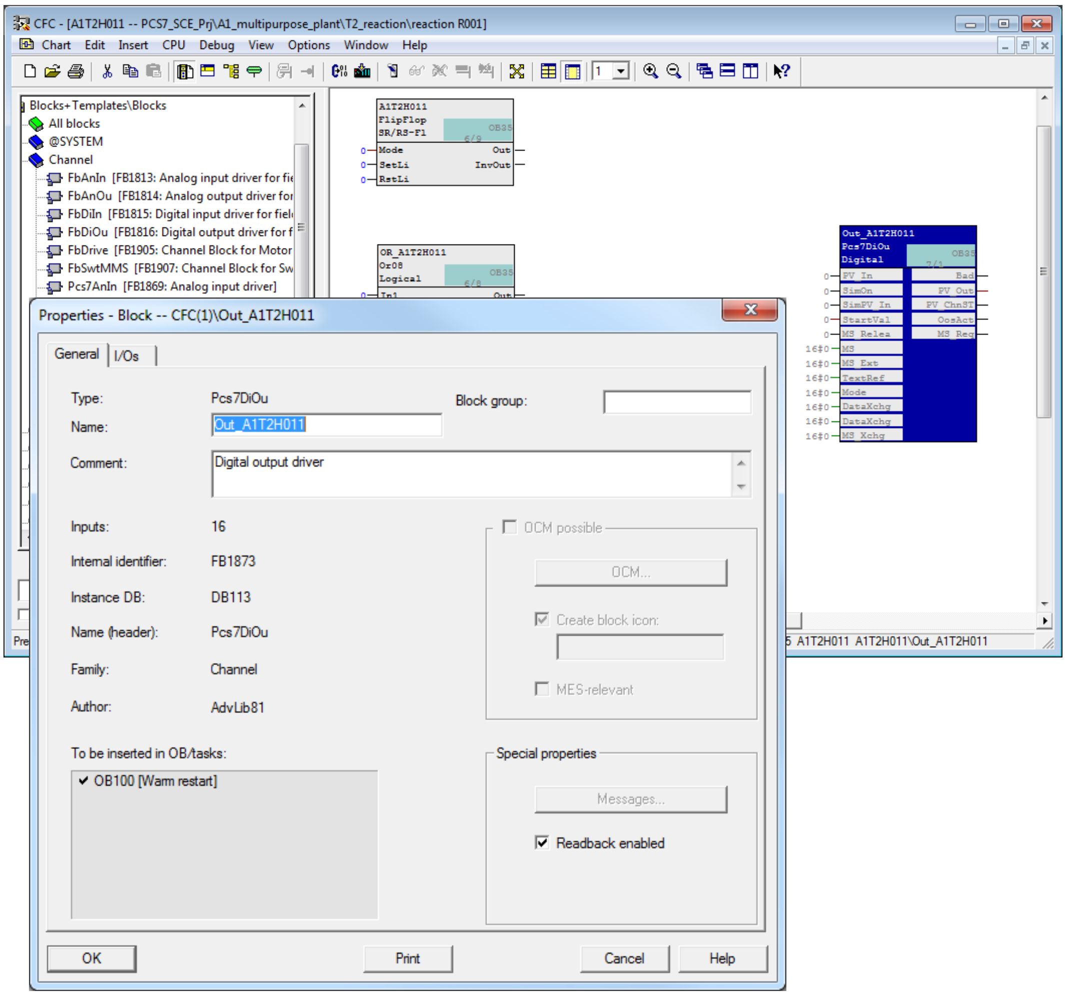
   (→ Libraries → PCS 7 AP Library V8.1 → Blocks+Templates\Blocks → Channel → Pcs7DiOu)



CFC - [A1T2H011 -- PCS7_SCE_Prj\A1_multipurpose_plant\T2_reaction\reaction R001]

Blocks+Templates\Blocks
- All blocks
- @SYSTEM
- Channel
  - FbAnIn [FB1813: Analog input driver for fie
  - FbAnOu [FB1814: Analog output driver for
  - FbDiIn [FB1815: Digital input driver for fiel
  - FbDiOu [FB1816: Digital output driver for f
  - FbDrive [FB1905: Channel Block for Motor
  - FbSwtMMS [FB1907: Channel Block for Sw
  - Pcs7AnIn [FB1869: Analog input driver]
  - Pcs7AnOu [FB1870: Analog output driver]
  - Pcs7Cnt1 [FB1833: Counter driver FM 350]
  - Pcs7Cnt2 [FB1834: Counter driver DI NAM
  - Pcs7Cnt3 [FB1835: Counter driver ET200 "1
  - Pcs7DiIn [FB1871: Digital input driver]
  - Pcs7DiIT [FB1872: Digital input driver with
  - Pcs7DiOu [FB1873: Digital output driver]
- CLK_FUNC
- COM_FUNC
- Control
- Convert
- Count
- DB_FUNCT

7. Assign names to the blocks as shown.



CFC - [A1T2H011 -- PCS7_SCE_Prj\A1_multipurpose_plant\T2_reaction\reaction R001]

Properties - Block -- CFC(1)\Out_A1T2H011

General | I/Os

Type:	Pcs7DiOu
Name:	Out_A1T2H011
Block group:	
Comment:	Digital output driver
Inputs:	16
Internal identifier:	FB1873
Instance DB:	DB113
Name (header):	Pcs7DiOu
Family:	Channel
Author:	AdvLib81

To be inserted in OB/tasks:
✔ OB100 [Warm restart]

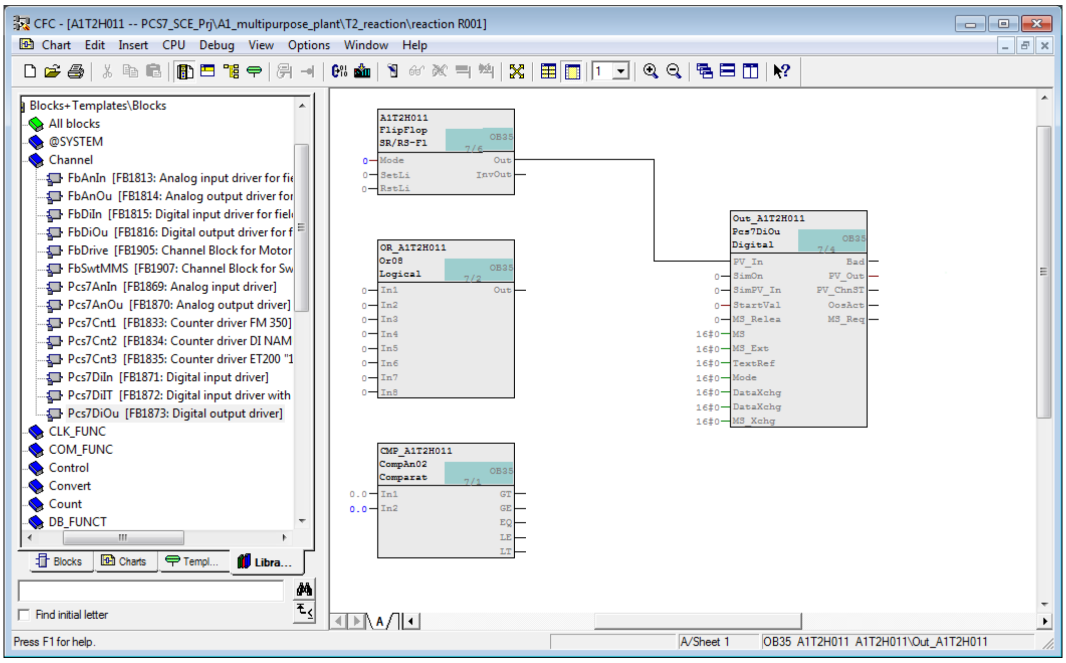
☐ OCM possible
☑ Create block icon:
☐ MES-relevant

Special properties
☑ Readback enabled

OCM...
Messages...
OK
Print
Cancel
Help
A1T2H011 FlipFlop SR/RS-F1
OR_A1T2H011 Or08 Logical
Out_A1T2H011 Pcs7DiOu Digital
A1T2H011 A1T2H011\Out_A1T2H011

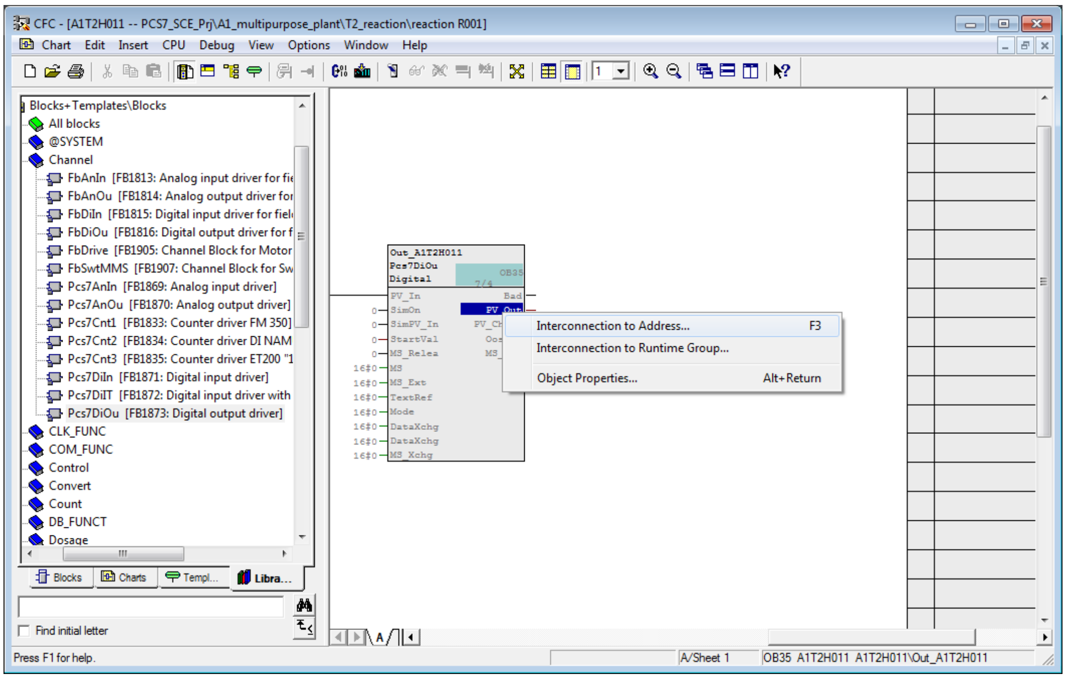
8. Now, the first interconnection of the blocks among each other is established. To this end, connect output 'Out' of the 'FlipFlop' block with input 'PV_In' of the 'Pcs7DiOu' block. The alignment that this connection shows is automatic and cannot be changed in the CFC Editor.

   (→ FlipFlop → Out → Pcs7DiOu → PV_In)


CFC - [A1T2H011 -- PCS7_SCE_Prj\A1_multipurpose_plant\T2_reaction\reaction R001]
Chart Edit Insert CPU Debug View Options Window Help
Blocks+Templates\Blocks
All blocks
@SYSTEM
Channel
FbAnIn [FB1813: Analog input driver for fie
FbAnOu [FB1814: Analog output driver for
FbDiIn [FB1815: Digital input driver for fiel
FbDiOu [FB1816: Digital output driver for f
FbDrive [FB1905: Channel Block for Motor
FbSwtMMS [FB1907: Channel Block for Sw
Pcs7AnIn [FB1869: Analog input driver]
Pcs7AnOu [FB1870: Analog output driver]
Pcs7Cnt1 [FB1833: Counter driver FM 350]
Pcs7Cnt2 [FB1834: Counter driver DI NAM
Pcs7Cnt3 [FB1835: Counter driver ET200 "1
Pcs7DiIn [FB1871: Digital input driver]
Pcs7DiIT [FB1872: Digital input driver with
Pcs7DiOu [FB1873: Digital output driver]
CLK_FUNC
COM_FUNC
Control
Convert
Count
DB_FUNCT
Blocks Charts Templ... Libra...
Find initial letter
A1T2H011 FlipFlop SR/RS-Fl OB35 7/6
Mode SetLi RstLi Out InvOut
OR_A1T2H011 Or08 Logical OB35 7/2
In1 In2 In3 In4 In5 In6 In7 In8 Out
CMP_A1T2H011 CompAn02 Comparat OB35 7/1
In1 In2 GT GE EQ LE LT
Out_A1T2H011 Pcs7DiOu Digital OB35 7/4
PV_In SimOn SimPV_In StartVal MS_Relea MS MS_Ext TextRef Mode DataXchg DataXchg MS_Xchg
Bad PV_Out PV_ChnST OosAct MS_Req
Press F1 for help.
A/Sheet 1
OB35 A1T2H011 A1T2H011\Out_A1T2H011

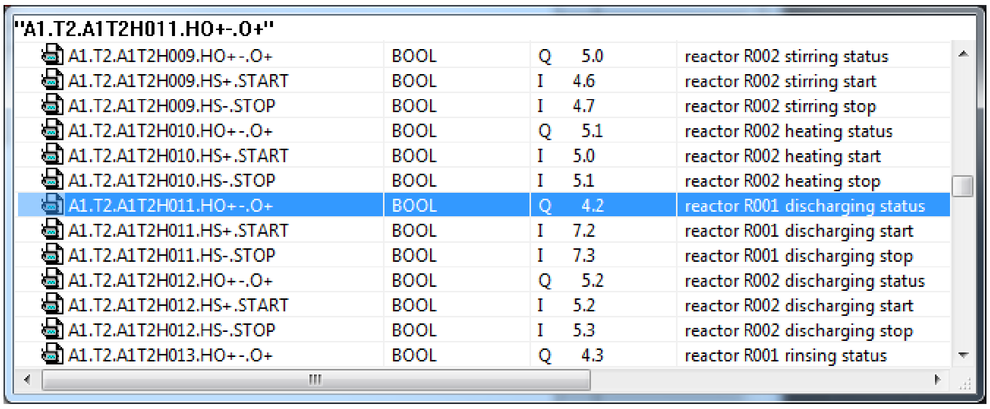
9. To display the status of the operator request, we now interconnect output 'PV_Out' of the block 'Pcs7DiOu' with the corresponding address from the symbol table.

   (Pcs7DiOu → PV_Out → Interconnection to Address)


CFC - [A1T2H011 -- PCS7_SCE_Prj\A1_multipurpose_plant\T2_reaction\reaction R001]
Chart Edit Insert CPU Debug View Options Window Help
Blocks+Templates\Blocks
Out_A1T2H011 Pcs7DiOu Digital OB35 7/4
Interconnection to Address... F3
Interconnection to Runtime Group...
Object Properties... Alt+Return
Press F1 for help.
A/Sheet 1
OB35 A1T2H011 A1T2H011\Out_A1T2H011

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 request.

    (→ A1.T2.A1T2H011.HO+-.O+)

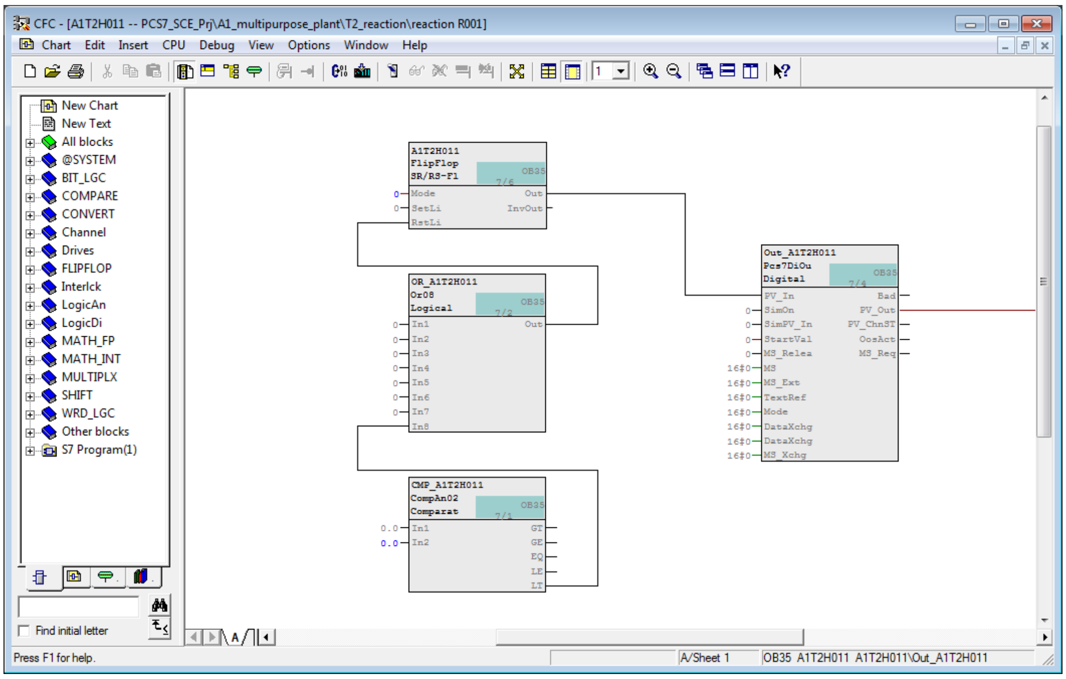


"A1.T2.A1T2H011.HO+-.O+"

A1.T2.A1T2H009.HO+-.O+	BOOL	Q	5.0	reactor R002 stirring status
A1.T2.A1T2H009.HS+.START	BOOL	I	4.6	reactor R002 stirring start
A1.T2.A1T2H009.HS-.STOP	BOOL	I	4.7	reactor R002 stirring stop
A1.T2.A1T2H010.HO+-.O+	BOOL	Q	5.1	reactor R002 heating status
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 discharging status
A1.T2.A1T2H011.HS+.START	BOOL	I	7.2	reactor R001 discharging start
A1.T2.A1T2H011.HS-.STOP	BOOL	I	7.3	reactor R001 discharging stop
A1.T2.A1T2H012.HO+-.O+	BOOL	Q	5.2	reactor R002 discharging status
A1.T2.A1T2H012.HS+.START	BOOL	I	5.2	reactor R002 discharging start
A1.T2.A1T2H012.HS-.STOP	BOOL	I	5.3	reactor R002 discharging stop
A1.T2.A1T2H013.HO+-.O+	BOOL	Q	4.3	reactor R001 rinsing status

11. Now, another interconnection is established among the blocks. Simply click on the output of the 'Or08' block and then on the input 'RstLi' of the 'FlipFlop' block. Next, the output 'LT' of block 'CompAn02' is connected to an input of the 'Or08' block.

    (→ Or08.Out → FlipFlop.RstLi → CompAn02.LT → Or08.In8)



CFC - [A1T2H011 -- PCS7_SCE_Prj\A1_multipurpose_plant\T2_reaction\reaction R001]

Chart
Edit
Insert
CPU
Debug
View
Options
Window
Help
New Chart
New Text
All blocks
@SYSTEM
BIT_LGC
COMPARE
CONVERT
Channel
Drives
FLIPFLOP
Interlck
LogicAn
LogicDi
MATH_FP
MATH_INT
MULTIPLX
SHIFT
WRD_LGC
Other blocks
S7 Program(1)
Find initial letter
A1T2H011 FlipFlop SR/RS-Fl OB35 7/6
Mode SetLi RstLi Out InvOut
OR_A1T2H011 Or08 Logical OB35 7/2
In1 In2 In3 In4 In5 In6 In7 In8 Out
CMP_A1T2H011 CompAn02 Comparat OB35 7/1
In1 In2 GT GE EQ LE LT
Out_A1T2H011 Pcs7DiOu Digital OB35 7/4
PV_In SimOn SimPV_In StartVal MS_Relea MS MS_Ext TextRef Mode DataXchg DataXchg MS_Xchg
Bad PV_Out PV_ChnST OosAct MS_Req
A/Sheet 1

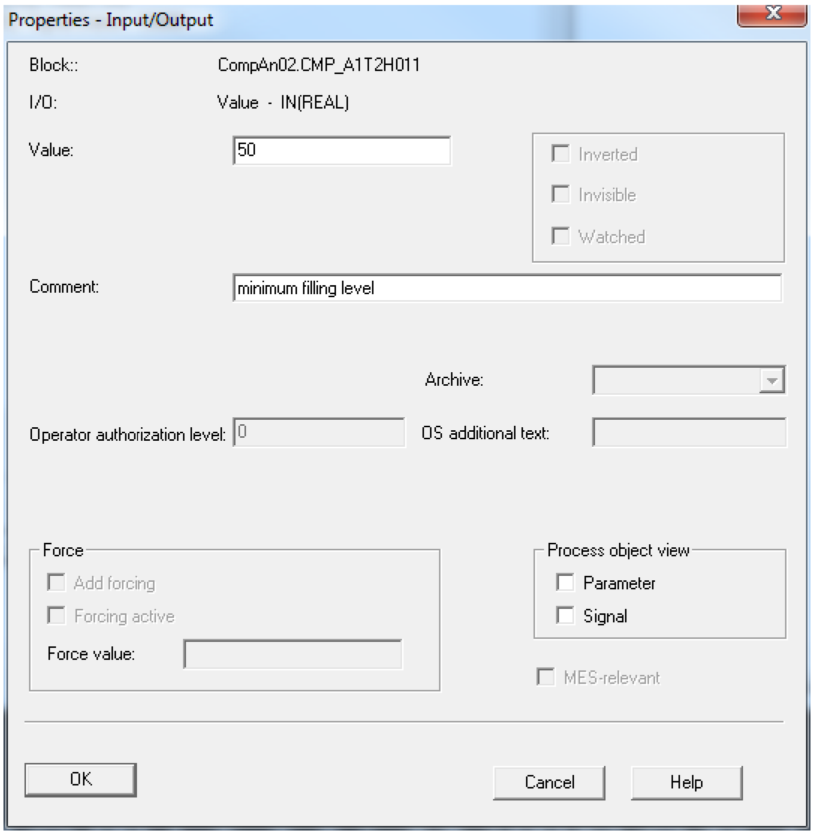
Press F1 for help.

A/Sheet 1 OB35 A1T2H011 A1T2H011\Out_A1T2H011

Note: The output 'LT' of block 'CompAn02' 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 accept this change with OK.

    (→ CompAn02 → In2 → Value → 50.0 → OK)



Properties - Input/Output

Block:: CompAn02.CMP_A1T2H011

I/O: Value - IN(REAL)

Value: 50

☐ Inverted
☐ Invisible
☐ Watched

Comment: minimum filling level

Archive:

Operator authorization level: 0

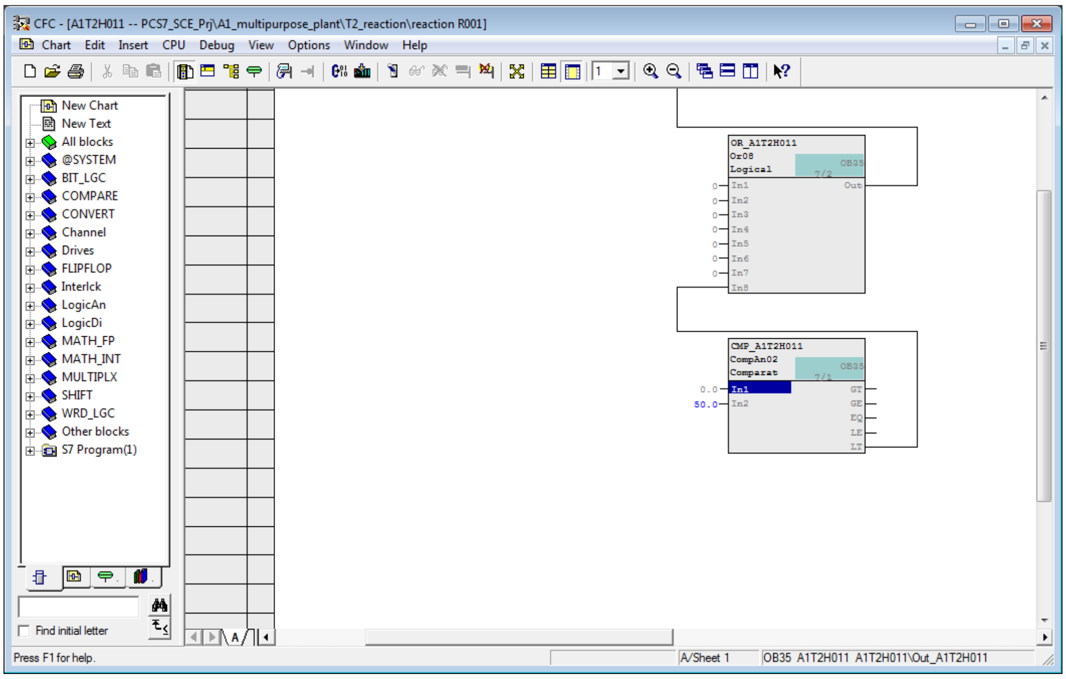
OS additional text:

Force
☐ Add forcing
☐ Forcing active
Force value:

Process object view
☐ Parameter
☐ Signal

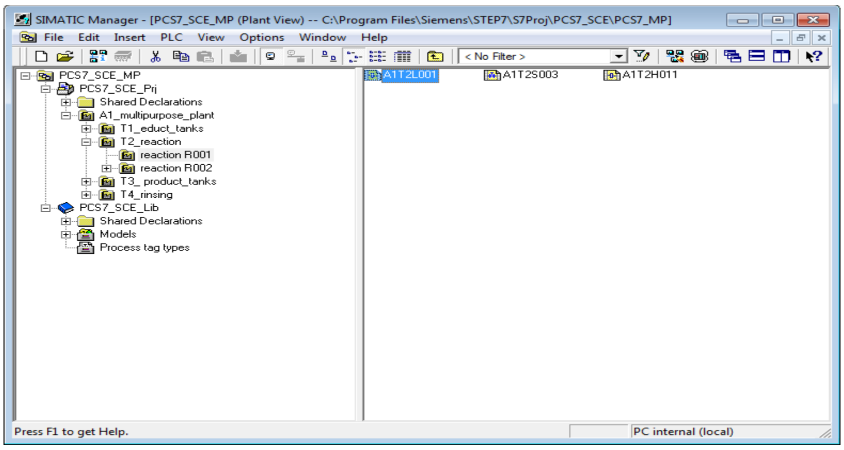
☐ MES-relevant

13. Now we establish a cross-chart connection of input 'In1' to the measured level of reactor =SCE.A1.T2.R001. To do this, highlight 'In1' at block 'CompAn02'. (→ CompAn02 → In1)

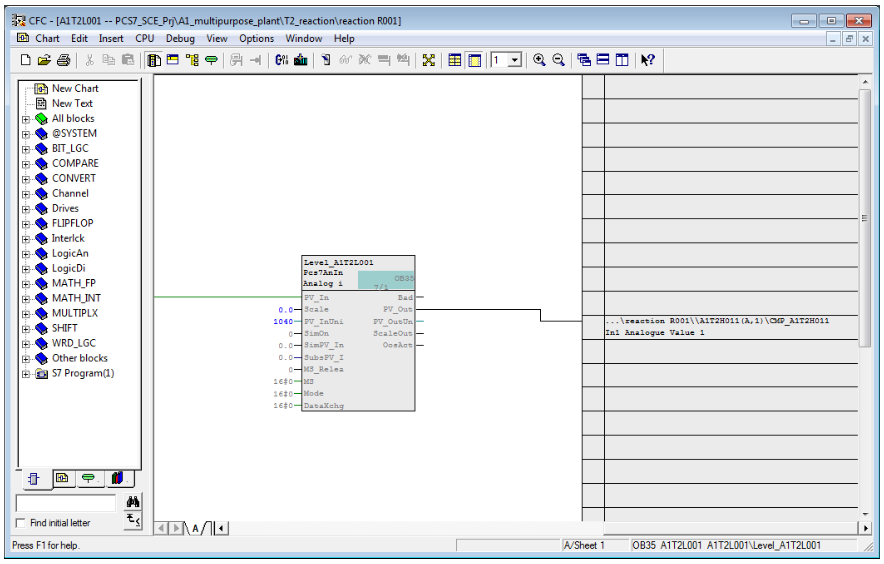


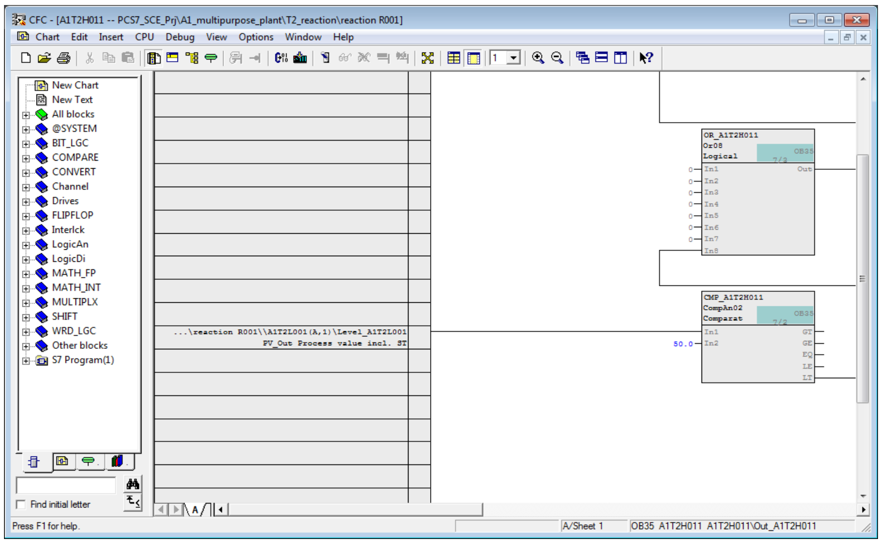
CFC - [A1T2H011 -- PCS7_SCE_Prj\A1_multipurpose_plant\T2_reaction\reaction R001]

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

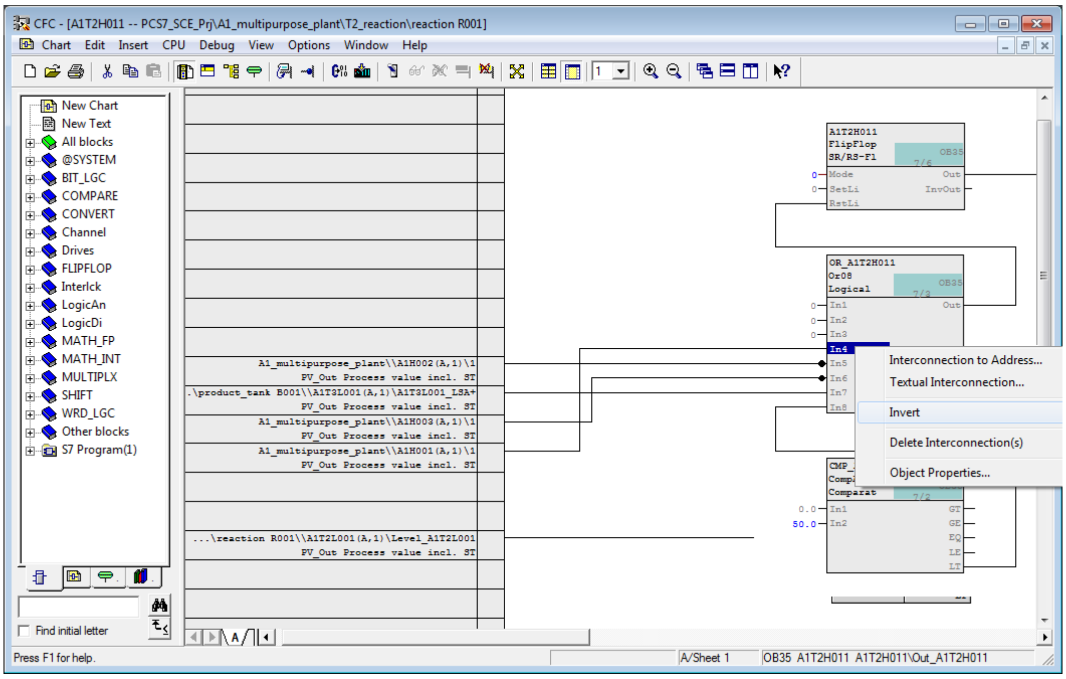


15. In the open chart 'A1T2L001' at block 'Pcs7AnIn' click on the output 'PV_Out'. The cross-chart connection is set up and displayed for both charts on the margin bar. For chart 'A1T2L001' the destination of the connection is shown on the right. For chart 'A1T2H011' the source of the connection is shown on the left. (→ A1T2L001 → Pcs7AnIn → PV_Out)

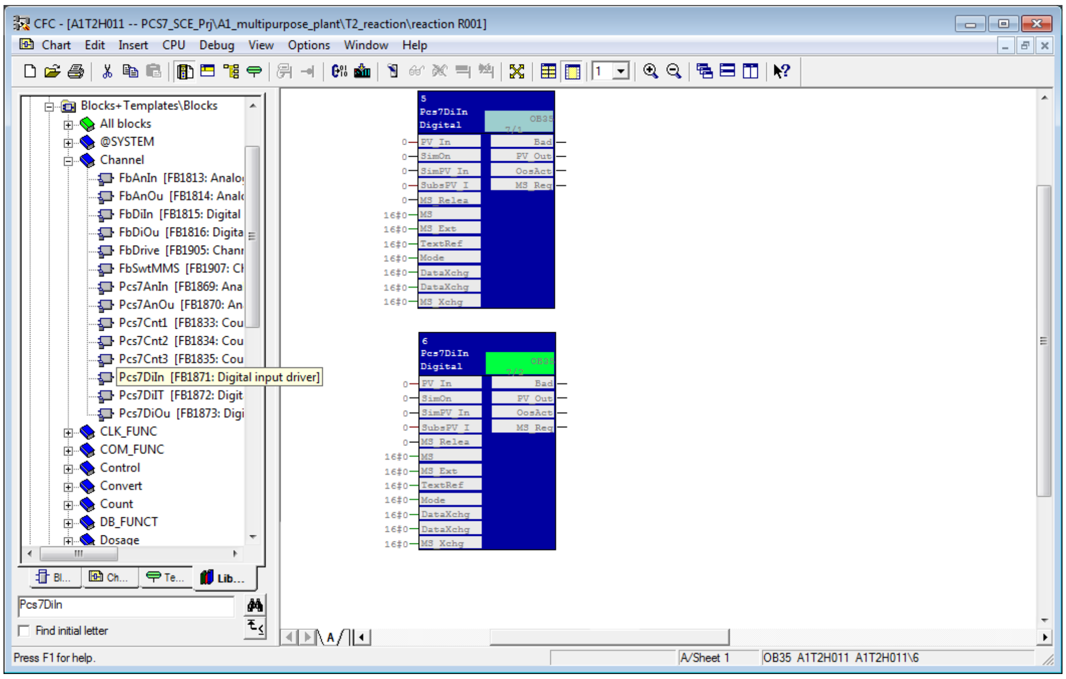




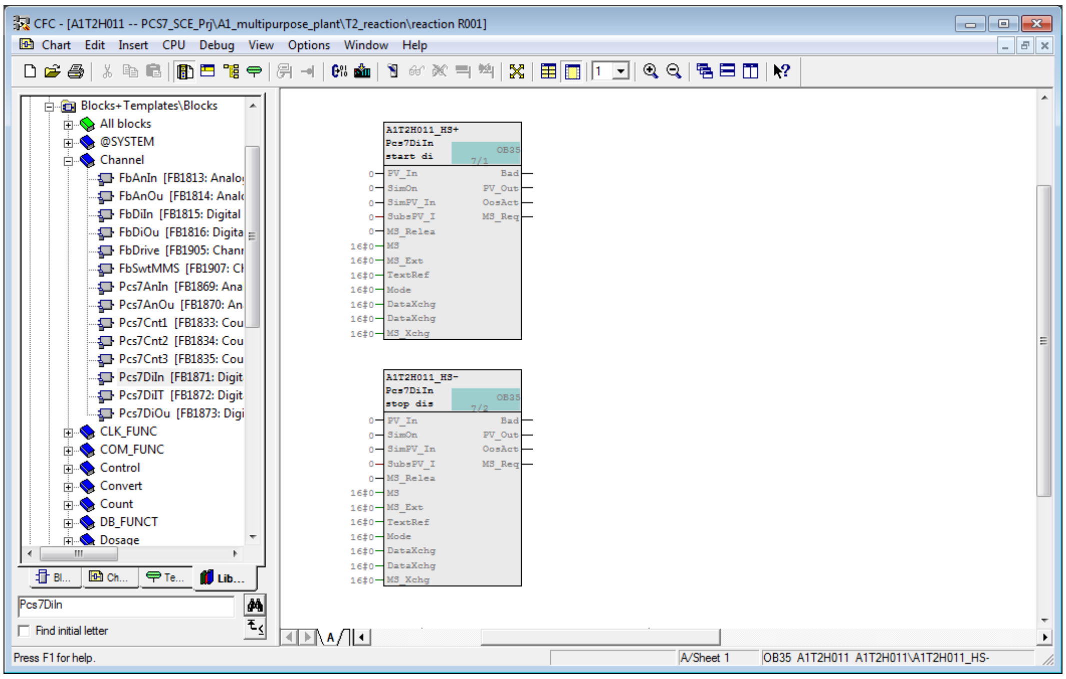
16. Now, the signals that request the reset have to be connected to block ‘Or08‘. These signals are shown below and are also listed in Table 1. Please note that some signals are connected inverted. To do this, right click on the connection to open the shortcut menu and select Invert.



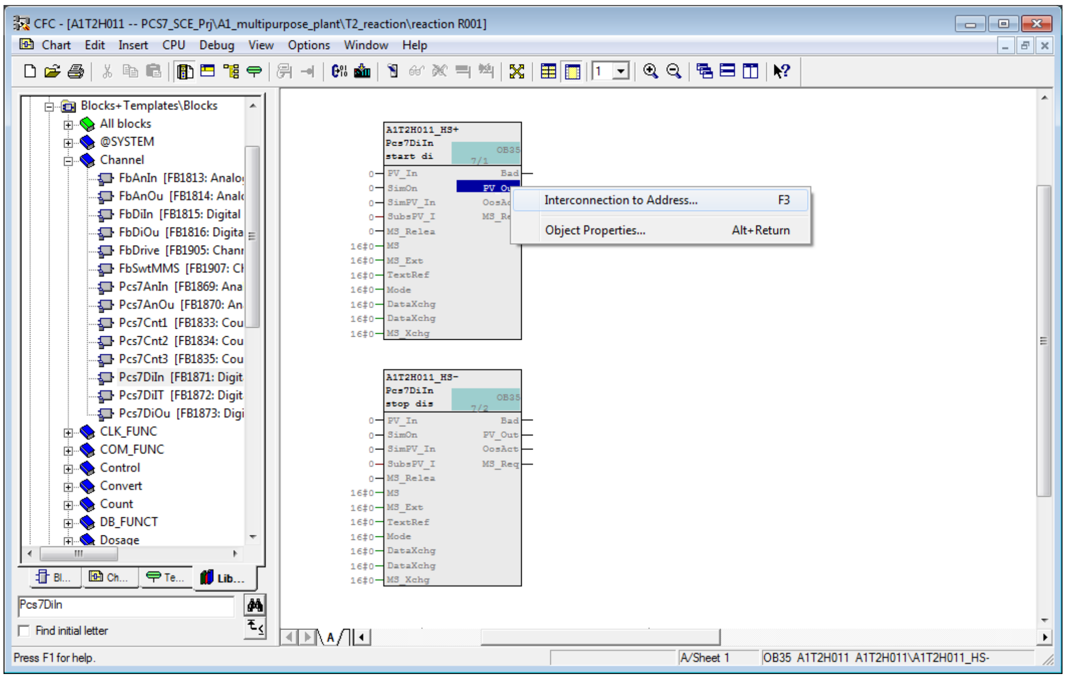
17. To complete the chart, the two signals needed for manually operating A1T2H011 Start and Stop are added. To enter them, insert two driver blocks for one digital input signal. (→ Libraries → PCS 7 AP Library V8.1 → Blocks+Templates\Blocks → Channel → Pcs7DiIn)



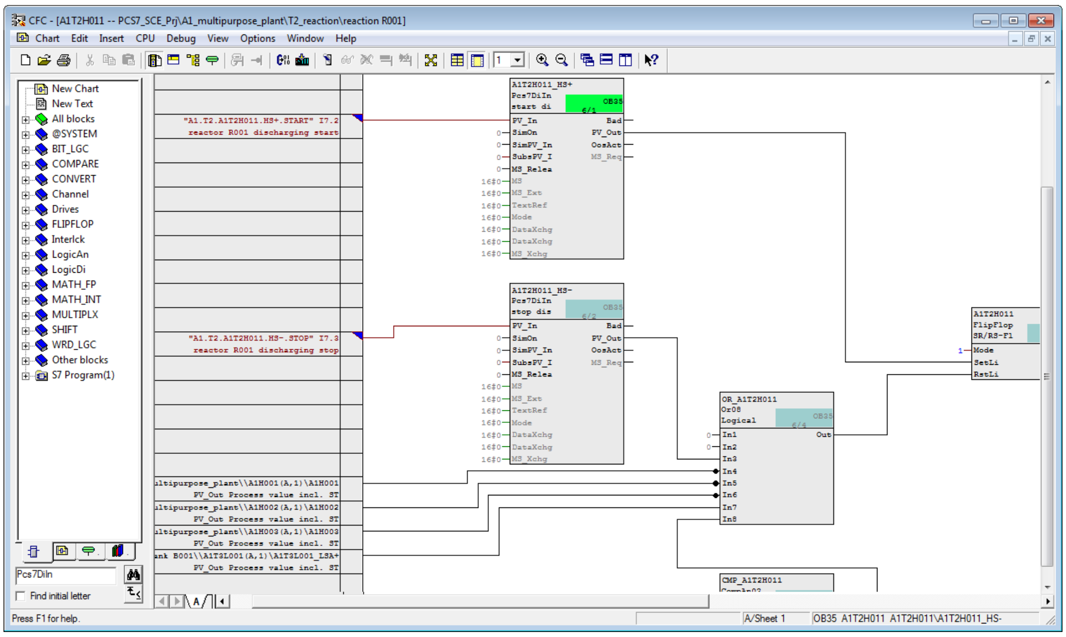
18. Next, to differentiate the blocks, change the name and add a comment. In the result, the blocks can now be easily distinguished.
(Pcs7DiIn → Object Properties → Name: A1T2H011_HS+ and A1T2H011_HS- → Comment: Start draining and stop draining)


CFC - [A1T2H011 -- PCS7_SCE_Prj\A1_multipurpose_plant\T2_reaction\reaction R001]
A1T2H011_HS+
Pcs7DiIn
start di
A1T2H011_HS-
Pcs7DiIn
stop dis
Press F1 for help.
A/Sheet 1
OB35 A1T2H011 A1T2H011\A1T2H011_HS-

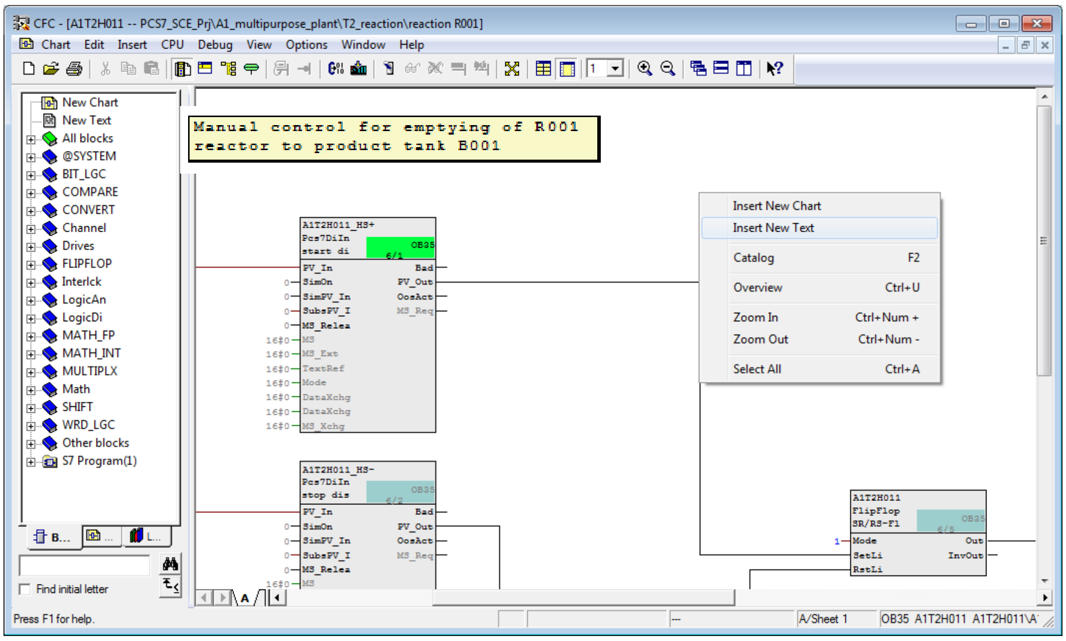
19. Then, the two driver blocks are connected to the respective signal. (Pcs7DiIn → PV_In → Interconnection to Address → A1T2H011.HS+.Start and A1T2H011.HS-.Stop)


CFC - [A1T2H011 -- PCS7_SCE_Prj\A1_multipurpose_plant\T2_reaction\reaction R001]
A1T2H011_HS+
Pcs7DiIn
start di
Interconnection to Address... F3
Object Properties... Alt+Return
A1T2H011_HS-
Pcs7DiIn
stop dis
Press F1 for help.
A/Sheet 1
OB35 A1T2H011 A1T2H011\A1T2H011_HS-

20. Now, we have to connect the output ‘PV_Out‘ of the block for starting with the input ‘SetLi‘ of the flipflop, and the output ‘PV_Out‘ of the block for stopping with an input of the block ‘Or08‘.



21. Next, we enter 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 of R001 reactor to product tank B001")



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

Table 1: Input Interconnections in Chart ‘A1T2H011/Sheet1’

Input	**Interconnected with**	**Inverted**
Pcs7DiIn.HS+.PV_In	‘A1.T2.A1T2H011.HS+.START‘/I7.2/ Reactor R001 Start draining	No
Pcs7DiIn.HS-.PV_In	‘A1.T2.A1T2H011.HS-.STOP’ / I7.3 / Reactor R001 Stop draining	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) / 1 PV_Out Process value incl. ST	No
CompAn02.In2	50.0	
FlipFlop.Mode	1	

⚠

Note: ‘A1T2L001(A,1) / 1 PV_out Process value incl. ST’ means:

– Chart A1T2L001

– Subchart A, Sheet 1

– Block 1

– Connection PV_out Process value incl. ST (STRUCT consisting of value and ST)

Table 2: Block Interconnections in Chart ‘A1T2H011/Sheet1’

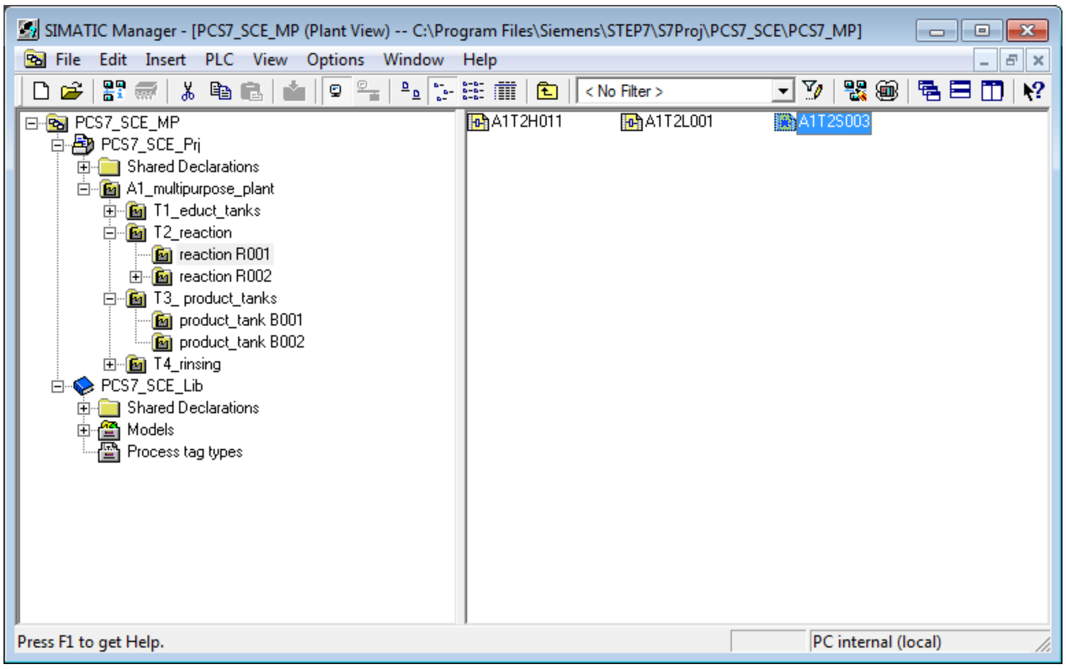
Input:	**Output:**	**Inverted**
FlipFlop.SetLi	Pcs7DiIn.HS+.PV_Out	No
FlipFlop.RstLi	Or08.Out	No
Or08.In3	Pcs7DiIn.HS-.PV_Out	No
Or08.In8	CompAn02.LT	No
Pcs7DiOu.PV_In	FlipFlop.Out	No

Table 3: Output Interconnections in Chart ‘A1T2H011/Sheet1’

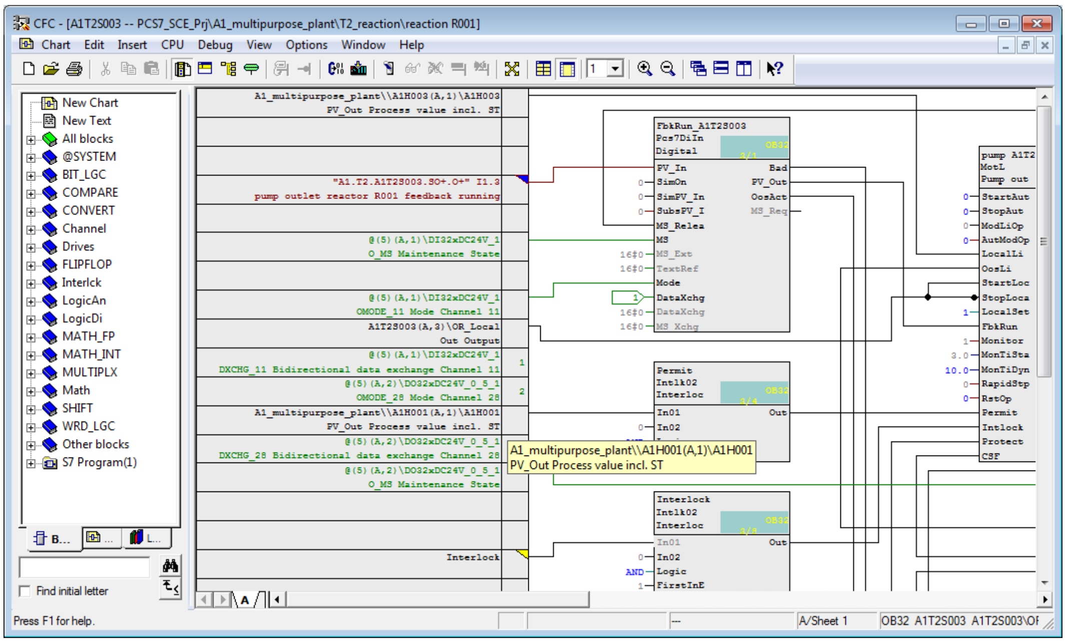
Output:	**Interconnection to:**	**Inverted**
Pcs7DiOu.PV_OUT	‘A1.T2.A1T2H011.HO+-.0+’ / Q4.2 / Reactor R001 Drain status value	No

23. Next, we establish the interlocking conditions for the pump ‘Drainage Reactor R001’. To this end, open CFC ‘A1T2S003’ in the Plant View with a double click.

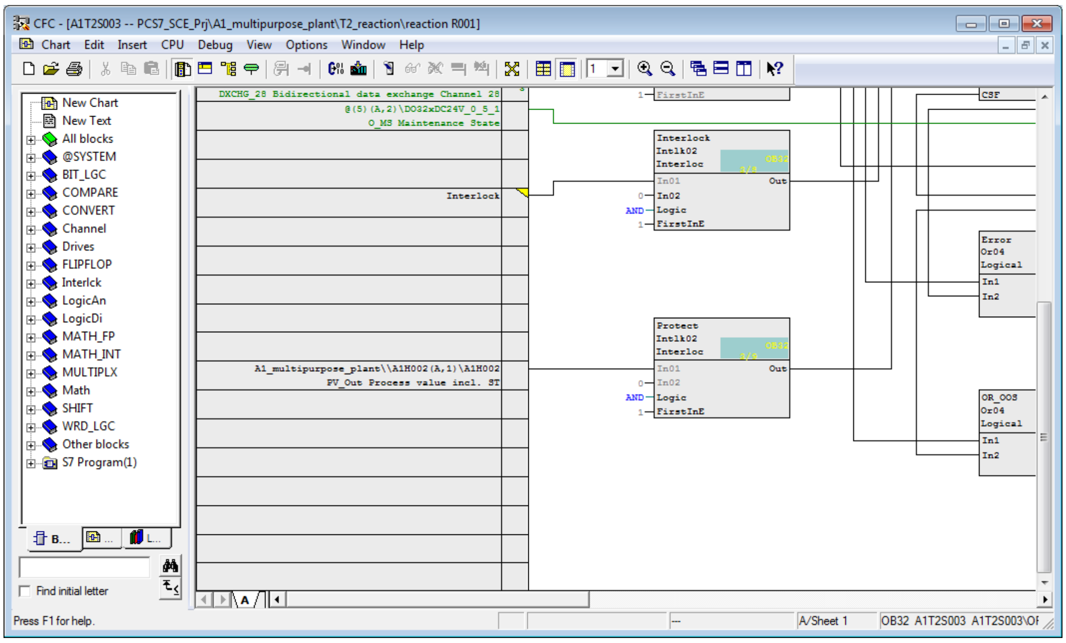
    (→ SIMATIC Manager → Plant View → A1T2S003)



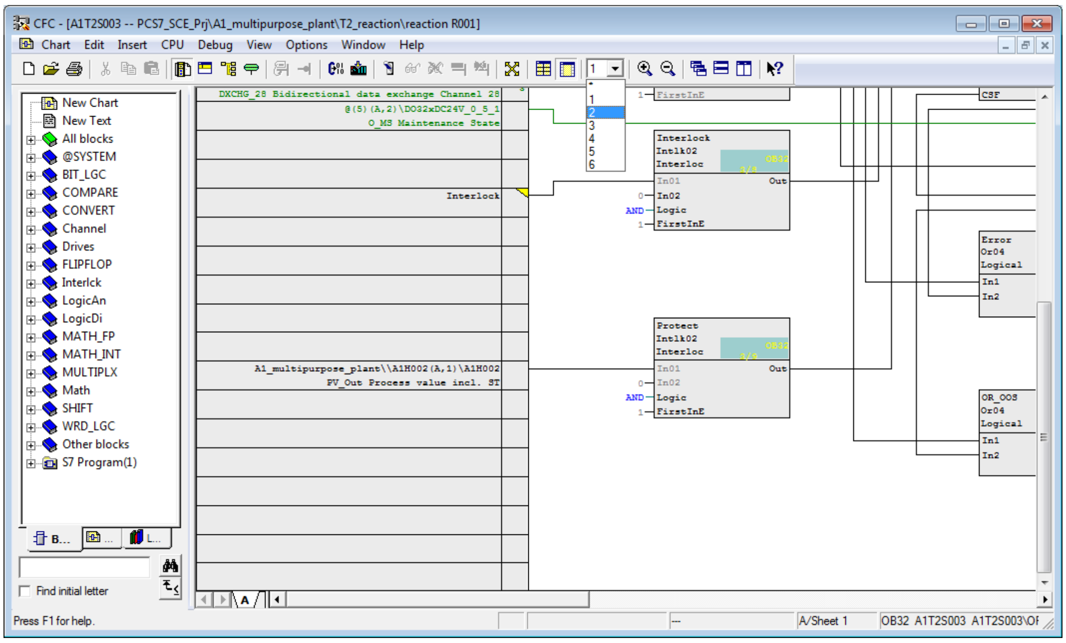
24. In addition to the individual drive function for the motor ‘MotL‘, the template Motor_Lean contains more blocks. Block ‘Intlk02‘ for locking ‘MotL‘ exists three times. The first block is called ‘Permit‘ and permits starting the motor only when the conditions are met. Here, we connect the signal of the main switch for plant A1H001. To this end, we first delete the connection to the placeholder ‘Permit‘ in the left margin. Then we establish a cross-chart connection. The result is shown below.


CFC - [A1T2S003 -- PCS7_SCE_Prj\A1_multipurpose_plant\T2_reaction\reaction R001]
Chart Edit Insert CPU Debug View Options Window Help
New Chart
New Text
All blocks
@SYSTEM
BIT_LGC
COMPARE
CONVERT
Channel
Drives
FLIPFLOP
Interlck
LogicAn
LogicDi
MATH_FP
MATH_INT
MULTIPLX
Math
SHIFT
WRD_LGC
Other blocks
S7 Program(1)
Find initial letter
A1_multipurpose_plant\\A1H003(A,1)\A1H003
PV_Out Process value incl. ST
"A1.T2.A1T2S003.SO+.O+" I1.3
pump outlet reactor R001 feedback running
FbkRun_A1T2S003
Pcs7DiIn
Digital
Permit
Intlk02
Interloc
Interlock
pump A1T2
MotL
Pump out
A1_multipurpose_plant\\A1H001(A,1)\A1H001
PV_Out Process value incl. ST
Press F1 for help.
A/Sheet 1
OB32 A1T2S003 A1T2S003\OF

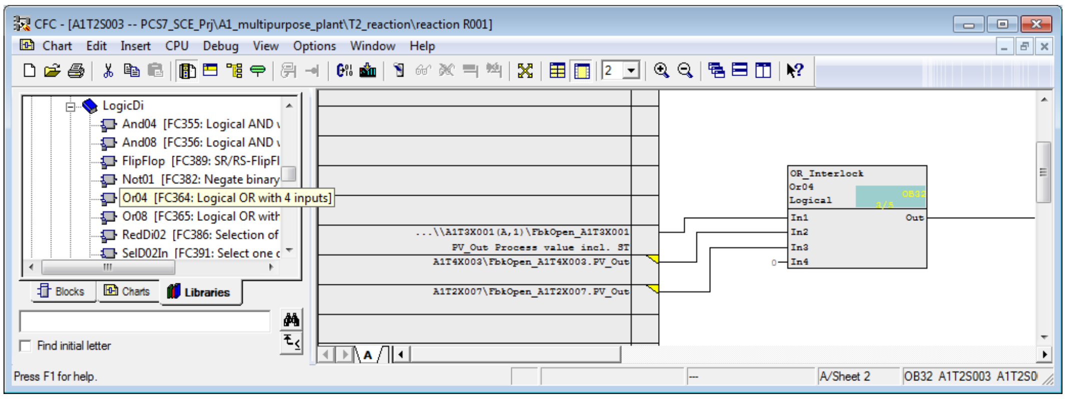
25. Now, we do the same for the block ‘Protect‘. ‘Protect‘ is used for connecting interlocks that require an acknowledgement for the motor to be enabled again. Here you connect EMERGENCY STOP.



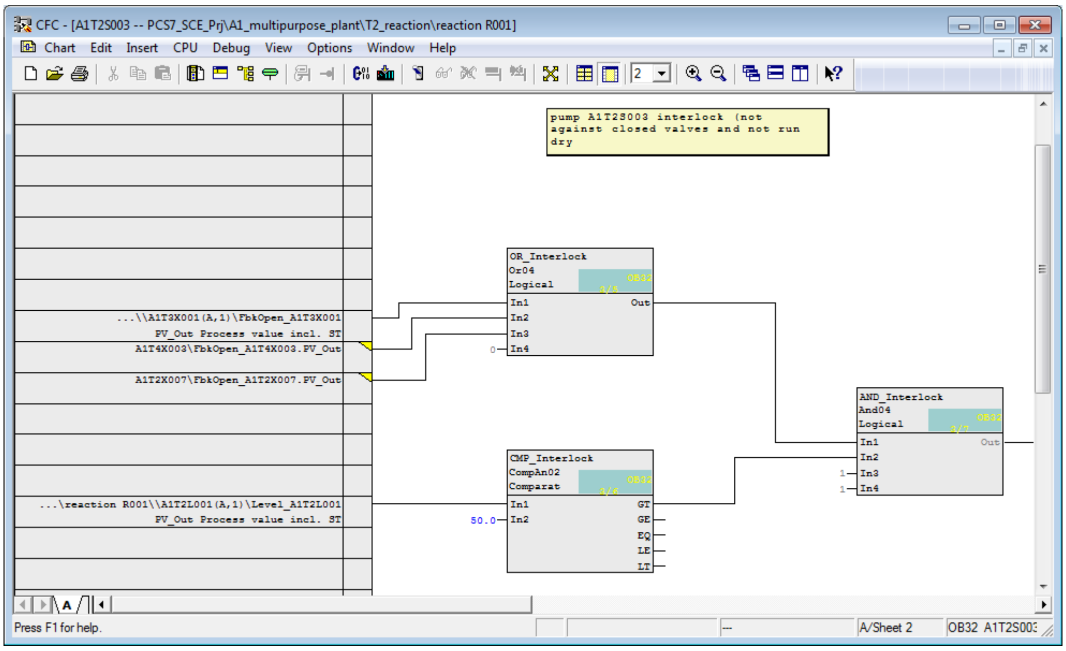
26. The ‘Interlock‘ block is intended for general interlock conditions. Here, you will implement the conditions from the Task (for example, at least one valve open). Because more than two conditions exist in this example, they have to be combined prior to being interconnected. To do this, we first have to go to Sheet 2 of the CFC.


CFC - [A1T2S003 -- PCS7_SCE_Prj\A1_multipurpose_plant\T2_reaction\reaction R001]
Chart Edit Insert CPU Debug View Options Window Help
1
2
3
4
5
6
Interlock
Protect
Error
OR_OOS
Press F1 for help.
A/Sheet 1

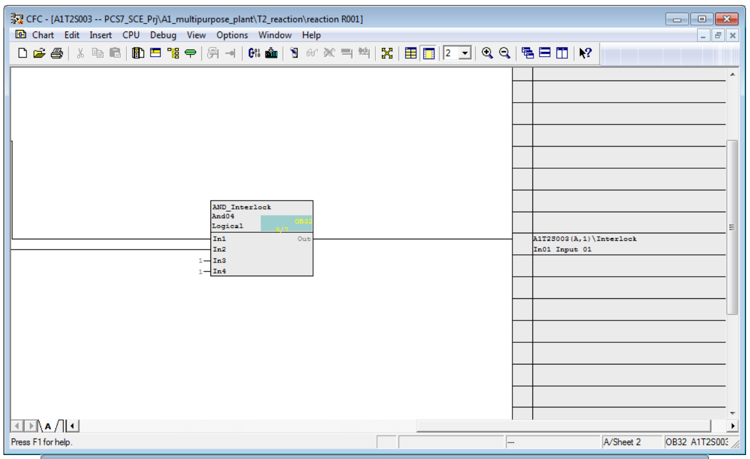
27. On the new sheet, we now insert an ‘Or04‘ block from the library. The feedback signals (FbkOpen) of the valves now have to be connected to this block. Because only 1 of the 3 valves has been set up so far, you can set up placeholders for these interconnections by means of textual interconnections. When compiled, they issue warnings but the program works anyhow.

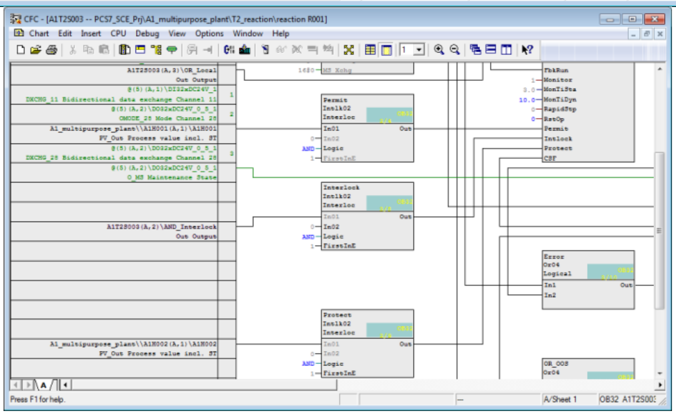

CFC - [A1T2S003 -- PCS7_SCE_Prj\A1_multipurpose_plant\T2_reaction\reaction R001]
Chart Edit Insert CPU Debug View Options Window Help
LogicDi
And04 [FC355: Logical AND
And08 [FC356: Logical AND
FlipFlop [FC389: SR/RS-FlipFl
Not01 [FC382: Negate binary
Or04 [FC364: Logical OR with 4 inputs]
Or08 [FC365: Logical OR with
RedDi02 [FC386: Selection of
SelD02In [FC391: Select one c
Blocks Charts Libraries
Find initial letter
...\\A1T3X001(A,1)\FbkOpen_A1T3X001
PV_Out Process value incl. ST
A1T4X003\FbkOpen_A1T4X003.PV_Out
A1T2X007\FbkOpen_A1T2X007.PV_Out
OR_Interlock
Or04
Logical
OB32
In1
In2
In3
In4
Out
Press F1 for help.
A/Sheet 2
OB32 A1T2S003 A1T2S0

28. Now, the minimum level has to be polled using the block ‘CompAn02‘, and the connections can be linked with ‘And04‘. The result looks like this.


CFC - [A1T2S003 -- PCS7_SCE_Prj\A1_multipurpose_plant\T2_reaction\reaction R001]
Chart Edit Insert CPU Debug View Options Window Help
pump A1T2S003 interlock (not against closed valves and not run dry
OR_Interlock
Or04
Logical
OB32
In1
In2
In3
In4
Out
...\\A1T3X001(A,1)\FbkOpen_A1T3X001
PV_Out Process value incl. ST
A1T4X003\FbkOpen_A1T4X003.PV_Out
A1T2X007\FbkOpen_A1T2X007.PV_Out
AND_Interlock
And04
Logical
OB32
CMP_Interlock
CompAn02
Comparat
OB32
...\reaction R001\\A1T2L001(A,1)\Level_A1T2L001
PV_Out Process value incl. ST
50.0
GT
GE
EQ
LE
LT
Press F1 for help.
A/Sheet 2
OB32 A1T2S003

29. Next, output ‘Out‘ of block ‘And04‘ has to be connected to input ‘In01‘ of the ‘Interlock‘ block.




CFC - [A1T2S003 -- PCS7_SCE_Prj\A1_multipurpose_plant\T2_reaction\reaction R001]
Permit
Intlk02
Interloc
Interlock
Protect
Error
Or04
Logical
A1T2S003(A,2)\AND_Interlock
Out Output
A1_multipurpose_plant\\A1H001(A,1)\A1H001
PV_Out Process value incl. ST
A1_multipurpose_plant\\A1H002(A,1)\A1H002
PV_Out Process value incl. ST
Press F1 for help.
A/Sheet 1
OB32 A1T2S003

30. Below, an overview of all new interconnections in chart ‘A1T2S003’ is provided once more.

Table 4: Input Interconnections in Chart ‘A1T2S003/Sheet1’

Input:	**Interconnected with:**	**Inverted**
Intlk02.Permit.In01	A1H001(A,1) / A1H001 PV_Out Process value incl. ST	No
Intlk02.Protect.In01	A1H002(A,1) / A1H002 PV_Out Process value incl. ST	No

Table 5: New Blocks in Chart ‘A1T2S003/Sheet2’

Block:	**Catalog/Folder:**
Or04/Or function with 4 inputs	Libraries/PCS7 APL V81/ Blocks+Templates\Blocks/LogicDi
And04/And function with 4 inputs	Libraries/PCS7 APL V81/ Blocks+Templates\Blocks/LogicDi
CompAn02/Comparison of analog values	Blocks / LogicAn

Table 6: Input Interconnections in Chart ‘A1T2S003/Sheet2’

Input:	Interconnected with:	Inverted
Or04.In1	A1T3X001(A,1) / FbkOpen PV_Out Process value incl. ST	No
Or04.In2	A1T4X003\FbkOpen_A1T4X003 (textual interconnection.)	No
Or04.In3	A1T2X007\FbkOpen_A1T2X007 (textual interconnection)	No
CompAn02.In1	A1T2L001(A,1) / Level_A1T2L001 PV_Out Process value incl. ST	
CompAn02.In2	Value: Value=50.0 Comment=Minimum level	

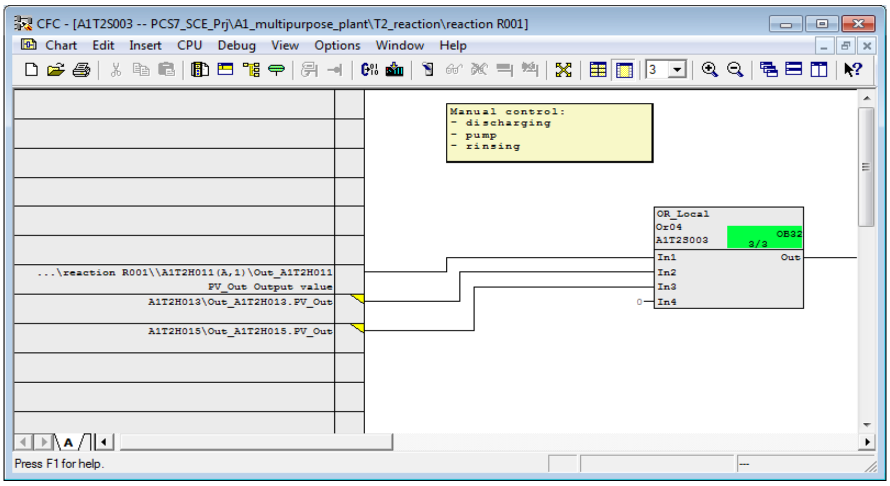
Table 7: Block Interconnections in Chart ‘A1T2S003/Sheet2’

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

Table 8: Block Interconnections between ‘A1T2S003/Sheet1‘ and ‘A1T2S003/Sheet2’

Input	Output	Inverted
Intlk02.Interlock.In01	And04.Out	No

31. In Sheet 1 of chart ‘A1T2S003’ we now set up interconnections for manual operation with A1T2H011 (to drain reactor R001). Because other manual operations can also access the pump, an ‘Or04‘ is set up on Sheet3.


CFC - [A1T2S003 -- PCS7_SCE_Prj\A1_multipurpose_plant\T2_reaction\reaction R001]

```
Manual control:
- discharging
- pump
- rinsing
```

...\reaction R001\\A1T2H011(A,1)\Out_A1T2H011 PV_Out Output value
A1T2H013\Out_A1T2H013.PV_Out
A1T2H015\Out_A1T2H015.PV_Out

OR_Local
Or04
A1T2S003
OB32
3/3
In1
In2
In3
In4
Out

Press F1 for help.

Table 9: New Blocks in Chart ‘A1T2S003/Sheet3’

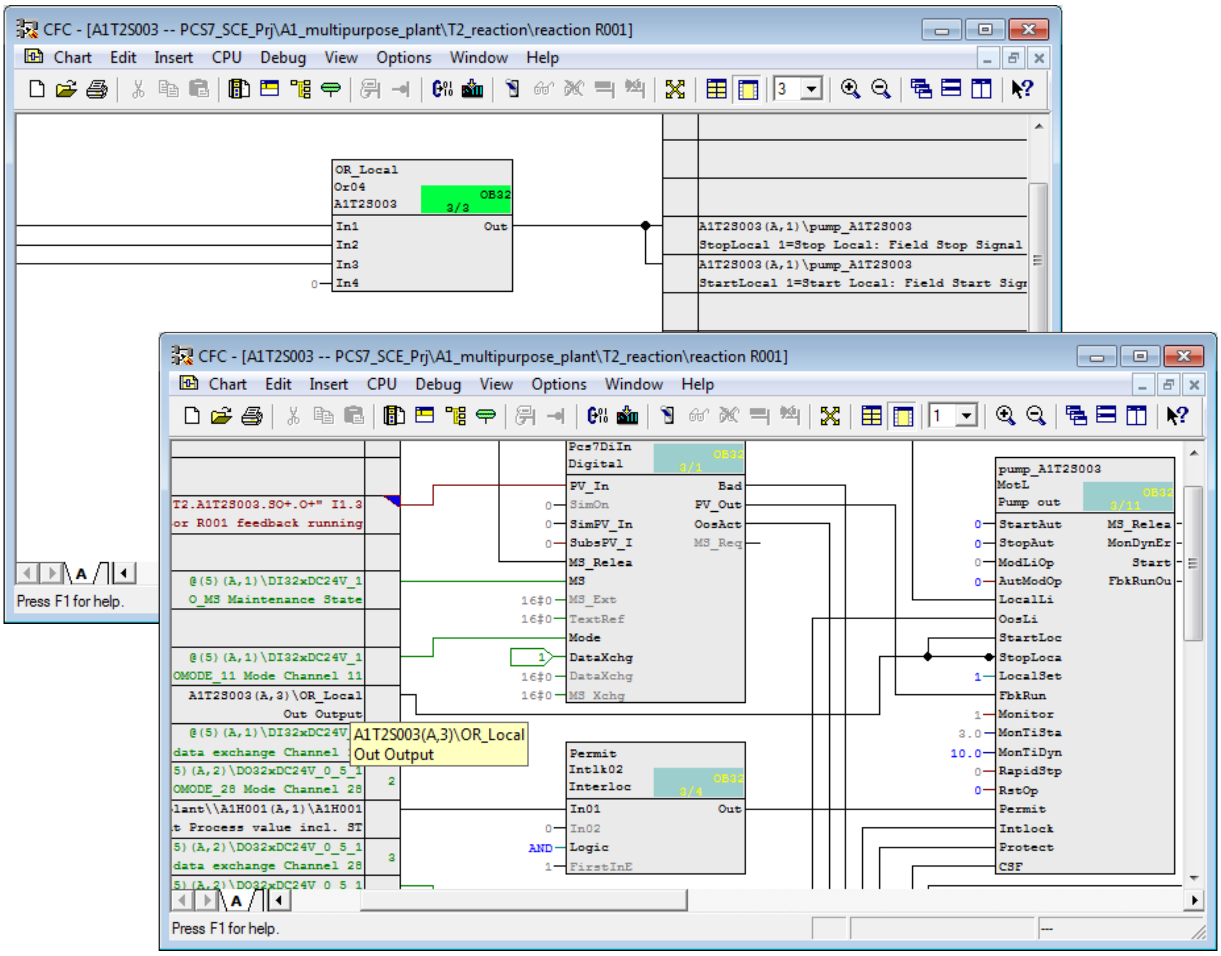
Block:	Catalog/Folder:
Or04 / Or-function with 4 inputs	Blocks/LogicDi

Table 10: Input Interconnections in Chart ‘A1T2S003/Sheet3’

Input:	Interconnection to:	Inverted
A1T2H011(A,1) / Out_A1T2H011 PV_Out Process value incl. ST	Or04.Or_Local.In1	No
A1T2H013\Out_A1T2H013.PV_Out	Or04.Or_Local.In2	No
A1T2H015\Out_A1T2H015.PV_Out	Or04.Or_Local.In3	No

Table 11: Block Interconnections between Chart ‘A1T2S003/Sheet1‘ and ‘A1T2S003/Sheet 3’

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



32. For the manual local Start and Stop to be effective, we have to switch to local operation.

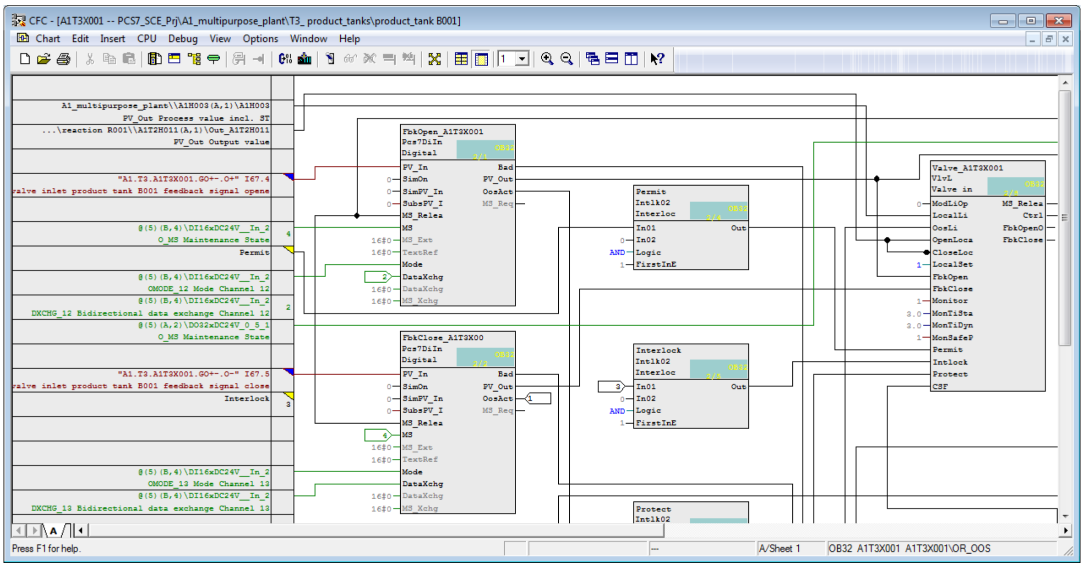
Table 12: Additional Input Interconnections in Chart ‘A1T2S003/Sheet1’

Input:	Interconnection to:	Inverted
MotL.Pump_A1T2S003. LocalLi	A1H003(A,1) / A1H003 PV_Out Process value incl. ST	No
MotL.Pump_A1T2S003. LocalSetting	1 (connection not visible → double click on block and open connections → Change value)	

33. 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. Below, the interconnections that were set up in chart A1T3X001 additionally for manual operation.

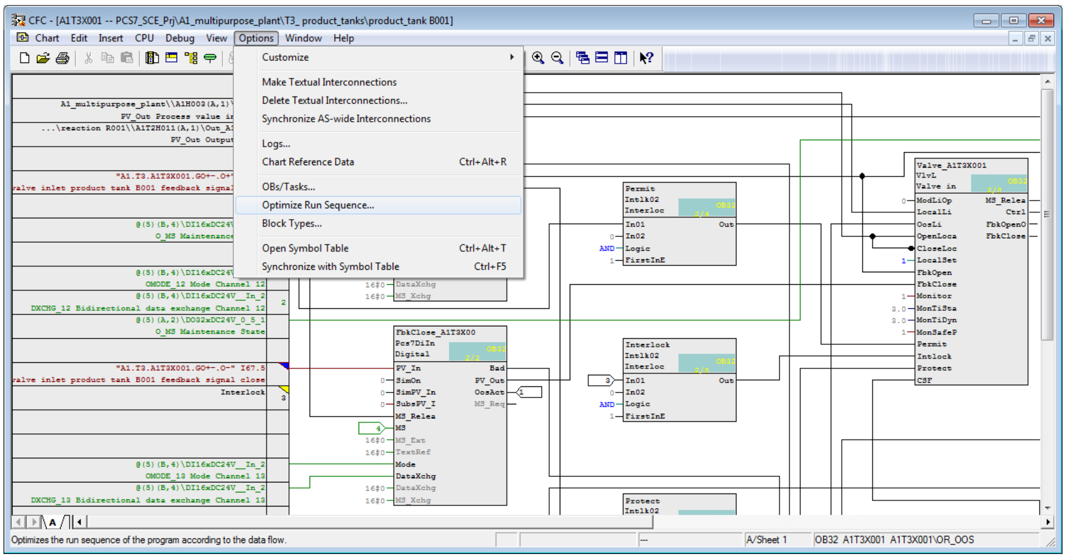
Table 13: Input Interconnections in Chart 'A1T3X001/Sheet1'

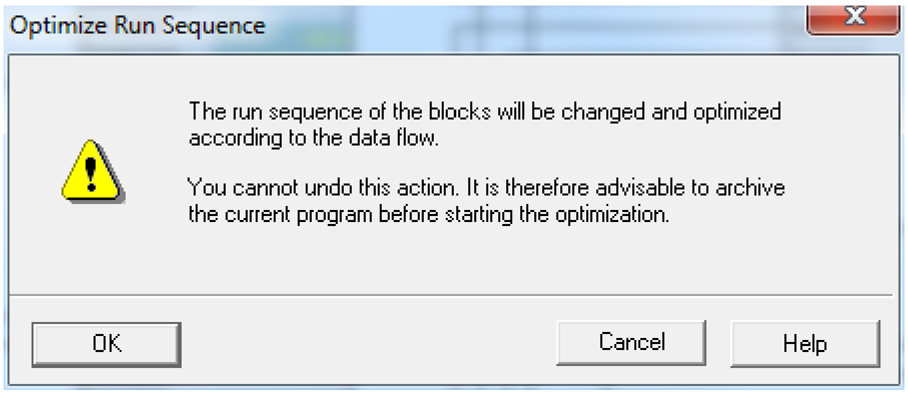
Input:	Interconnection to:	Inverted
VlvL.Valve_A1T3X001.LocalLi	A1H003(A,1) / A1H003 PV_Out Process value incl. ST	No
VlvL.Valve_A1T3X001.OpenLocal	A1T2H011(A,1) / Out_A1T2H011 PV_Out Process value incl. ST	No
VlvL.Valve_A1T3X001.CloseLocal	A1T2H011(A,1) / Out_A1T2H011 PV_Out Process value incl. ST	Yes
VlvL.Valve_A1T3X001.LocalSetting	1	


CFC - [A1T3X001 -- PCS7_SCE_Prj\A1_multipurpose_plant\T3_ product_tanks\product_tank B001]
Chart Edit Insert CPU Debug View Options Window Help
FbkOpen_A1T3X001
FbkClose_A1T3X00
Permit
Interlock
Protect
Valve_A1T3X001
Press F1 for help.
A/Sheet 1
OB32 A1T3X001 A1T3X001\OR_OOS

34. All blocks used in the charts are integrated in the run sequence of the entire program when inserted. By clicking on the symbol the run sequence can be displayed. To improve the data flow within the overall program, we recommend you optimize the sequence after the program is generated. This is done in the menu of the CFC editor under Options -> Optimize Run Sequence (independant from the current view.)

    (→ Extras → Optimize Run Sequence → OK)


CFC - [A1T3X001 -- PCS7_SCE_Prj\A1_multipurpose_plant\T3_product_tanks\product_tank B001]
Chart Edit Insert CPU Debug View Options Window Help
Customize
Make Textual Interconnections
Delete Textual Interconnections...
Synchronize AS-wide Interconnections
Logs...
Chart Reference Data Ctrl+Alt+R
OBs/Tasks...
Optimize Run Sequence...
Block Types...
Open Symbol Table Ctrl+Alt+T
Synchronize with Symbol Table Ctrl+F5
Optimizes the run sequence of the program according to the data flow.
A/Sheet 1
OB32 A1T3X001 A1T3X001\OR_OOS


Optimize Run Sequence

The run sequence of the blocks will be changed and optimized according to the data flow.

You cannot undo this action. It is therefore advisable to archive the current program before starting the optimization.

Note: A block should be selected beforehand. Otherwise you will end up in the general overview without any clue as to where the blocks are called.

EXERCISES

In the exercises, we apply what we have learned from theory and from the step by step instructions. The multi-project from the step by step instructions (PCS7_SCE_0105_R1503_en.zip) is to be used and expanded.

In the following exercises, you can design and implement your own additional interlocks. Keep in mind that for interlocking valves, only the EMERGENCY STOP switch, and the main switch as well as the level of the respective tank (or levels of the respective tanks) are needed.

TASKS

1. Complete the interlocks for the valve that already exists:

– A1T3X001

2. Set up the CFC for the level of educt tank B001:

– A1T1L001

3. Set up the CFCs for the following valves, including interlocks:

– A1T1X004

– A1T2X001

4. Set up the CFC for the following pump including interlocks:

– A1T1S001

5. Set up 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.