DDC Suite 2.7

Functions

SAIA BURGESS CONTROLS

DDC Suite 2.7 / Functions

Syntax and instructions for actions in the workshop

Please follow the instructions of the trainer.

Please

- use the same symbol names
- use the same group names
- place the FBoxes as precisely as possible at the same position
- do not work faster or differently, even if you are an experienced programmer

This workshop will show you some basic mechanisms, structured working methods and structured symbol organisation. Don't worry, you don't have to

- learn all FBoxes during the workshop
- be familiar with the programming of building automation applications
- be an "old hand" in programming

Once you have learned the mechanisms and philosophy, you will see the advantages that working with the DDC Suite can offer you as SI.

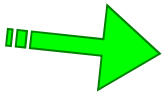
DDC Suite 2.7 / Functions

Syntax and instructions for actions in the workshop

Click here with the left mouse button

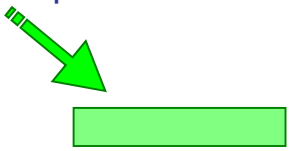
Double-click here with the left mouse button

Click here with the right mouse button



Follow the green arrow to the next step

Example



Enter the blue text in the green text box

Pay attention to the yellow area

Changes/different work sequence to the previous versions

Functions

SAIA BURGESS CONTROLS

DDC Suite 2.7 / Functions

This handbook describes the functions of DDC Suite beyond the normal control and regulation functions.

In principle, all FBox families (e.g. binary, integer, counter, timer, HVAC and DDC Suite, etc.) can be used to program an application that is required for the control and regulation functions of the system. Beyond this, however, additional functions are required, such as:

- Fault indicator light should flash again after a newly triggered alarm
- Grouping alarms
- Alarm list in a web server
- Collecting trend data
- Connection to a BACnet BMS
- Naming of data points according to planner/end customer specifications
- …

These are mostly functions that increase customer comfort when operating and monitoring. Although they are not strictly necessary for controlling the system, they are often requested. To reduce workload, DDC Suite 2.7 offers many of these functions as an integrated component of the FBoxes.

DDC Suite 2.7 / Functions

Section

Tip: Always start with a project template from DDC Suite 2.7. These contain many pre-configured basic settings and FBoxes that you would need to use anyway. This handbook explains the precise mechanisms used by the FBoxes.

1. Alarm logging

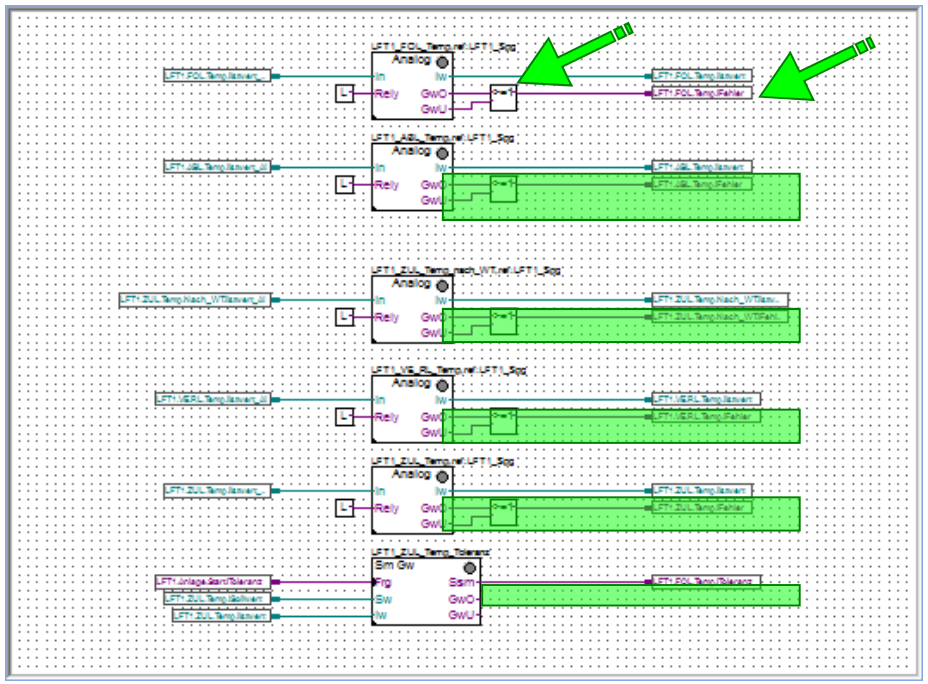
SAIA BURGESS CONTROLS

DDC Suite 2.7 / Functions

Alarm logging – Conventional

Many FBoxes provide alarm statuses at one or multiple outputs. The analogue FBox often also uses an Or-FBox. This groups the two limit alarms of an FBox.

A symbol must also be defined. This symbol is then used to further group alarms. This is required for all alarms that are intended to trigger a further reaction (system off, signalisation).



DDC Suite 2.7 / Functions

Alarm logging – Conventional

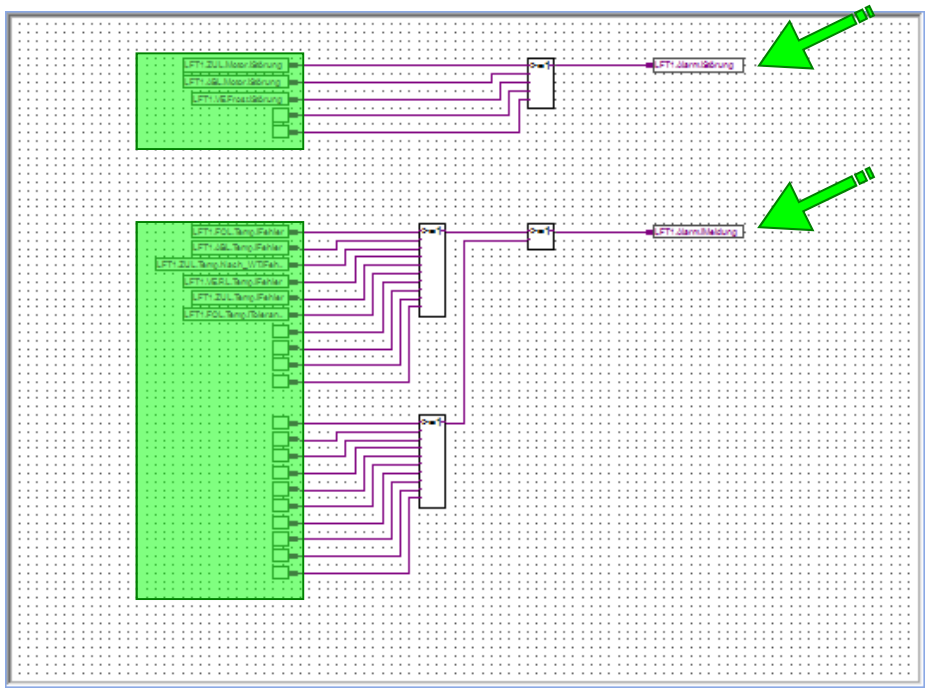
These alarms are usually compiled in 2 or 3 groups. This is usually as follows:

-Fault → Switches off the system immediately

-Report → Secondary faults, such as pumps in alarm

-Informative → Maintenance or filter, no real faults

This requires at least one additional page in Fupla with Or-FBoxes.



DDC Suite 2.7 / Functions

Alarm logging – Conventional

This grouping of alarms is standard practice. However, remember at all times to do the following:

-Search for all alarms in the program and incorporate them in the respective alarm chain

-Integrate new alarms in the alarm grouping

-Remove alarms that have been omitted from the alarm grouping

It is easy to make a mistake here, as symbols can be forgotten when grouping the alarms. Often, the alarm grouping is not updated (symbols expanded or removed) when adding or removing alarm FBoxes.

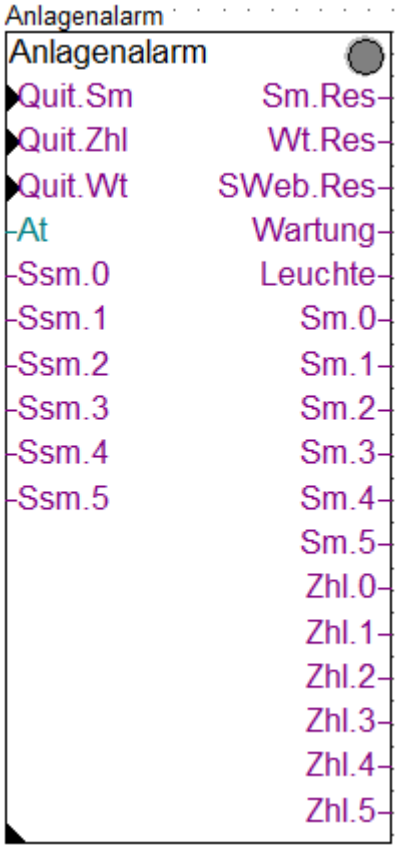
If the signalling (lights) is to flash for a new alarm, additional Or chains with Flanken FBoxes (flank) are required.

DDC Suite 2.7 / Functions

Alarm logging – Automated

The FBox Anlagenalarm (system alarm) provides all of these functions.

-Grouping of all alarms in the program
-Allocation in up to 5 groups
-Determination of the number of pending and newly triggered alarms
-Signalling taking new alarms into account
-Acknowledgement function via button, virtual button (web, BMS), valid for all alarms, for every group or for each individual alarm (web)
-Incorporation of higher-level alarms

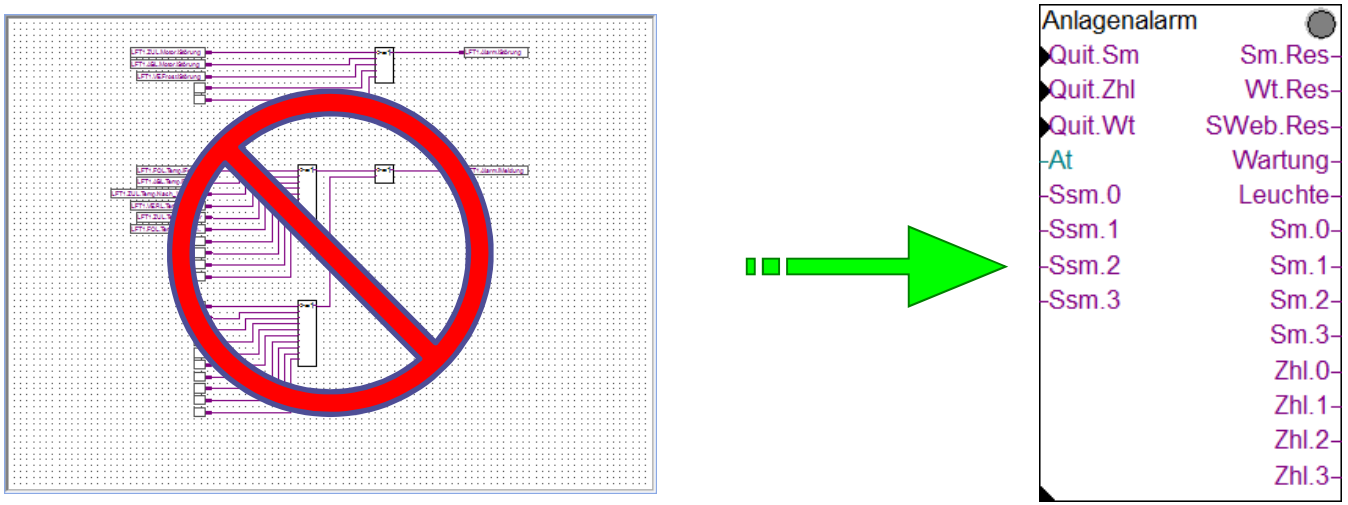


DDC Suite 2.7 / Functions

Alarm logging – Automated

This FBox therefore replaces the Fupla page(s) with the grouping of alarms, and provides additional functions.

This completely removes the need to link the alarms with additional Or-FBoxes, lines and symbols, making programming much simpler.



DDC Suite 2.7 / Functions

Alarm logging – Automated

In the setting parameters, you can enter a designation for each of the 5 alarm groups. This keeps things clear and helps you understand which group is used for which purpose.

In this case, the groups are used as follows:

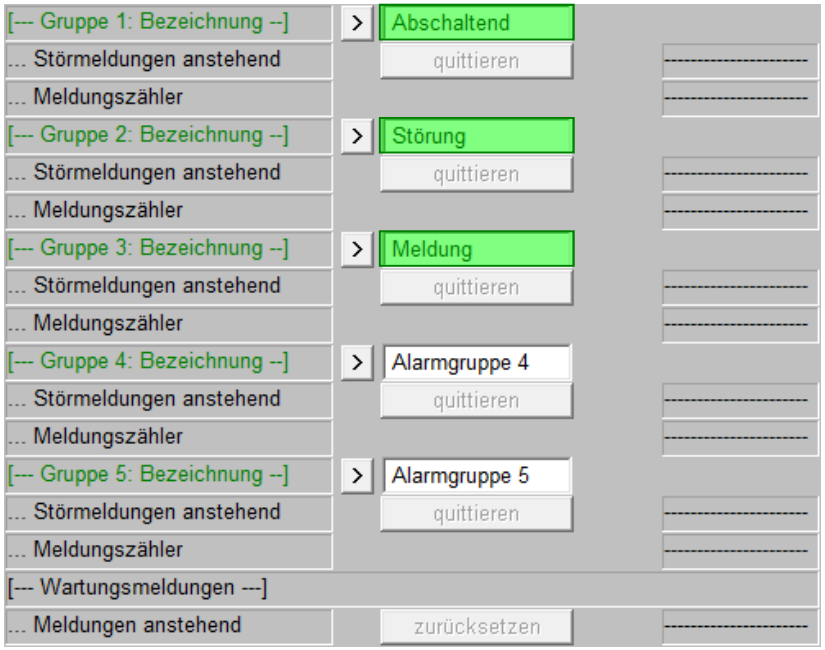
-1 → All alarms in this group should lead to the system being switched off

-2 → All alarms in this group are secondary and should only be logged

-3 → All alarms in this group are only reports and should only be locked

-4/5 → These groups are not used and were only given a designation

As you can see, maintenance notifications are also compiled in this FBox. Maintenance messages always accumulate in this group.



DDC Suite 2.7 / Functions

Alarm logging – Automated

The FBox provides 2 outputs for each alarm group:

-**Sm.1** to **5** → High, if at least one alarm in this group is still pending

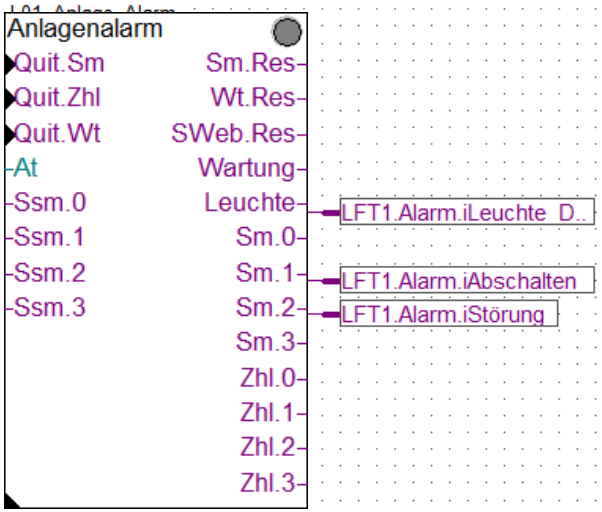
-**Zhl.1** to **5** → High, if at least one alarm in this group has re-occurred. New here means that it has occurred since the last time the new alarm notification was acknowledged

The output **Leuchte** (light) can be used directly to control a signalisation. In the basic setting, this output only represents alarm group 1:

-Low → No alarm pending, no new alarm since last acknowledgement of the new alarm notification

-Flashing → New alarm detected

-High → New alarm notification acknowledged and at least one pending alarm



Info: The new alarm notification can be acknowledged via the input **Quit.Zhl**

DDC Suite 2.7 / Functions

Alarm logging – Automated

Parts of the BACnet alarm management system were used to allocate which alarm should be incorporated in which alarm group Don't worry – this also works without BACnet, but using it can make it more flexible. If BACnet is required, you can therefore use 2 functions at once.

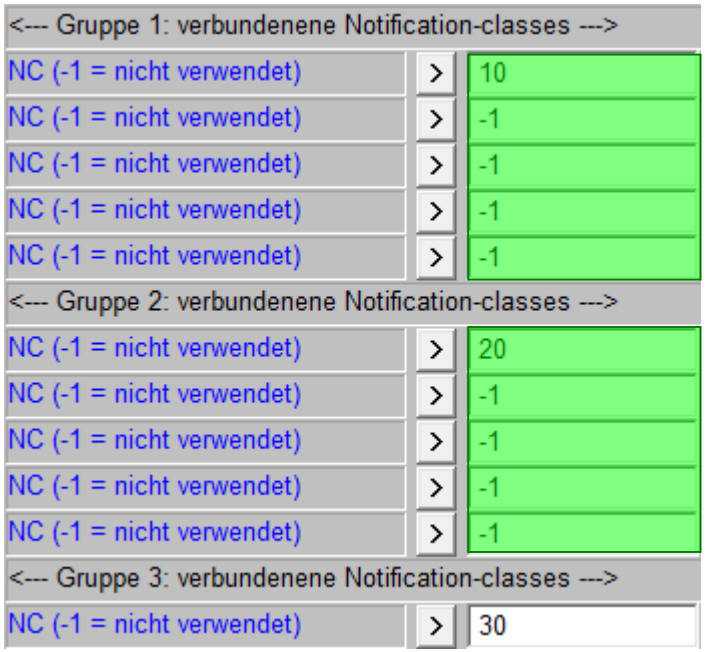
BACnet also contains alarm groups, which are known as notification classes (abb.: NC). Theoretically there are up to 4,194,303 alarm groups, although these are often used only for finer subdivision, and only in a limited number (less than 20).

In our application, 3 alarm groups are usually sufficient.

-All alarms that should be collected in alarm group 1 must be parametrised for NC 10

-Alarms for alarm group 2 must be parametrised for NC 20

-Alarms for alarm group 3 must be parametrised for NC 30



DDC Suite 2.7 / Functions

Alarm logging – Automated

Why is it possible to define multiple NCs for a single alarm group?

It is possible that the customer requires a finer subdivision of alarms in BACnet, e.g.:

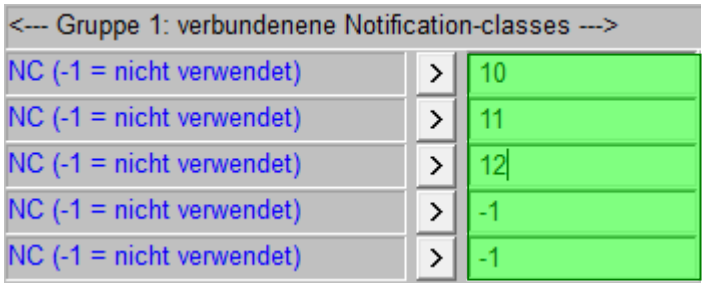
-NC 10, deactivating, only thermal alarms (motor protection, PTC resistor, etc.)

-NC 11, deactivating, feedback faults

-NC 12, deactivating, manual interventions

This is certainly helpful for the operator station of the BMS, but is less important in our application. Deactivating is deactivating. Alarms with NC 10, 11 and 12 should therefore be grouped in alarm group 1.

NC 10 is pre-set for alarm group 1, NC 20 for alarm group 2 and NC 30 for alarm group 3. Alarm groups 4 and 5 are inactive, as they both contain NC -1. Only use the value -1 when an entry is not used.

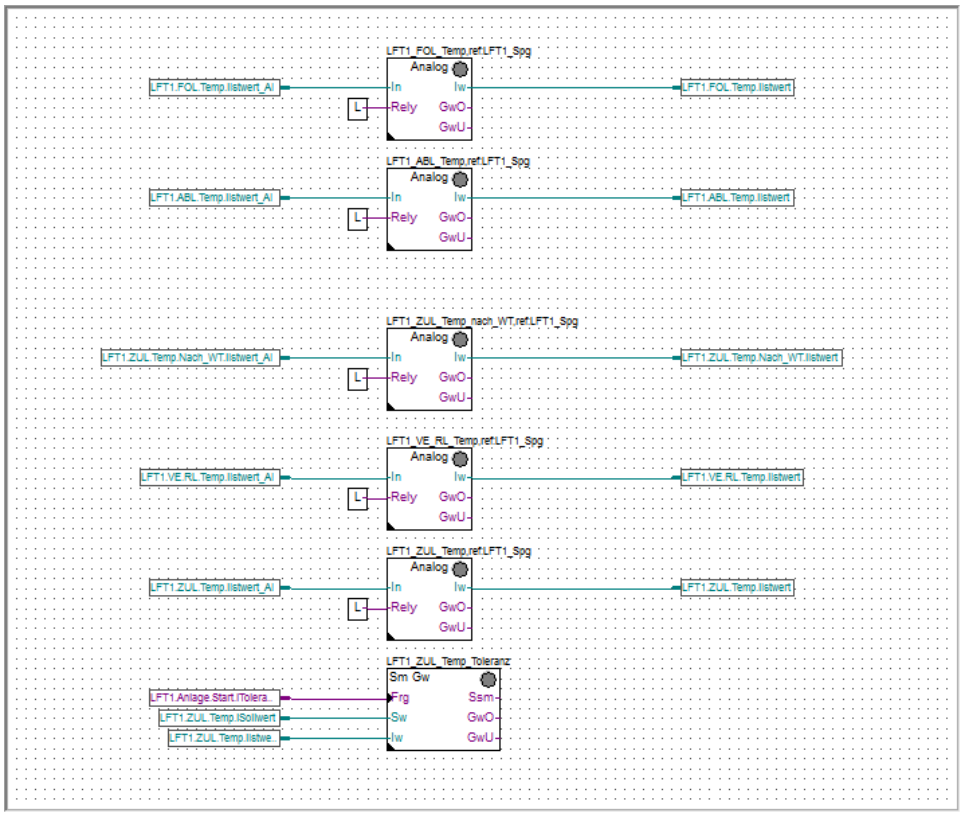


DDC Suite 2.7 / Functions

Alarm logging – Automated

If an FBox with alarm is now used, there is no need to connect anything at the output of the FBox.

This represents a simplification compared to "conventional" alarm chains. The alarms are automatically included in the allocated alarm group.



DDC Suite 2.7 / Functions

Alarm logging – Automated

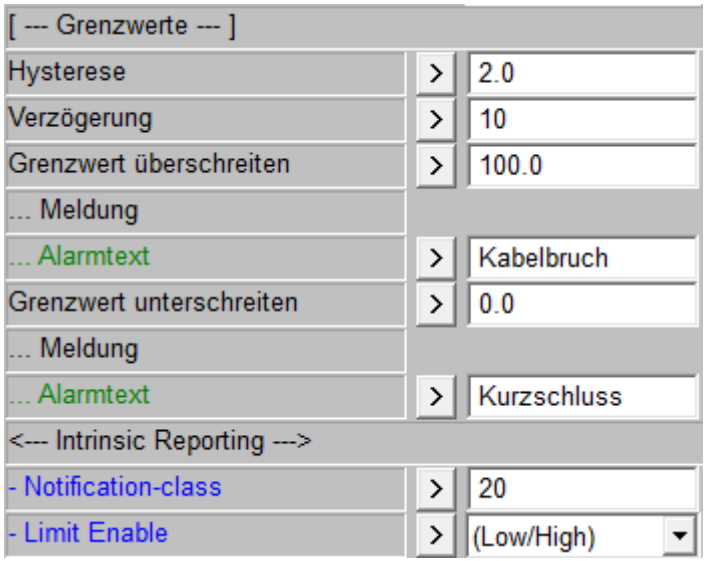
NC 10, 20 and 30 are pre-set in the FBoxes with alarm. This can be left alone in most cases.

In the measurement value FBox of the analogue value family, for example, NC 20 is pre-set. This means the limit value notifications are registered in alarm group 2.

FBoxes have many parameters. What is the quickest way to find out whether an FBox supports alarms, and where must the NC be entered?

Each FBox has in its parameters a group called **[--- Systemfunktionen ---]** (system functions). If this group contains the parameter **PCD Alarmverwaltung (Index)** (PCD alarm management), the FBox is capable of registering its alarms automatically in the required alarm group.

The set value is irrelevant for the alarm grouping. We will return to these parameters later on.



DDC Suite 2.7 / Functions

Alarm logging – Automated

If the FBox supports alarms, the allocation of the NC may need to be adjusted. Each alarm is included in a group **[--- Group designation ---]**.

In the image to the right, we see the limit value floating FBox from the faults family. The simplest way to find the alarms is to

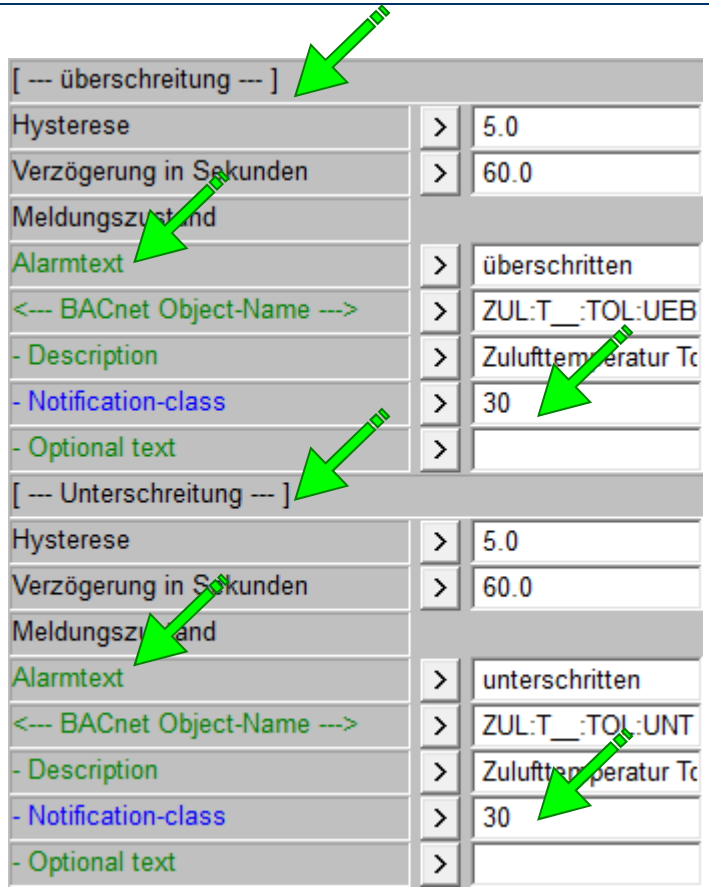
-search for a group [-- … ---], then

-for the property **Alarm text** (in green text since PG5 2.2), and then

-the property **– Notification class** (in blue text since PG5 2.2)

In most cases, each alarm has its own notification class. The measurement value FBox has only one notification class for both alarms (exceed and fall below). This then applies for both alarms.

If an alarm does NOT flow into the alarm group, enter it as notification class -1.



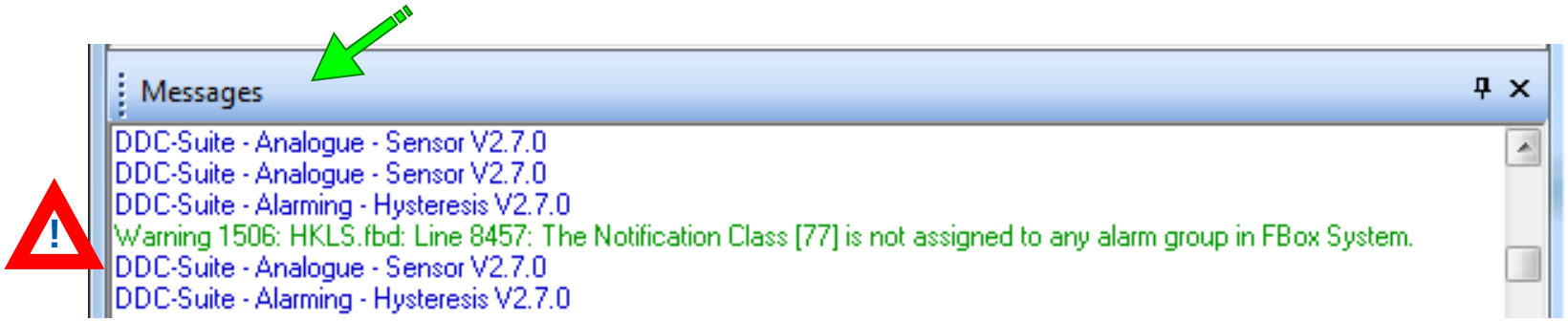
DDC Suite 2.7 / Functions

Alarm logging – Automated

The allocation is parametrised and the alarm grouping is ready.

It is easy to make typos when adjusting the parameter notification class, such as entering 21 instead of 12. Whilst the program is being compiled, check the FBox for whether the notification class has been allocated to an alarm group. For this reason, it is imperative that the system alarm FBox is always placed in front of the FBoxes with alarm.

If the NC of the alarm FBox cannot be defined in the system alarm FBox, an error message will occur. This indicates that an invalid allocation was performed.

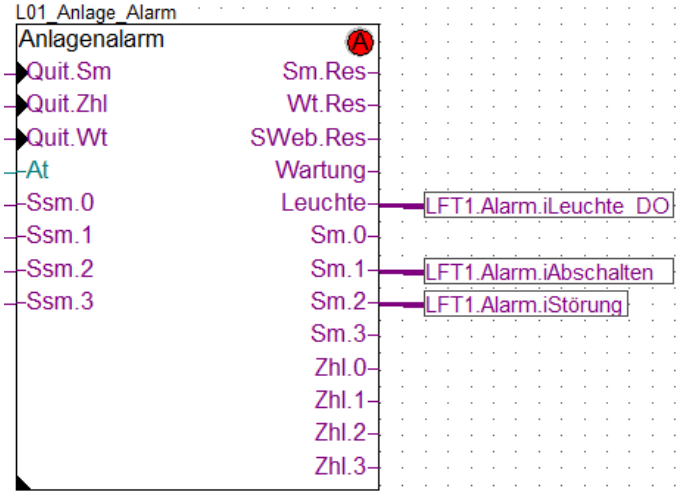


DDC Suite 2.7 / Functions

Alarm logging – Automated

During operation, the current alarm status of the individual groups can be seen on the FBox outputs.

The outputs must be wired to execute the required action. The system alarm FBox does not automatically switch a system off, but instead only collects alarm information!

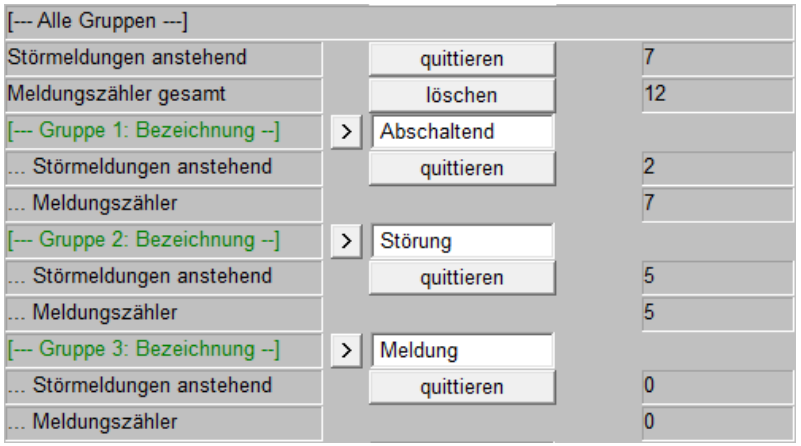


The following is visible in the parameters:

-The sum of all pending alarms in all groups

-The sum of all new alarms in all groups

-Pending/new alarms for each group



The alarms can be acknowledged via the FBox input **Quit.Sm**, the button **quittieren** (acknowledge), or by acknowledging via the **InitLib FBox**.

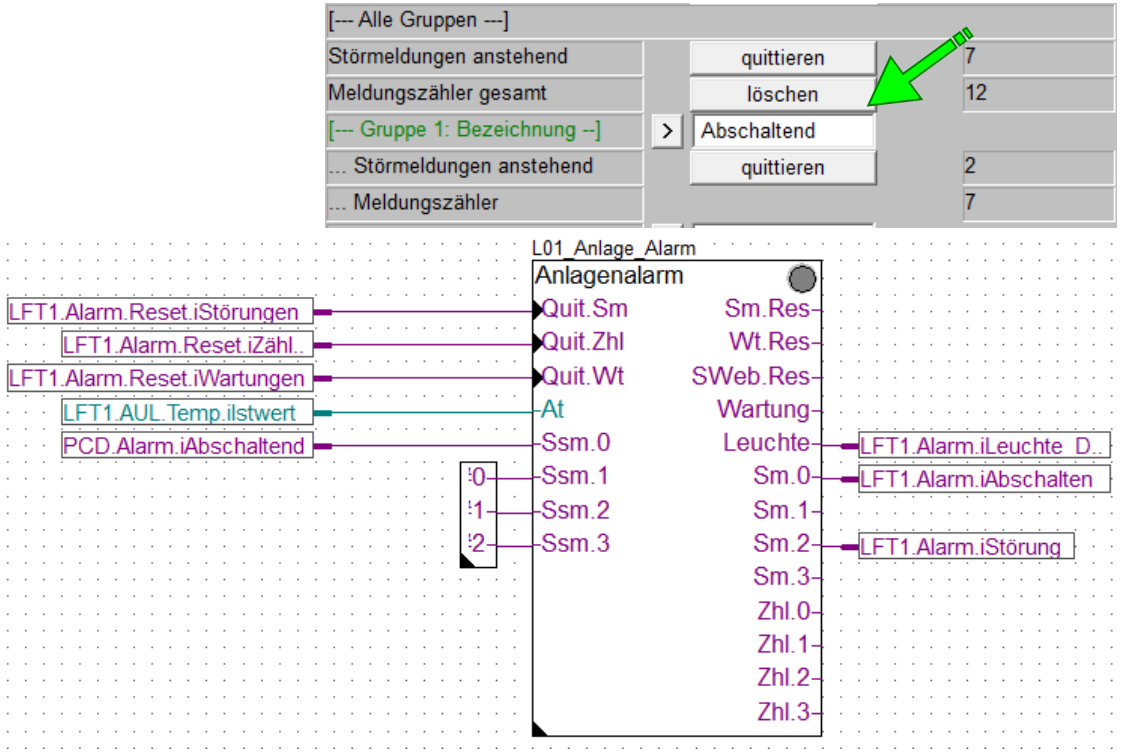
DDC Suite 2.7 / Functions

Alarm logging – Automated

The output **Leuchte** (light) will flash as long as at least one new alarm has been detected in group 1. The new alarm notification can occur via the **löschen** (delete) button or the FBox input Quit.Zhl. Both resets the new alarm notifications and counters of all groups!

Usually, there will be a button on the control cabinet for "Siren off" and "Acknowledge". In this case, these can be place directly at the two inputs **Quit.Zhl** and **Quit.Sm**.

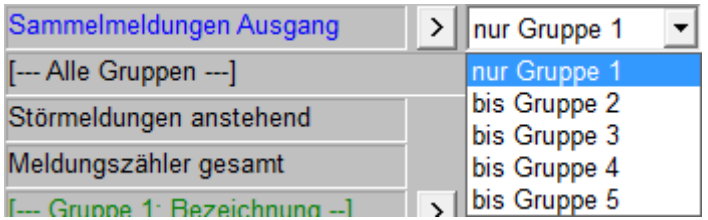
If only one acknowledgement button is available, you can activate the input **Quit.Zhl** with a short button press, and the input **Quit.Sm** with a long button press (e.g. longer than 1.5 seconds).



DDC Suite 2.7 / Functions

Alarm logging – Automated

It may be necessary to subdivide deactivating alarms into 2 or 3 groups in order to trigger additional reactions, such as:



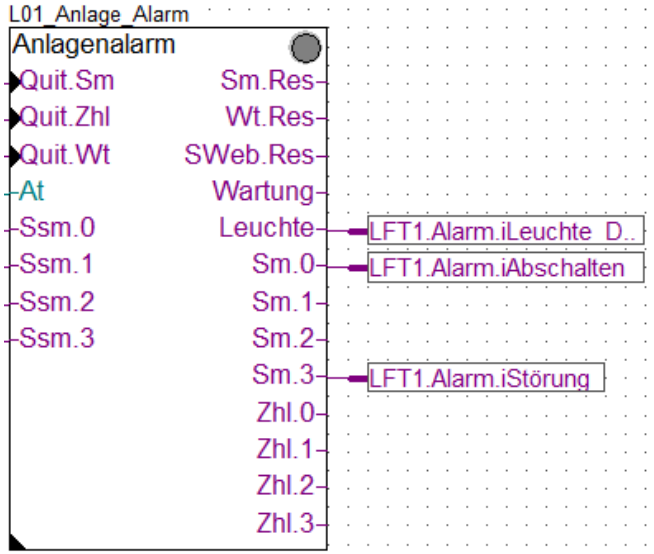
-Group 1 → Deactivating, thermal alarms such as motor protection, PTC thermistor, etc.

-Group 2 → Deactivating, manual interventions

-Group 3 →Secondary faults

-Group 4 → Filter notifications, tolerance monitoring

In this case, it is possible to define a grouping from group 1 to group 5 in the parameter **Sammelmeldung Ausgang** (collective message output).



Here, groups 1 and 2 need to be combined, as both have a deactivating effect. Combined groups are shown at the FBox outputs **Sm.0** and **Zhl.0**

.

Group 1 is included in the collective message as a default setting, i.e. **Sm.0** and **Sm.1** as well as **Zhl.0** and **Zhl.1** are identical.

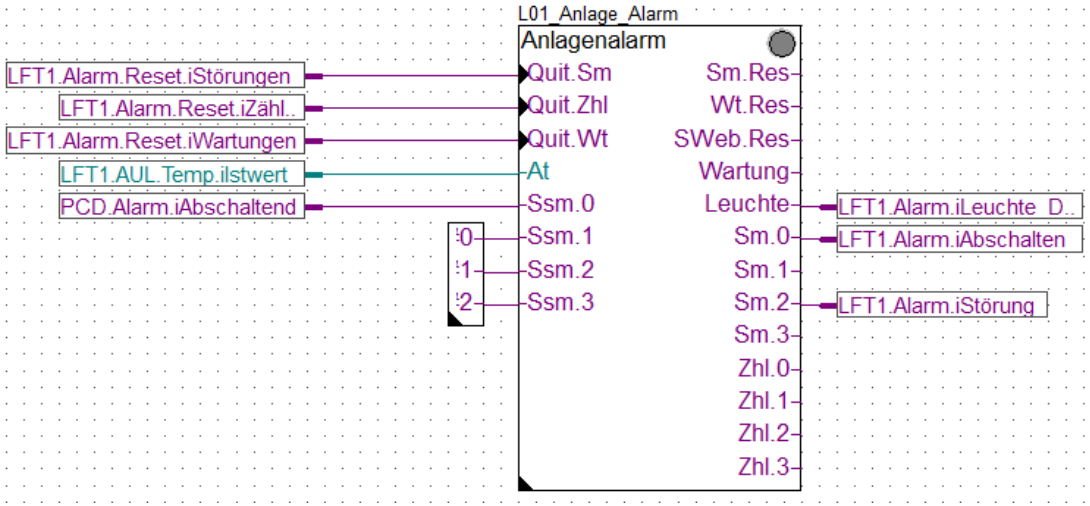
DDC Suite 2.7 / Functions

Alarm logging – Automated

If an FBox has a pending maintenance notification, this is displayed at the output **Wartung** (maintenance). You can reset maintenance and reset the FBoxes maintenance counters via the input **Quit.Wrt** or the **reset** button (zurücksetzen).

Connect the outdoor temperature at the input **At**. From this FBox, this replaces the value **At** of the FBox **InitLib**.

The inputs **Ssm.0** to **5** are forwarded to the outputs **Sm.0** to **5**. This makes it easy to integrate higher-level alarms, such as fire emergency stop.



DDC Suite 2.7 / Functions

Alarm logging – Automated

In most cases, the switching cabinet has only one combined fault signal light, or it is necessary to have a combination of the most important fault notifications from all systems.

The **Sammelalarm** FBox (combined alarm) can help here. The program may contain only one of these. It must also be placed in front of all **Anlagenalarm** FBoxes (system alarm).

The **Sammelalarm** FBox (combined alarm) detects the alarms present at the output **Sm.0**, **Zhl.0** and **Light** from all **Anlagenalarm** FBoxes (system alarm), and combines then at the output **Sm**, **Zhl** and **Light** of the **Sammelalarm** FBox. The flashing control (= new alarm notification) has priority at the output **Leuchte** (light).

2. Alarm logging – Separately according to system

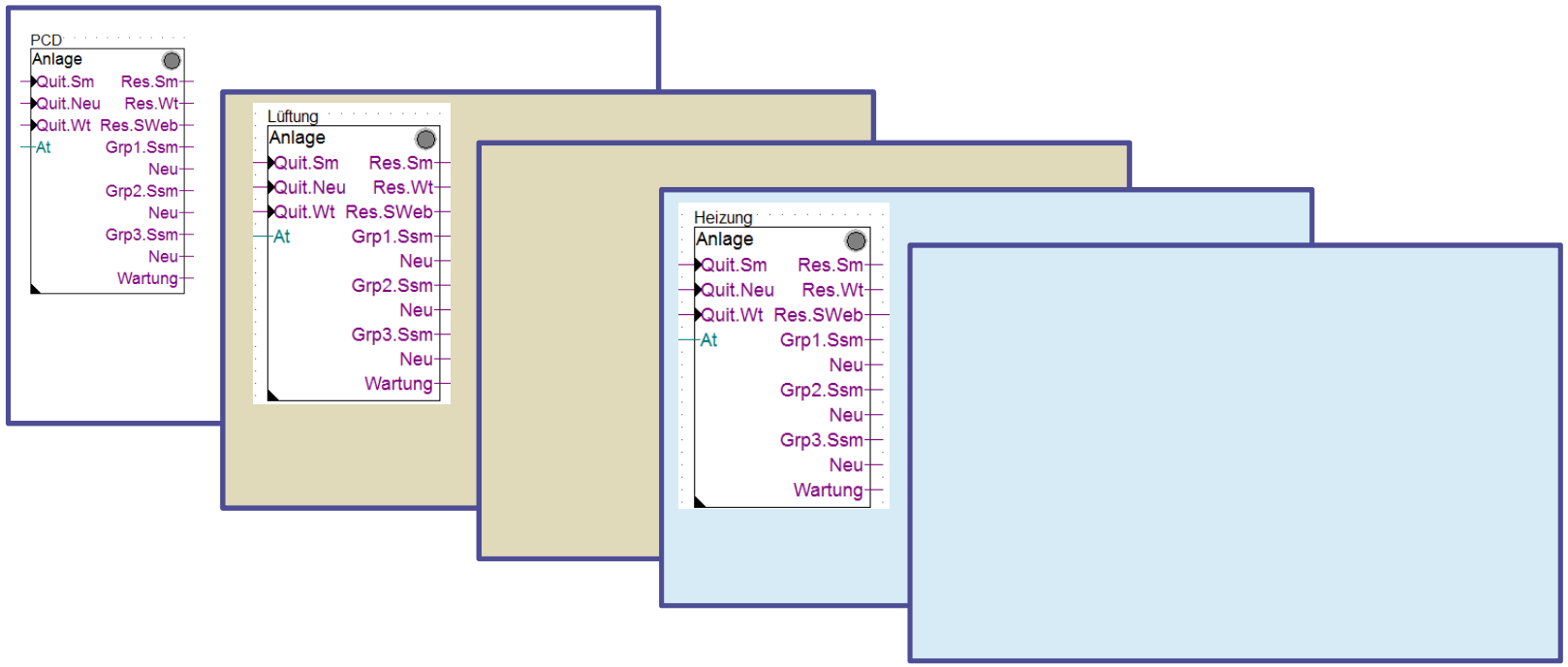
SAIA BURGESS CONTROLS

DDC Suite 2.7 / Functions

Alarm logging – Separately according to system

In most cases, multiple systems are programmed for each PCD. This means that all alarms would accrue in a single **Anlagenalarm** FBox (system alarm), which would not be of much benefit in most cases.

That is why it is possible to place the **Anlagenalarm** FBox (system alarm) multiple times. It is normally always placed at the start of a system, and sometimes also at the start of the program:

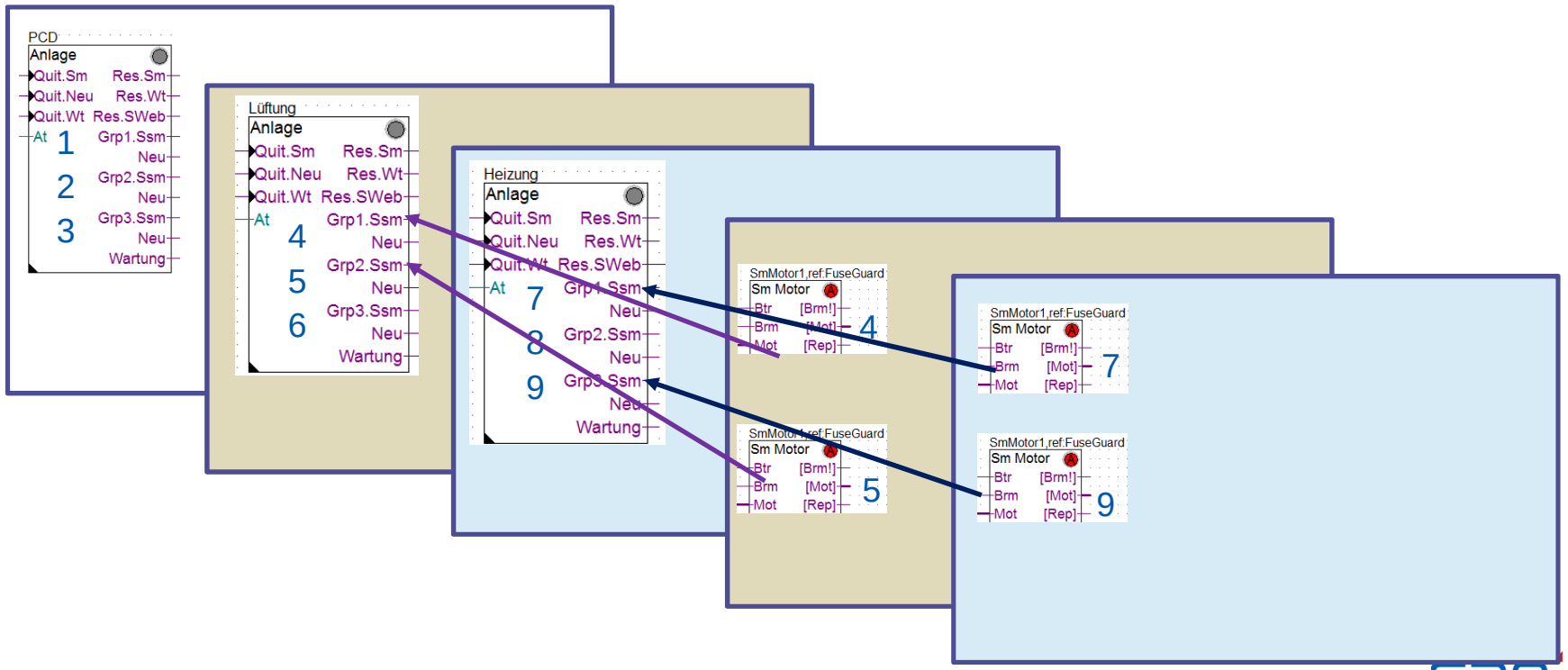


DDC Suite 2.7 / Functions

Alarm logging – Separately according to system

Each system can use its own notification classes, resulting in a clear separation according to the unique numbering.

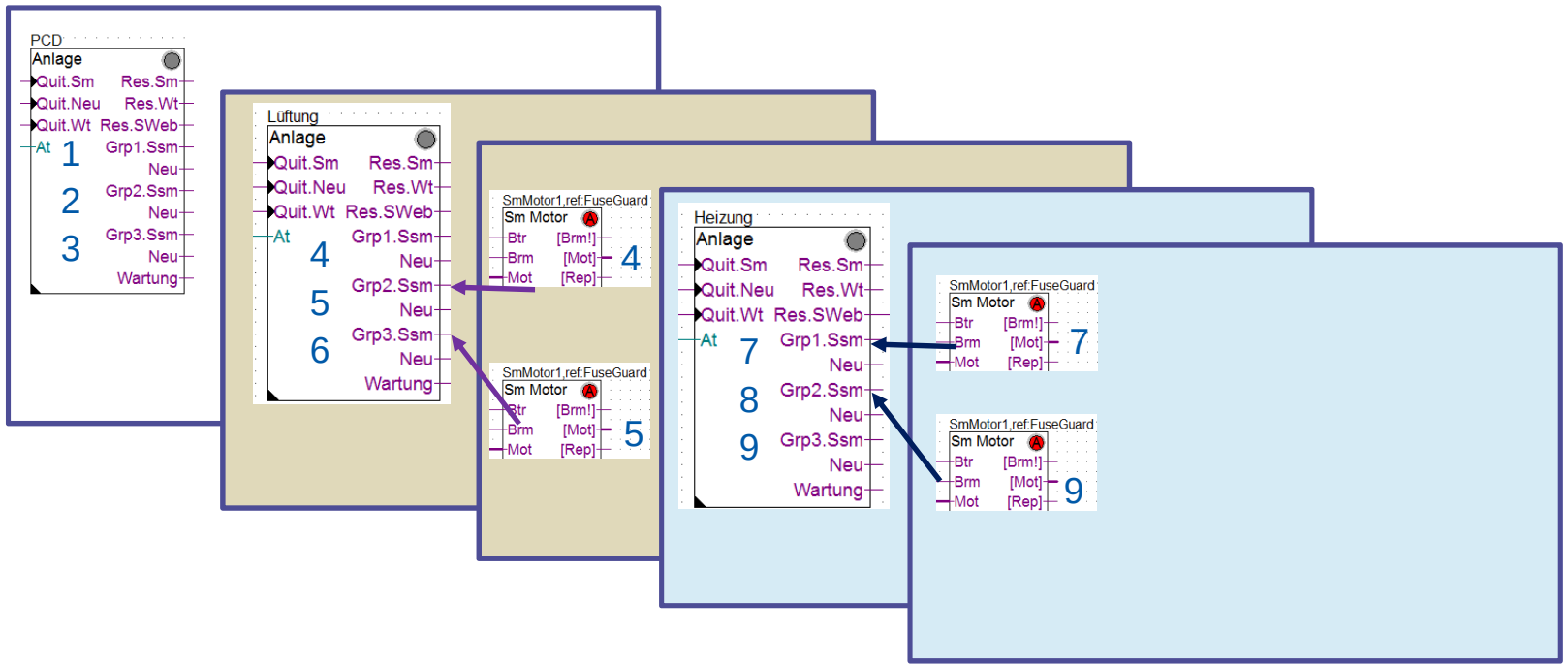
In this case, all **Anlagenalarm** FBoxes (system alarm) could be at the start of the program, followed by all other pages in any order. Although this may be necessary in some cases, it should be avoided where possible. (Info: It is not possible here to acknowledge separately according to systems)



DDC Suite 2.7 / Functions

Alarm logging – Separately according to system

This would be the ideal case. All system pages are coherent and well organised. Each system has clear notification classes.

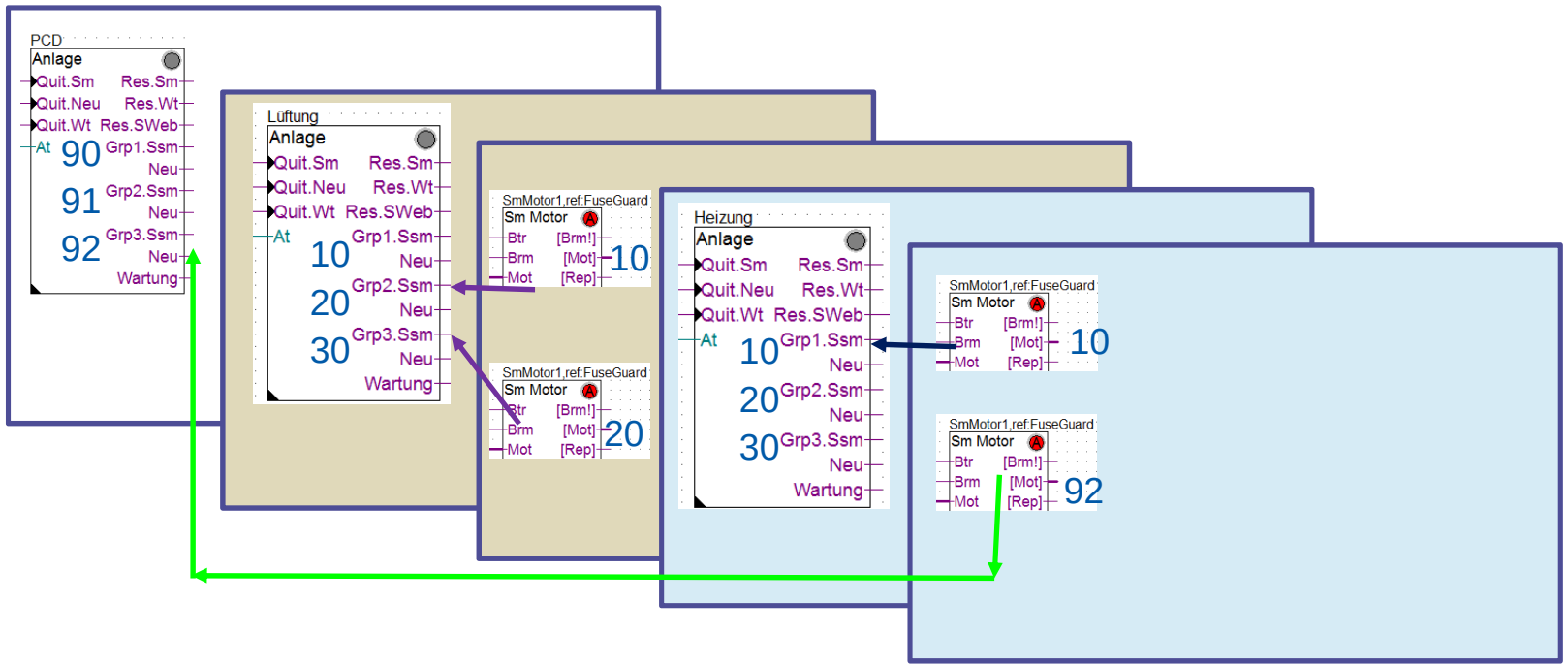


DDC Suite 2.7 / Functions

Alarm logging – Separately according to system

With BACnet, you may encounter situations in which the notification classes for all systems are identical. This is also easy to implement.

Here, the NC 10, 20 and 30 are identical for all systems. The sequence of the **Anlagenalarm** FBoxes (system alarm), however, determines the alarm groups in which the alarms are registered. This means that despite identical NCs, it is easy to clearly separate according to systems.



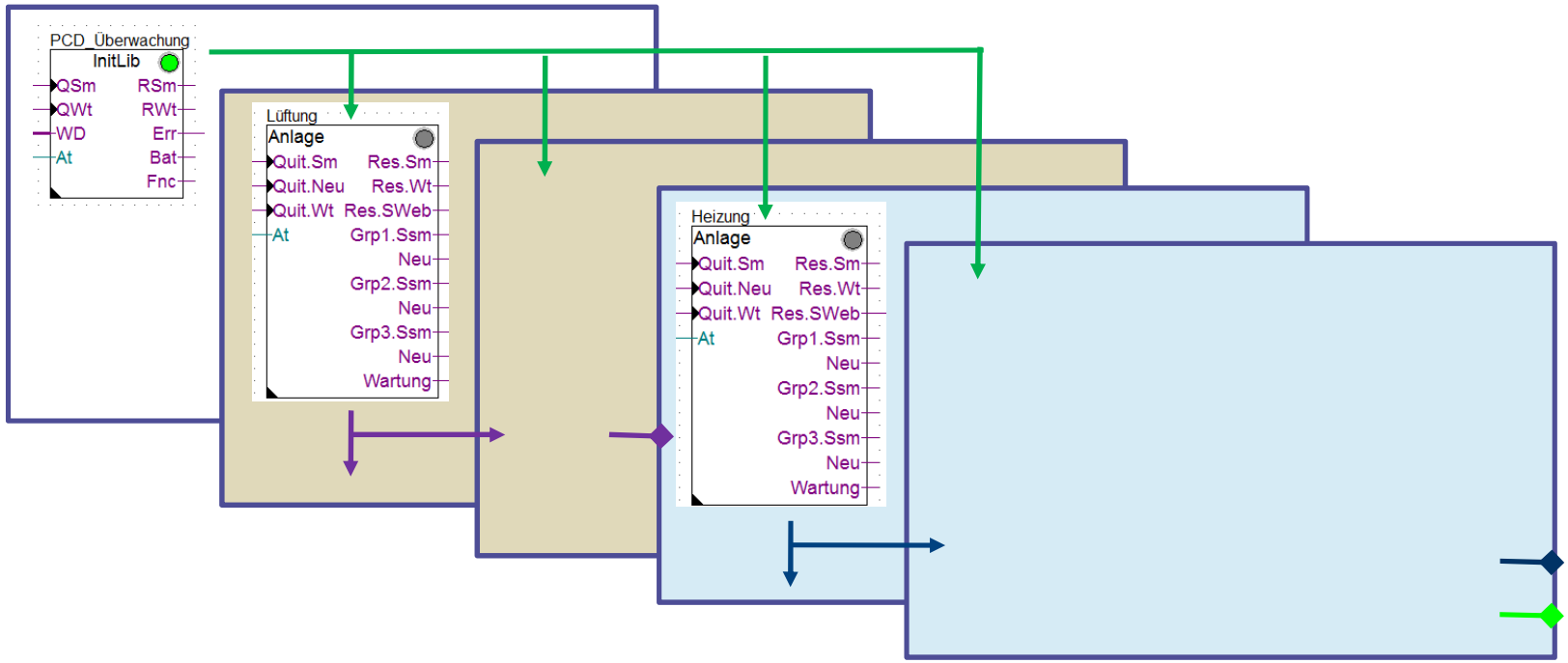
DDC Suite 2.7 / Functions

Alarm logging – Separately according to system

This also applies for acknowledgement of alarms and maintenance messages.

If acknowledging via FBox Anlagenalarm (system alarm) (input **Quit.Sm**/**Quit.Wart**) or the internal buttons, only alarms and maintenance events up to the next **Anlagenalarm** FBox (or end of the program) are reset.

If acknowledging via FBox InitLib (input **QSm**/**QWt**) or the internal buttons, ALL alarms and maintenance events up to the end of the program are reset.



3. Alarm logging – Alarm lists

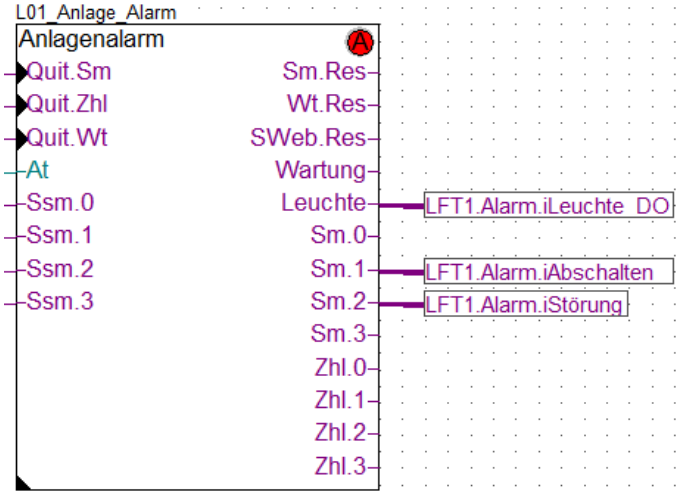
SAIA BURGESS CONTROLS

DDC Suite 2.7 / Functions

Alarm logging – Alarm lists

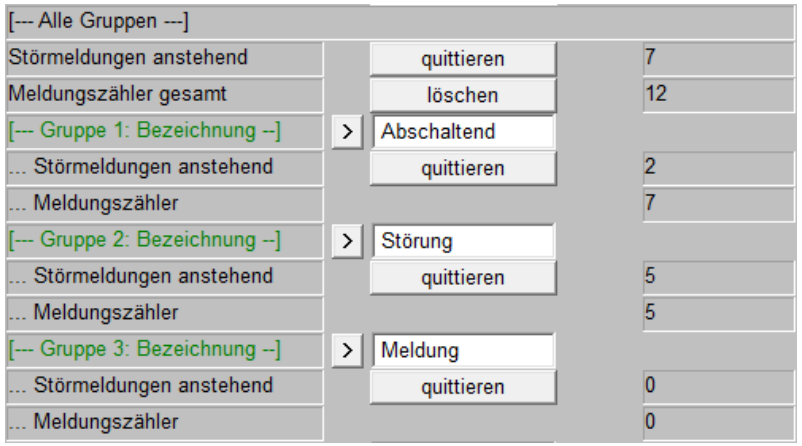
Using Anlagenalarm FBoxes (system alarm) makes alarm logging very convenient.

All alarms within the systems are now registered as groups without complex wiring.



The counting of pending and newly arisen alarms is also very neat.

There is one problem with this kind of alarm grouping. Although we can see exactly how many alarms are pending, we cannot see which alarm can be found on which Fupla page.

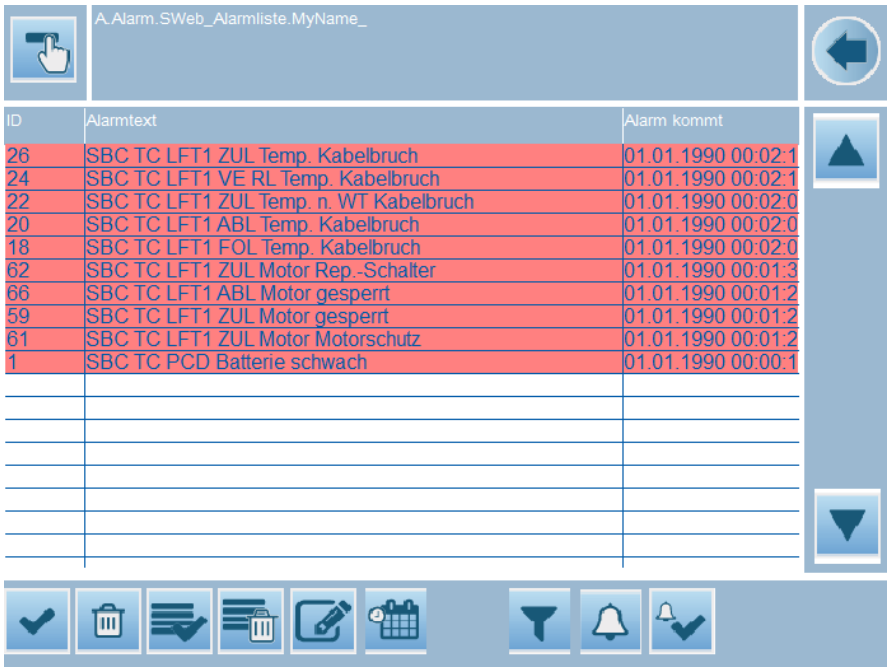


DDC Suite 2.7 / Functions

Alarm logging – Alarm lists

Most time during commissioning is spent dealing with alarms. As a PCD usually contains multiple programmed systems, a general overview of the alarms would be helpful.

The ideal solution would be an alarm list with coloured representation and time and date stamp. This can be implemented fairly easily.



DDC Suite 2.7 / Functions

Alarm logging – Alarm lists

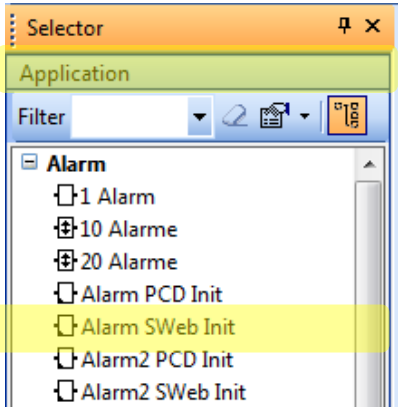
Alarm lists are set up using "… Init" FBoxes from the FBox family Alarm.

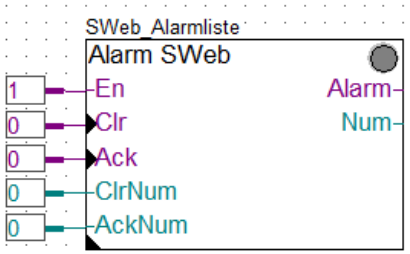
The simplest way to do this is using the **Alarm SWeb Init** FBox. This must have a clear name, in this example **SWeb_Alarmliste** (alarm list). The name may not contain any special characters, and may be no longer than 20 characters. The FBox should be placed on one of the first pages in the program.

The **Anzahl der Alarme** (number of alarms) should be adjusted if necessary. An alarm list can administer up to 1000 alarms.

It is also possible to create multiple alarm lists, such as one for each system. Each alarm list must be given a unique name.

In most cases however, only a single alarm list is used. This also makes it easier to control.





DDC Suite 2.7 / Functions

Alarm logging – Alarm lists

As it is possible to have multiple alarm lists, make sure that you set which alarm should appear in which alarm list.

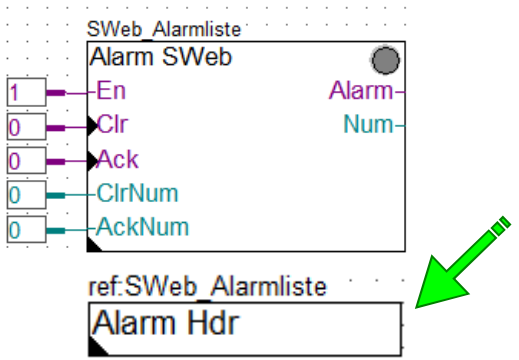
This requires the FBox **Alarm Hdr** even if you are using only a single alarm list. Ideally, this FBox should be placed after the FBox used to create the alarm list.

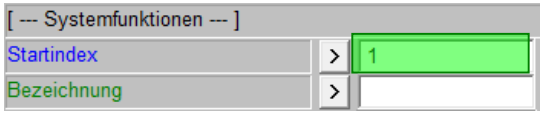
Adjust the reference of the FBox to the name of an alarm list, in this case **SWeb_Alarmliste**. This means that all alarms that are parametrised for an alarm list in the following FBoxes appear in the alarm list named **SWeb_Alarmliste**. Set the start index (= first alarm number) to 1. The parameter **Bezeichnung** (designation) remains empty.

The FBox can be repositioned in order to:

-allow the start index (= new, first alarm number) for the second system (e.g. with alarm number 101) to begin

-place the reference on a different alarm list. This means that the following alarms will appear in the referenced alarm list. In this case, always give the new start index. Attention! Don't forget to use consecutive alarm numbers for each alarm list!





Tip: If using multiple alarm lists, work according to separate lists when processing in the program.

-1. List + all associated alarm FBoxes in sequence

-2. List + all associated alarm FBoxes in sequence

Avoid constantly switching the alarm lists

DDC Suite 2.7 / Functions

Alarm logging – Alarm lists

A useful alarm list requires clear and effective alarm texts. On top of this, systems are programmed that are often (virtually) identical. No one wants to type in all this text again just because of different system numbers.

In order to create alarm texts efficiently, you can use a AKS (station identification system, SIS) with text blocks. A AKS generally has a hierarchical structure, for example:

Building Room System Installation location Component Designation

This can be specified by the planner/customer. As an example, we will take the **SBC** control centre in Murten with the **ventilation system** in the **training centre**. A **supply air temperature sensor** (Zulufttemperaturfühler) should receive the following alarm text in the event of a **broken cable** (Kabelbruch) or **short circuit** (Kurzschluss):

SBC TC LFT1 ZUL Temp. Kabelbruch
SBC TC LFT1 ZUL Temp. Kurzschluss

The exhaust air pressure sensor (Abluftdruckfühler) is almost identical, the only difference is the sensor designation

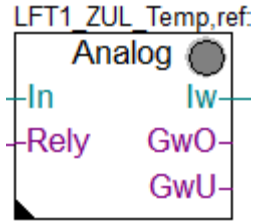
SBC TC LFT1 ABL Druck Kabelbruch
SBC TC LFT1 ABL Druck Kurzschluss

DDC Suite 2.7 / Functions

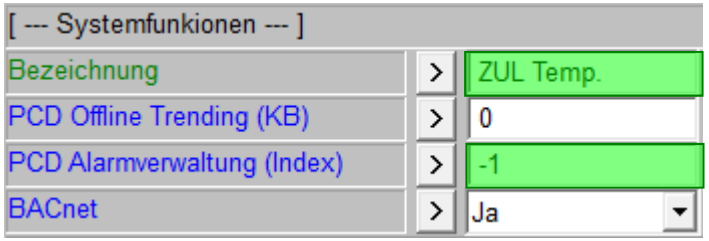
Alarm logging – Alarm lists

As you can see, the first part of the alarm text for a system is always identical. The component designation usually corresponds to an FBox, e.g. Messwert (measurement value). Detailed alarm text is also identical for (almost) all Messwert FBoxes.

Let's start with the Messwert FBox (measurement value). In the system functions group, we can determine the component to be processed with this FBox in the parameter **Bezeichnung** (designation). In this case **ZUL Temp.** (supply air temperature)



The parameter **PCD Alarmverwaltung (Index)** (PCD alarm management) was already mentioned previously. This parameter can be used to determine whether and how the alarms should appear in the alarm list:



--1 → The alarm numbers are automatically calculated (**recommended**), alarms will appear in the alarm list

- 0 → The alarms do NOT appear in the alarm list

-1..1000 →Set alarm number, alarms appear in the alarm list

DDC Suite 2.7 / Functions

Alarm logging – Alarm lists

Now we have to enter the alarm texts for exceeding (=broken cable) and falling below (=short-circuit).

The easiest way to do this is to search for the property ... Alarm text – in green text since PG5 2.2 – and enter the corresponding texts.

The alarm texts generated by the FBox consist of:

Designation + alarm text

We therefore get the following alarm texts from the FBox

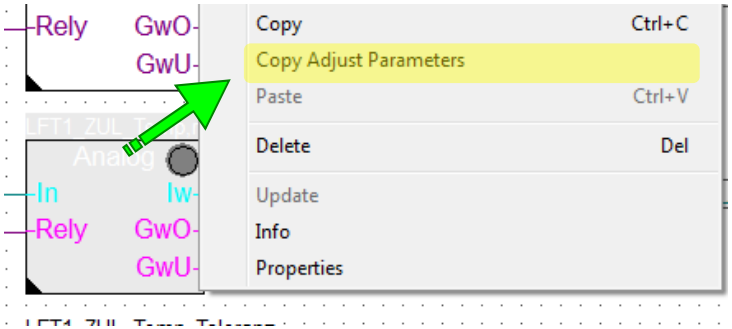
ZUL Temp. Kabelbruch
ZUL Temp. Kurzschluss

Tip: Always enter a space at the end of the text in the designation parameter.

DDC Suite 2.7 / Functions

Alarm logging – Alarm lists

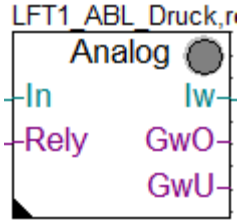
The procedure is exactly the same for the exhaust air pressure sensor. Here, you can save yourself the typing by copying the adjust parameter from the FBox of the Supply air Temp. measurement value (ZUL Temp.) to that of the Exhaust air Pressure measurement value (ABL Druck) (ideally before setting other parameters). This means you only have to change the component designation to **ABL Druck** (Exhaust air Pressure).



We therefore get the following alarm texts from the FBox

ABL Druck Kabelbruch

ABL Druck Kurzschluss



DDC Suite 2.7 / Functions

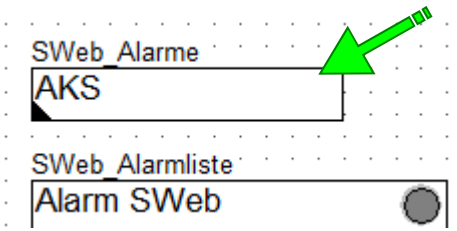
Alarm logging – Alarm lists

As you can see, the alarm text from the FBoxes is very detailed.
All that's missing is the common predesignation:

SBC TC LFT1 ZUL Temp. Kabelbruch

SBC TC LFT1 ABL Druck Kurzschluss

Although the only thing that usually changes is the system designation, you may also find that only the room changes and the system number remains the same. There must therefore be a way to precisely adjust a part of the AKS. It is therefore necessary to define the text blocks as independent elements.



The first step is to introduce an AKS using the FBox **AKS anlegen** (create AKS). The best place to position this FBox is in front of the FBox that creates the alarm list.

The parameter "Verwenden für" (Use for) is set to **Alarmierung** (alarming).

DDC Suite 2.7 / Functions

Alarm logging – Alarm lists

A AKS provides 10 text blocks (levels). These are simply arranged together in order.

Tip: Leave a space at the end of each text block to make the resulting text easier to read.

We arrange the levels as follows

-1 → SBC (=Building)
-2 → TC (=Room training centre)
-3 → SK (=Switching cabinet)

These are combined to form **SBC TC SK**

SK (=Switching cabinet) was chosen on purpose, as the FBox is usually positioned at the beginning of the program. There are sure to be other FBoxes here, such as a central outdoor air temperature. The alarm text that would then be created would be as follows

SBC TC SK AUL Temp. Kabelbruch
SBC TC SK AUL Temp. Kurzschluss

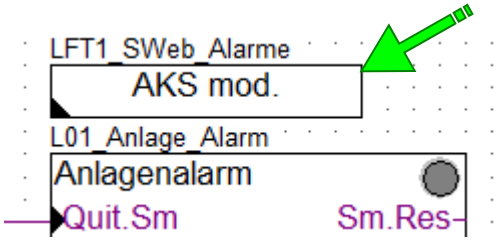
DDC Suite 2.7 / Functions

Alarm logging – Alarm lists

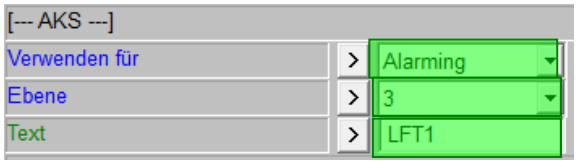
As you can see, the alarm text is composed according to the formula

AKS level 1 to 10 + FBox designation + alarm text

zusammen gebaut. All that remains for our ventilation system is to change the text block **SK** to **LFT1**.



The best way to do this is on the first page of our ventilation system.



As we only want to change ONE text in the AKS, we use the FBox **AKS anpassen** (adjust AKS). Here we can set the level at which the AKS is adjusted, as well as the new text. This should now be **LFT1**. The resulting alarm texts are then as follows:

SBC TC LFT1 ZUL Temp. Kabelbruch
SBC TC LFT1 ZUL Temp. Kurzschluss
SBC TC LFT1 ABL Druck Kabelbruch
SBC TC LFT1 ABL Druck Kurzschluss

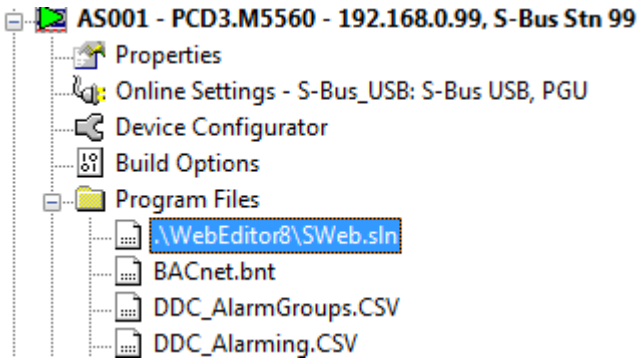
DDC Suite 2.7 / Functions

Alarm logging – Alarm lists

Parametrising and alarm list, creating the AKS and entering the texts in the FBoxes takes relatively little time. These are also saved when used as a template. If you use a template, you usually only need to change one or two text blocks, such as the system number or the building designation.

Now that the program has been prepared for an alarm list, it can be loaded into the PCD for commissioning.

All you have to do is create a WebEditor project with a page for the alarm list (see the WebEditor documents).



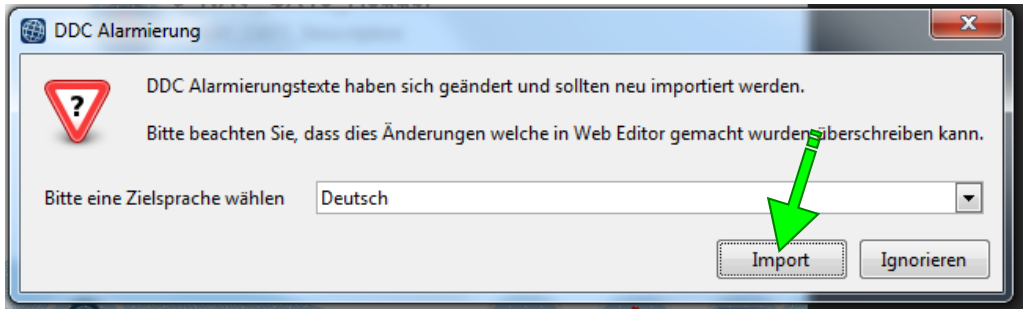
DDC Suite 2.7 / Functions

Alarm logging – Alarm lists

When you open the WebEditor, it will check whether there are any automatically generated alarm texts available.

If new alarm texts or changes (e.g. correction of a typo) are detected, the WebEditor checks whether the modified alarm texts should be loaded into the WebEditor project.

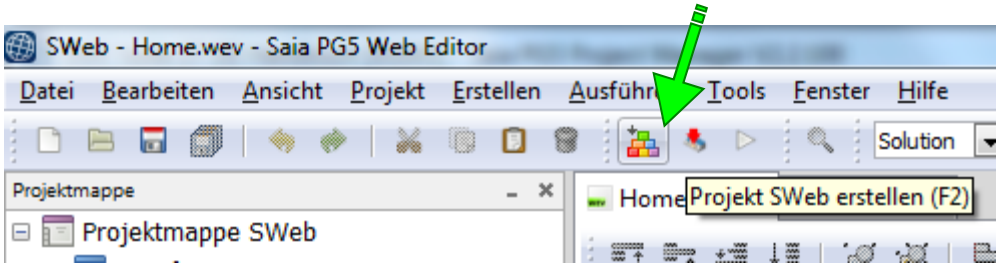
The alarm texts in the WebEditor project are always updated by clicking on **Import**.



DDC Suite 2.7 / Functions

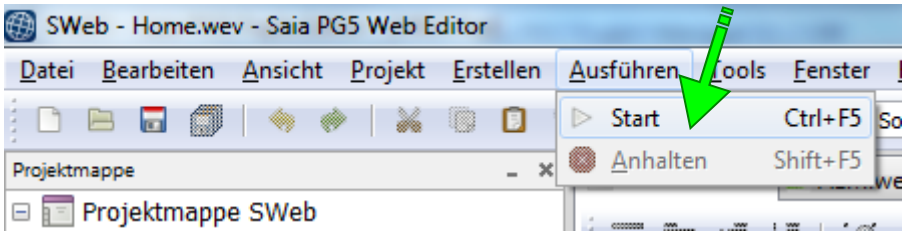
Alarm logging – Alarm lists

You then need to create the WebEditor project. You can do so via the icon with the coloured bricks or by pressing the function key F2.



Once it has been successfully created, you can open the WebEditor project. In PG5 2.2, it is no longer necessary to download the web project to the controller. This means that you can use the web alarm list even if the PCD has no file system.

Thanks to the new feature Run/Start, the web application is started directly on the PC, with only the data read from the PCD.

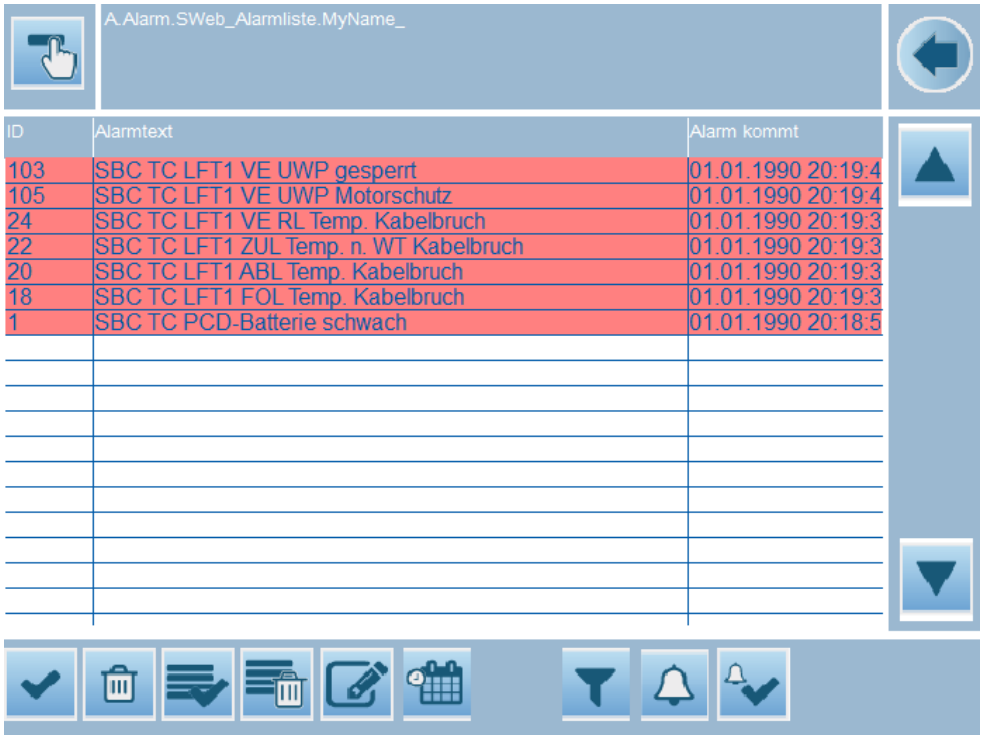


DDC Suite 2.7 / Functions

Alarm logging – Alarm lists

The alarm list now reflects the current state of alarms in the PCD. This makes it an ideal expansion to alarm grouping with the FBox **Anlagenalarm** (system alarm). This is very helpful for commissioning.

Often, you will need to create a web visualisation, resulting in double benefit.



DDC Suite 2.7 / Functions

Alarm logging – Alarm lists

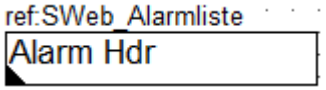
In the FBox **Alarm Hdr**, the acknowledgement of an alarm to the FBox has been restored as a default setting. The FBox will automatically reset the alarm if possible. This would be the case if:

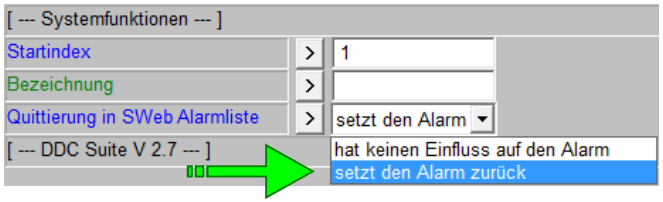
-the input status is already the normal status

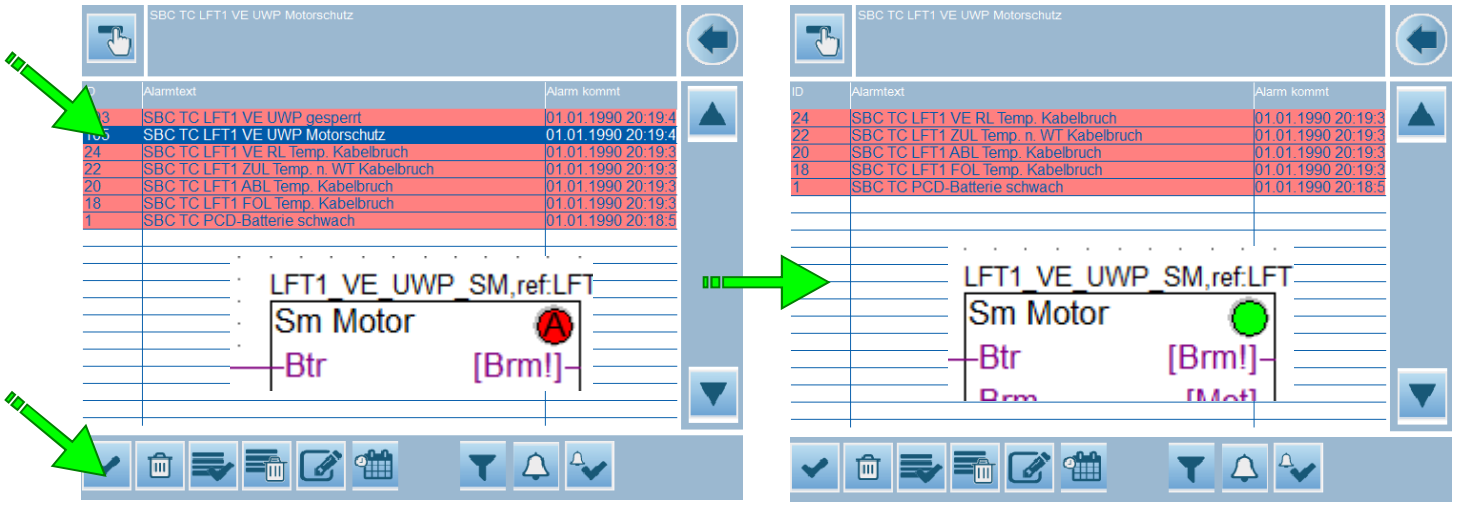
-as soon as the input status reaches the normal status

-immediately in time-delayed alarms (operating feedback absent)

This feature can be deactivated if required.







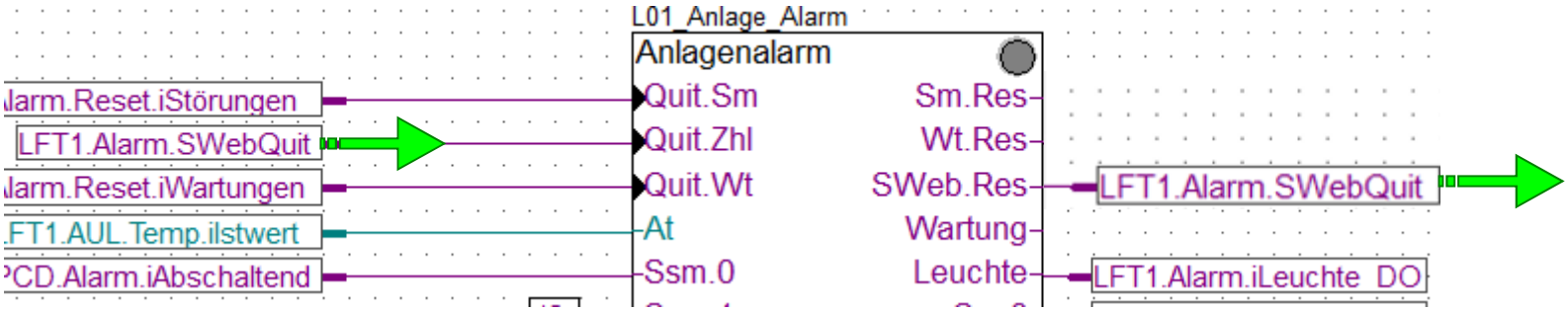
DDC Suite 2.7 / Functions

Alarm logging – Alarm lists

If acknowledging in the SWeb alarm list, it may be desirable to reset the signalisation (light) if this still flashes (new alarm notification).

The **Anlagenalarm** FBox (system alarm) offers an integrated solution. The output **SWeb.Res** output also has a high for a program cycle if an alarm that is registered in this system alarm FBox has been acknowledged in the SWeb alarm list.

This pulse can be tapped and connected at the input **Quit.Zhl**. Delete the new alarm notifications to stop the output **Leuchte** (light) flashing. If no alarm is active, the light will go out. The light will illuminate permanently if at least one alarm is active.



DDC Suite 2.7 / Functions

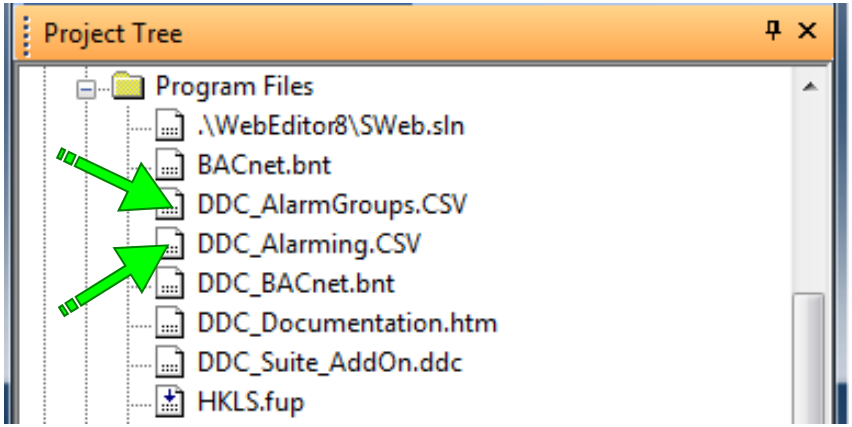
Alarm logging – Alarm lists

Tip: You can quickly check the parametrised alarm data points if necessary in an overview.

2 files are listed under **Program Files** that are updated after every successful compilation:

-DDC_Alarming.CSV → A list of all alarms that will appear in alarm lists, together with alarm number and alarm text

-DDC_AlarmGroups.CSV → A list of all alarms, along with their allocation to the system alarm FBoxes and the set notification classes



DDC Suite 2.7 / Functions

Alarm logging – Alarm lists

Each alarm list is expanded with a line "ListDefinition=x" and the name of the alarm list in the following column. The "x" is a consecutive numbering, and helps allocation. The following information is visible for each alarm:

-Column A: List_x → Allocation to alarm list (=ListDefinition=x)

-Column B: Alarm number

-Column C: Internal tag with the alarm number, is used by the WebEditor

-Column D: Alarm text

This file cannot be edited, as the file is deleted and reconstructed at the next compilation.

It serves only to check the alarm texts and alarm numbers.

DDC_Alarming.CSV file

	A	B	C	D
1	ListDefinition=1	SWeb_Alarmliste		
2	List_1	1	Alarm_1	SBC TC PCD-Batterie schwach
3	List_1	2	Alarm_2	SBC TC PCD-Interner Fehler
4	List_1	3	Alarm_3	SBC TC PCD-230VAC
5	List_1	4	Alarm_4	SBC TC PCD-24VAC
6	List_1	5	Alarm_5	SBC TC PCD-24VDC
7	List_1	6	Alarm_6	SBC TC PCD-Phasenwächter
8	List_1	7	Alarm_7	SBC TC PCD-Steuerkreis
9	List_1	8	Alarm_8	SBC TC PCD-BMA
10	List_1	9	Alarm_9	SBC TC PCD-AUL Temp. Kabelbruch
11	List_1	10	Alarm_10	SBC TC PCD-AUL Temp. Kurzschluss
12	List_1	11	Alarm_11	SBC TC LFT1 230VAC
13	List_1	12	Alarm_12	SBC TC LFT1 24VAC
14	List_1	13	Alarm_13	SBC TC LFT1 24VDC
15	List_1	14	Alarm_14	SBC TC LFT1 Phasenwächter
16	List_1	15	Alarm_15	SBC TC LFT1 Steuerkreis

DDC Suite 2.7 / Functions

Alarm logging – Alarm lists

The sequence of the entries corresponds to that of the FBoxes. Each system alarm FBox indicates:

-Column A: Name of the FBox
-Column B: Unused
-Column C: Alarm group 1...5
-Column D: Alarm group designation
-Columns E-F: Allocated NC of the alarm group

Each FBox that supports alarming indicates:

-Column A: Name of the allocated system alarm FBox
-Column B: NC (-1 = not allocated to a group)
-Column C: Name of the FBox
-Column D: Alarm text

This file cannot be edited, as the file is deleted and reconstructed at the next compilation. It serves only to check the alarm texts and alarm numbers.

DDC_AlarmGroups.CSV file

	A	B	C	D	E	F	G	H	I
1									
2	PCD_Alarmgruppen		Alarmgruppe	Bezeichnung	1. NC	2. NC	3. NC	4. NC	5. NC
3	PCD_Alarmgruppen		1	Abschaltend	10	-1	-1	-1	-1
4	PCD_Alarmgruppen		2	Störung	20	-1	-1	-1	-1
5	PCD_Alarmgruppen		3	Meldung	30	-1	-1	-1	-1
6	PCD_Alarmgruppen		4	Alarmgruppe 4	-1	-1	-1	-1	-1
7	PCD_Alarmgruppen		5	Alarmgruppe 5	-1	-1	-1	-1	-1
8									
9	PCD_Alarmgruppen	30	PCD_Intern_Überwachung	SBC TC PCD-Batterie schwach					
10	PCD_Alarmgruppen	30	PCD_Intern_Überwachung	SBC TC PCD-Interner Fehler					
11	PCD_Alarmgruppen	10	PCD_Spg	SBC TC PCD-230VAC					
12	PCD_Alarmgruppen	10	PCD_Spg	SBC TC PCD-24VAC					
13	PCD_Alarmgruppen	10	PCD_Spg	SBC TC PCD-24VDC					
14	PCD_Alarmgruppen	10	PCD_Spg	SBC TC PCD-Phasenwächter					
15	PCD_Alarmgruppen	10	PCD_Spg	SBC TC PCD-Steuerkreis					
16	PCD_Alarmgruppen	10	PCD_BMA	SBC TC PCD-BMA					
17	PCD_Alarmgruppen	20	PCD_AUL_Temp	SBC TC PCD-AUL Temp. Kabelbruch					
18	PCD_Alarmgruppen	20	PCD_AUL_Temp	SBC TC PCD-AUL Temp. Kurzschluss					
19									
20	L01_Anlage_Alarm		Alarmgruppe	Bezeichnung	1. NC	2. NC	3. NC	4. NC	5. NC
21	L01_Anlage_Alarm		1	Abschaltend	10	-1	-1	-1	-1
22	L01_Anlage_Alarm		2	Störung	20	-1	-1	-1	-1
23	L01_Anlage_Alarm		3	Meldung	30	-1	-1	-1	-1
24	L01_Anlage_Alarm		4	Alarmgruppe 4	-1	-1	-1	-1	-1
25	L01_Anlage_Alarm		5	Alarmgruppe 5	-1	-1	-1	-1	-1
26									
27	L01_Anlage_Alarm	10	LFT1_Spg	SBC TC LFT1 230VAC					
28	L01_Anlage_Alarm	10	LFT1_Spg	SBC TC LFT1 24VAC					
29	L01_Anlage_Alarm	10	LFT1_Spg	SBC TC LFT1 24VDC					
30	L01_Anlage_Alarm	10	LFT1_Spg	SBC TC LFT1 Phasenwächter					

DDC Suite 2.7 / Functions

Alarm logging – Alarm lists

As you can see, DDC Suite 2.7 offers a comprehensive and integrated alarm grouping function.

-Alarm grouping separately according to systems

-Division of alarms in up to 5 groups

-Alarm acknowledgement: Centrally for the entire program, per system, per group for each system or for each alarm via web visualisation

-Efficient checking of alarm texts by use of a AKS (station identification system, SIS)

-Automatic creation of alarm texts and inclusion in the web visualisation, even for commissioning, with just a few mouse clicks

-Completely prepared in the system templates

4. BACnet

SAIA BURGESS CONTROLS

DDC Suite 2.7 / Functions

BACnet

What is BACnet?

- BACnet stands for **B**uilding **A**utomation and **C**ontrol**net**work
- BACnet is a data protocol for exchanging data from different systems and devices in building automation
- BACnet describes the representation of objects and how they interact with other objects, not their internal function
- BACnet is a registered trademark of ASHRAE
- BACnet supports countless network standards and topologies, incl. the internet protocol (BACnet/IP)

DDC Suite 2.7 makes it easier to configure the PCD as a BACnet device. BACnet is not a pure data protocol, however, but also provides a large number of integrated functions. For this reason, we will look at configuration in the FBoxes in more detail.

We urgently recommend attending a basic BACnet course, in order to understand the internal functions and mechanisms.

DDC Suite 2.7 / Functions

BACnet

As with alarming, a AKS (station identification system, SIS) is required for data points that are provided for BACnet. BACnet requires two AKS, however:

-The **object name**: This is usually a technical address at which the individual levels are specified, often with a set length. The subdivision of the levels is also often marked with special characters. Examples:

SBCTC_SK_AULT__ = Each level must always have 3 characters. «_» is a fill character

SBC:TC/SK-AUL:T = Levels can be of different lengths. Special characters are the separators

AKS systems in BACnet are usually defined and specified by the planner or end customer, and are to implemented precisely as specified.

-The **description**: This text clearly defines the data point. Example:

SBC Murten training centre switching cabinet outdoor air temperature

This usually follows the structure of the **object name**, and is also defined and specified by the planner or end customer. Deviations are mostly accepted, however.

Each BACnet object possesses the properties **object name** and **description**. Both AKS need to be parametrised. Over the course of this manual, we will focus roughly on the AKS that we have already used for the alarming.

DDC Suite 2.7 / Functions

BACnet

We enter the FBox **AKS festlegen** twice on a Fupla page used for the basic BACnet configuration. We name the FBoxes to make the purpose for a AKS clear.

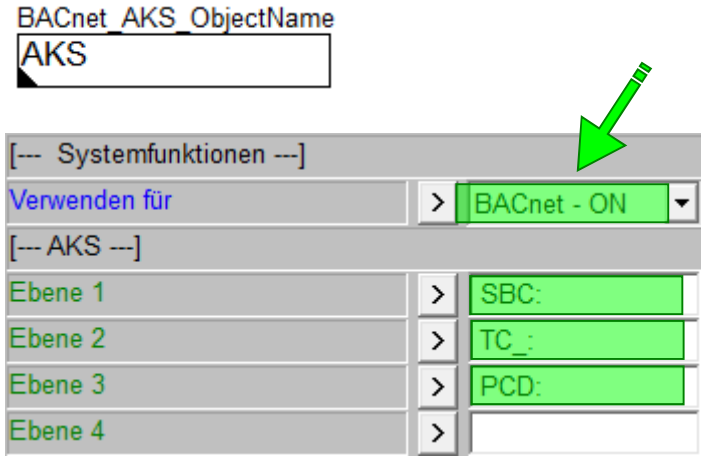
In the FBox that the AKS should create for the **object name**, the option **BACnet – ON** (=**O**bject**N**ame) in the parameter **Verwenden für** (use for) should be activated.

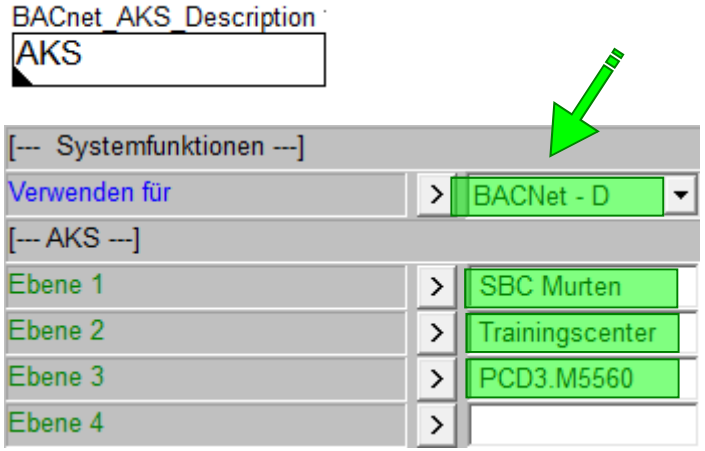
We then enter the technical addresses for the levels. The specification is as follows: Each level must be 3 characters long, with an additional colon (":") used as a separator.

In the FBox that the AKS should create for the **description**, the option **BACnet – D** (=**D**escription) in the parameter **Verwenden für** (use for) should be activated.

The texts should follow the levels of the object name, but can be named as you wish. A space (" ") should be used as a separator between the levels.

As you can see, the FBoxes **AKS festlegen** are used to reconstruct the part of the AKS that is identical for all following BACnet objects.

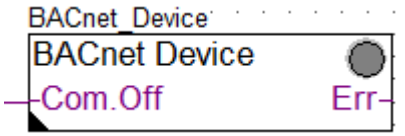




DDC Suite 2.7 / Functions

BACnet

Next, we require the FBox **BACnet Device**.



The parameter **BACnet aktivieren** (activate BACnet) is used to centrally control whether the FBoxes in DDC Suite 2.7 should support BACnet.

No → BACnet is completely deactivated, even if the parameter for BACnet is not set to no in the subsequent FBoxes.

Yes → BACnet is activated and is supported in the subsequent FBoxes. We will explain later what exactly

The settings for (BACnet) **Standard** and **Character set** may require coordination with the BACnet BMS.

The **DataLink Layer** (=communication) is usually defined in the device configurator (BACnet card).

The **ID**, **name**, **description** and **location** are generally specified by the BACnet planner. The ID and name **must** be entered once in the network!

DDC Suite 2.7 / Functions

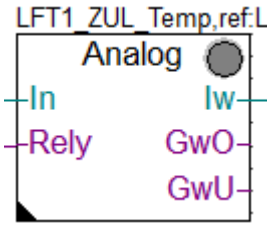
BACnet

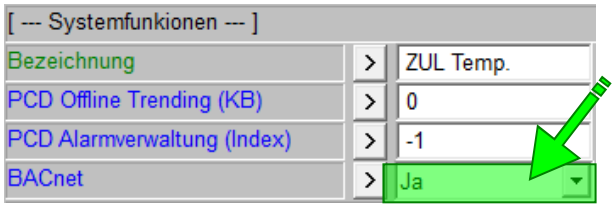
It is now possible to select whether BACnet should be supported in the FBoxes of DDC Suite 2.7. In the most simple scenario, an FBox only provides a single BACnet object.

If a parameter **BACnet** is present in the group **[--- Systemfunktionen ---]** (system functions), the FBoxes can provide objects.

You can read the objects that are created in properties. The help section of the FBoxes always contains a section called **Systemfunktion BACnet**. This explains which objects should be created and at which selection in the parameter **BACnet**.

The AI is therefore always created in the **Messwert** FBox (measurement value). We will ignore the TR object at this point.





Systemfunktion BACnet

Diese FBox kann BACnet Objekte automatisch erzeugen. Die entsprechenden Optionen im Parameter "BACnet" legen folgende BACnet Objekte an:

Option Objekt	Bemerkung	Beschreibung	
Ja	AI	-	Physikalischer Messwert
Ja	TR	optional	Physikalischer Messwert

DDC Suite 2.7 / Functions

BACnet

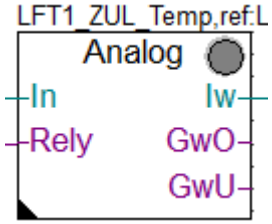
The properties then contain (at least) one parameter **<--- BACnet Object Name --->**. Enter the rest of the technical address here (AKS BACnet – ON).

But wait – This FBox has so many parameters, like correction, card type, etc. Which value does this BACnet object represent?

It is **always** the first value that is available online and can be found above the parameter **<--- BACnet Object Name --->** . In this case, the parameter **Physikal. Wert (Korrigiert)** (physical value (corrected)).

Enter **ZUL:T__** as the technical address, and **Zulufttemperatur** (supply air temperature) for the description.

Set the parameter **<--- BACnet Trendlog --->** to **No**, we will come to it later.



DDC Suite 2.7 / Functions

BACnet

As mentioned at the start, BACnet has several integrated functions and mechanisms.

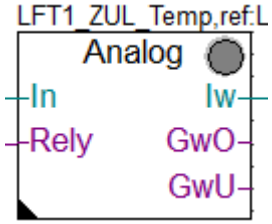
This means that a simple BACnet = Yes and name/description is not enough.

Set the parameter **– Notification class** to **-1**. We had already used this parameter to combine the alarms in this FBox in an alarm group. This parameter is therefore used twice.

We need the alarming for BACnet later, so let's simply deactivate it for now with **-1**. We will see later how this double benefit can be used smartly.

In general, all data points in BACnet have a unit or status texts. For this reason, select **Grad Celsius** (degrees Celsius) for the supply air temperature in the parameter **Einheit** (unit).

The PCD transmits these values automatically when they change, meaning that the BACnet BMS does not have to keep querying all values. To prevent this happening too often, you can set the threshold in the parameter **COV Hysterese** (COV hysteresis). In this case, the PCD would transmit a value change if the value has changed since the last transmission by +/- 1.0 degree.



<--- Intrinsic Reporting --->		
- Notification-class	>	-1
- Limit Enable	>	(Low/High)
<--- Reliability --->	>	Phys. Wert < M
[--- Definitionen ---]		
Einheit	>	Grad Celsius
COV Hysterese	>	1.0

DDC Suite 2.7 / Functions

BACnet

You may occasionally find that the predefined unit list does not contain the required unit.

The **Einheiten** (units) FBox can help here. This FBox is optional, and therefore does not have to be positioned.

The FBoxes detect the language set for them in PG5, and therefore automatically use the correct unit designations for EN, DE, FR and IT. All other languages always use EN.

This FBox can be used to override the language used for the unit texts. This can be helpful if you prefer to program with EN, for example, but require DE as the target language.

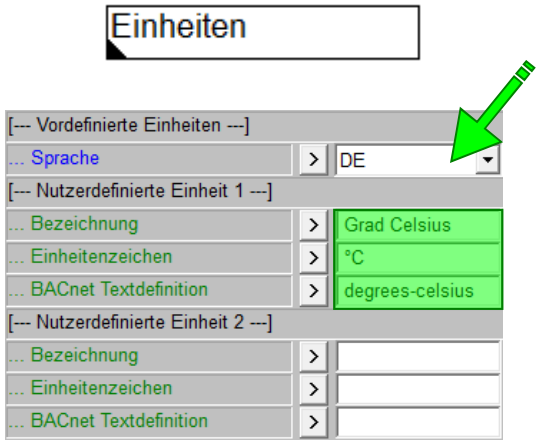
You can also set up to 10 user-defined units.

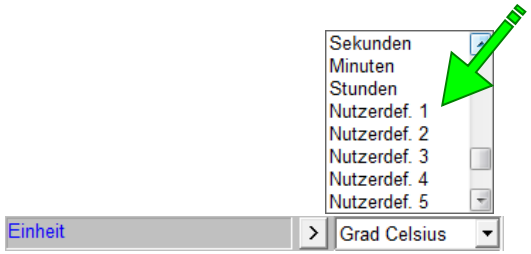
-**Bezeichnung** (Designation) → The standard text of the unit

-**Einheitenzeichen** (Unit symbol) → Is used in trend displays, for example

-**BACnet Textdefinition** (BACnet text definition) → BACnet-compliant unit designation

The 10 user-defined units can then be accessed via the units list. The representation is always neutral here, however, i.e. «**Nutzerdef. 1.. 10**» (user def. 1...10)





DDC Suite 2.7 / Functions

BACnet

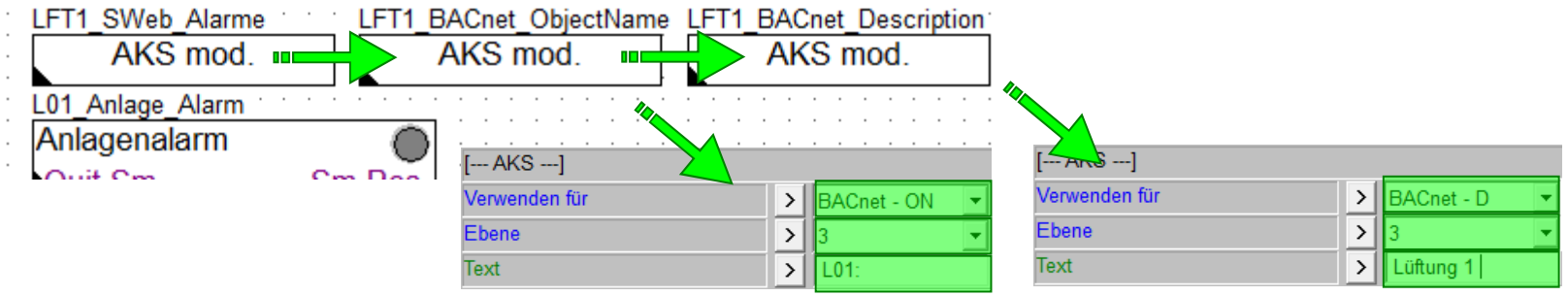
We therefore end up with a BACnet object in line with **AKS BACnet - ON level 1..10 + FBox ON** and **AKS BACnet - D level 1..10 + FBox D** with

Object name SBC:TC_:PCD:ZUL:T__
Description SBC Murten training centre PCD3.M5560 Supply air temperature

This looks good, but it is not what we expected. This is an object for the ventilation system. The third level of the AKS must therefore be adjusted. The best way to do this is on the first page of our ventilation system where we already incorporated it for the alarm list.

We now required 2 FBoxes **AKS anpassen** (adjust AKS), however. One for the ObjectName (BACnet - ON) and another for the Description (BACnet - D). The third level of each should be adjusted

BACnet – ON →**L01:**
BACnet – D →**Lüftung 1** (ventilation 1)



DDC Suite 2.7 / Functions

BACnet

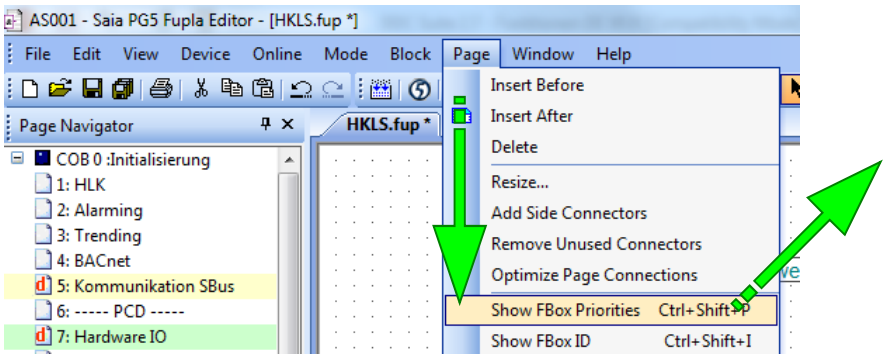
After adjusting level 3, the resulting text is then:

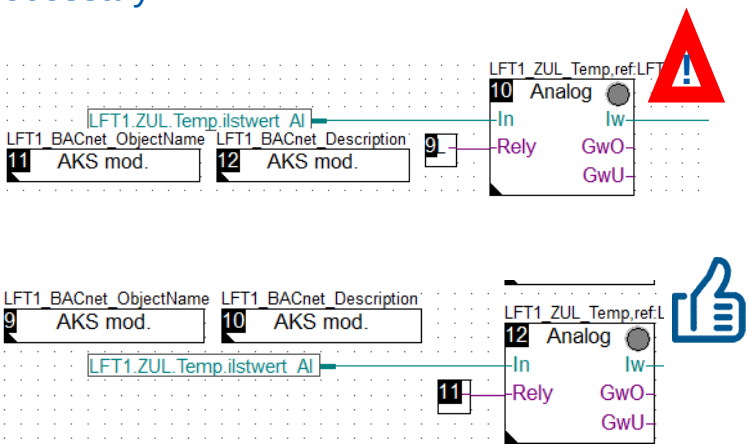
ObjectName	SBC:TC_:L01:ZUL:T__
Description	SBC Murten training centre Ventilation 1 Supply air temperature

Tip: The FBoxes **AKS festlegen** (def AKS) and **AKS anpassen** (adjust AKS) can be positioned as often as needed. This does not occupy data points in the PCD.

Depending on the AKS of the planner/end customer, it may be necessary to change a level in the middle of a system, e.g. if the room number should always represent the installation location of the component.

The only important thing to remember here is to place these FBoxes **in front of** the FBoxes that generate the BACnet objects. You may find that although this is optically correct, the resulting texts are not. In this case, check the "FBox Priorities" and shift the FBoxes if necessary.





DDC Suite 2.7 / Functions

BACnet

In most cases, the FBoxes create multiple BACnet objects. The FBox Motor, for example, can provide up to 5 objects.

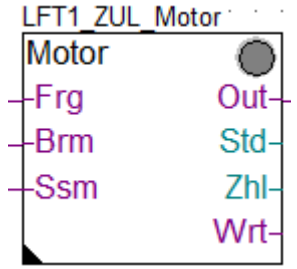
HMI (=FBox output), + Rm (=operating feedback), + WT (=maintenance message) ….

This makes it easy to check the number of prepared objects. Always bear in mind the maximum number of objects that a PCD can process!

If you only require the object for the output (HMI) and the operating hours (hrs), you automatically receive the objects that are in between in the list.

There is a solution however with which you can control whether each object should be created.

We first recommend always choosing the bottom option in the parameter **BACnet**. This pre-sets the FBox to create all BACnet objects.



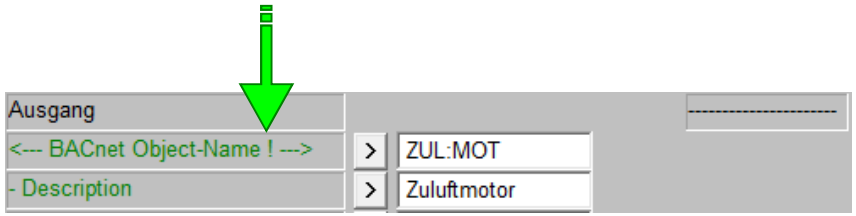
DDC Suite 2.7 / Functions

BACnet

If you do not want to create the **Motorstörungen** (motor faults, input Ssm) as a BACnet object, for example, leave the parameter **<--- BACnet Object Name --->** empty or delete the text present.

Please note: If the object name is empty, this object will not be created. This is fully sufficient, although we recommend also leaving the description empty for the purposes of clarity.

But: In some cases it is not possible to suppress objects. The HMI object is mandatory for the **Motor** FBox!



If an exclamation mark ("!") is included in the parameter **<--- BACnet Object Name ! --->**, this object is created. Even if the object name is empty.

The result will probably be a compilation error, as the supply and exhaust air fans create the HMI object with the same name (AKS BACnet – ON level 1..10). Object names must be unique!

DDC Suite 2.7 / Functions

BACnet

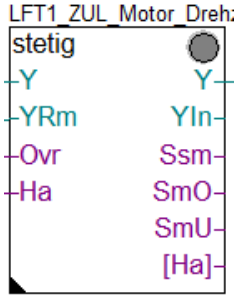
Some FBoxes have very many online parameters, and can also create a high number of BACnet objects. In order to slightly limit the number of parameters, some related BACnet objects use common text blocks.

Example using the FBox **Stetig** (constant). There are up to 3 objects here for the feedback. The feedback itself, a feedback error too high and a feedback error too low.

That is why there is a single parameter **<--- BACnet common --->**. Enter the technical address and description, which is identical for all 3 objects.

Each object will then have its own parameter as the process continues. **<--- BACnet Object Name --->**, but no description (saves parameters). The BACnet ObjectName and Description are then compiled as follows:

AKS BACnet – ON Level 1..10 + BACnet common ON + BACnet ON
AKS BACnet – D Level 1..10 + BACnet Description



[--- Rückmeldung ---]		
Rückmeldung in %		
<--- BACnet common --->	>	ZUL:MOT:Y__:
- Description	>	Zuluftmotor Drehza
- Optional text	>	
- Notification-class	>	30
<--- BACnet Object-Name --->	>	RM_
Rückmeldung vorhanden	>	Ja
Rohwert minimal	>	0
Rohwert maximal	>	1000
Laufzeit	>	180.0
Hysterese Rückmeldung	>	5.0
Quittierpflichtig	>	Nein
Meldungsunterdrückung	>	bei bel. Spg.
Meldung Überschreitung		
Alarmtext	>	RM zu hoch
<--- BACnet Object-Name --->	>	SM_:RMA
Meldung Unterschreitung		
Alarmtext	>	RM zu niedrig
<--- BACnet Object-Name --->	>	SM_:RMZ

DDC Suite 2.7 / Functions

BACnet

But that is not the whole truth. There is also hidden, pre-defined texts in the background.

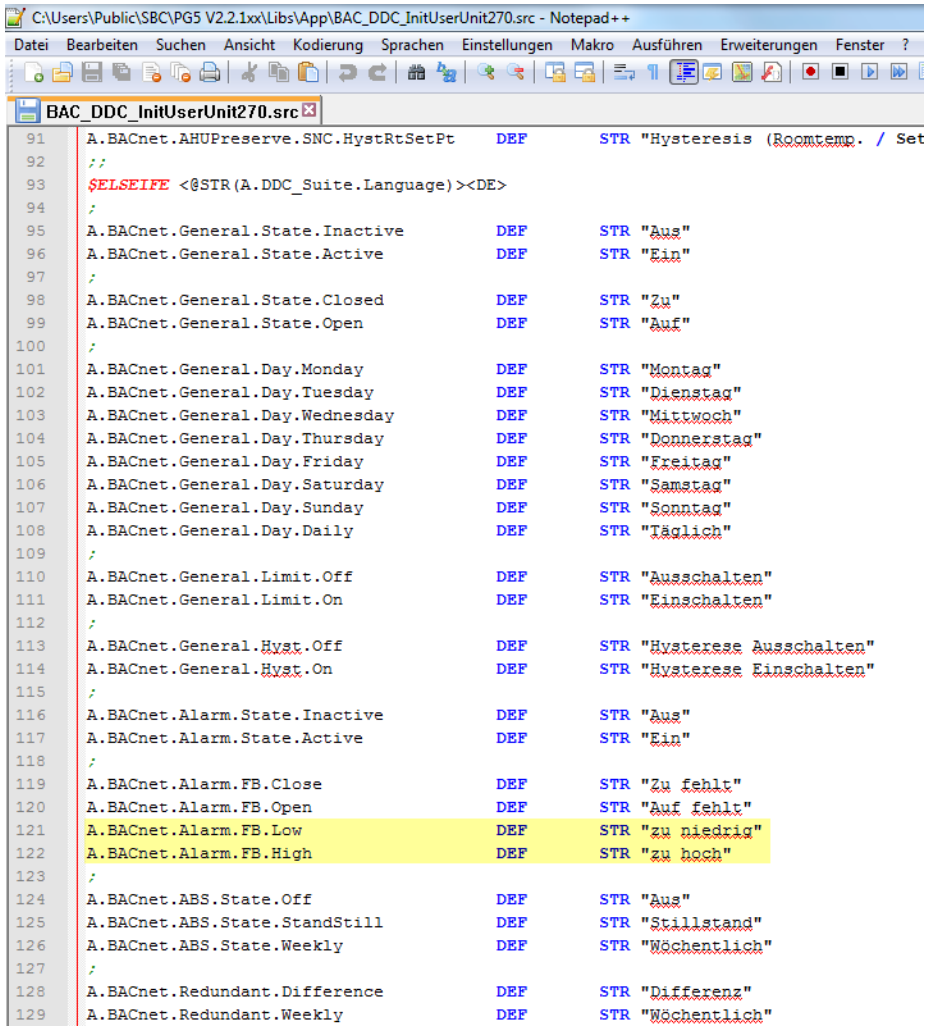
These are used for related objects to implement a differentiation in the description.

These texts are stored in the file

BAC_DDC_InitUserUnit270.src

. It is possible to adjust this file, but this requires detailed knowledge of how these texts are used.

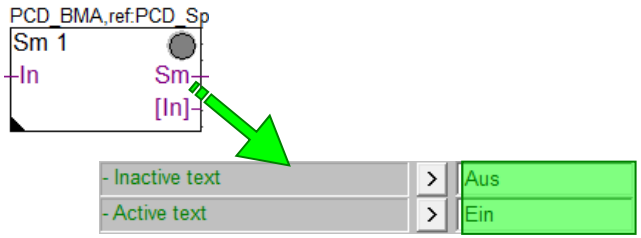
Please refer here to the one-day course "DDC Suite Advanced". This course shows what modifications you can perform on the pre-defined objects and texts, and how to perform them.



DDC Suite 2.7 / Functions

BACnet

Binary objects have 2 status texts instead of a unit. In FBoxes with only one object, you can directly parametrise texts for the statuses **Inactive Text** and **Active Text**.

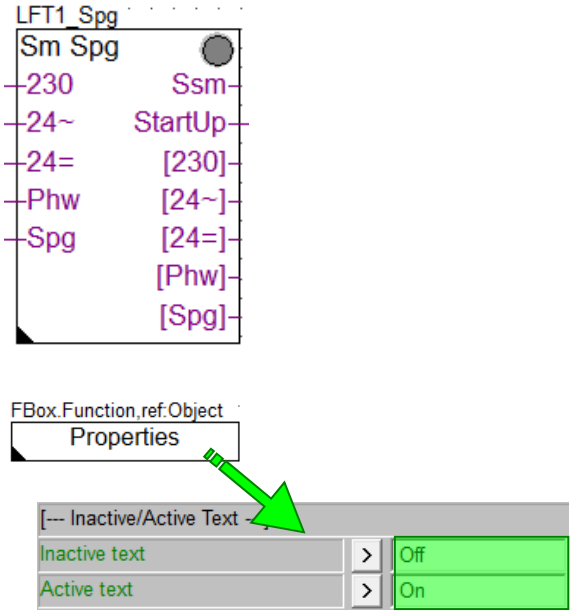


FBoxes with multiple objects do not offer this option. Here, the sum of parameters has been reduced. In most cases, these status texts are "Off" and "On", and are stored in the aforementioned file.

If you now want an object to use other status texts, adjusting the text file will lead to an unwanted result: all objects will now have the modified status texts.

In this case, we need a specialised FBox that we can use to adjust the status texts for the phase monitor in the **Sm Spg** FBox.

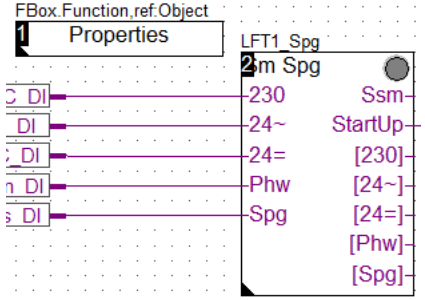
This is possible using the **Properties** FBox.



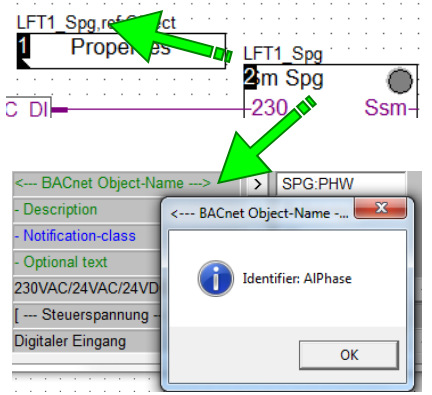
DDC Suite 2.7 / Functions

BACnet

We first have to ensure that the **Properties** FBox is placed in front of the FBox that is to use the alternative status texts.

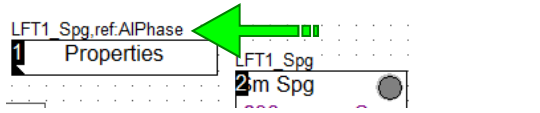


Next, the name of the Properties FBox must match that of the target FBox (**Sm Spg**): **LFT1_Spg**

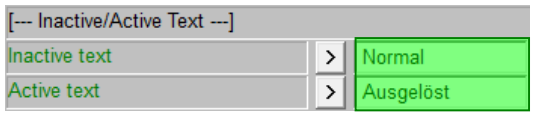


As the FBox creates (up to) 5 objects, the object that should use the new status texts should be identified. To do this, click on the property **<--- BACnet Object-Name --->** of the phase monitor. In the following dialogue, the internal object designator **AIPhase**.

This object designator must now be entered as reference in the Properties FBox. This means it is exactly defined which object should be adjusted in which FBox.



All that remains is to define the new status texts.



DDC Suite 2.7 / Functions

BACnet

If the BACnet objects are rudimentarily parametrised, i.e.:

- **Notification class = -1** (= no alarming)
- **Trendlog = No** (=no trend data)

the BACnet configuration is quick and easy to implement. You can of course experiment with it a bit.

A BACnet browser is provided with PG5 2.2 (must be installed separately). Without a licence, you can still use the network to read the BACnet Device = PCD, and to test whether the parametrised objects are present and whether value changes have been sent automatically by the PCD.

Familiarise yourself with this. When switching to a BMS, this may not work once or twice as there are different implementation versions. If you can see everything with the BACnet browser, you have done everything right.

In reality, it is unfortunately unlikely that we will have such projects, as BACnet has many integrated functions and mechanisms.

5. BACnet - Alarming

SAIA BURGESS CONTROLS

DDC Suite 2.7 / Functions

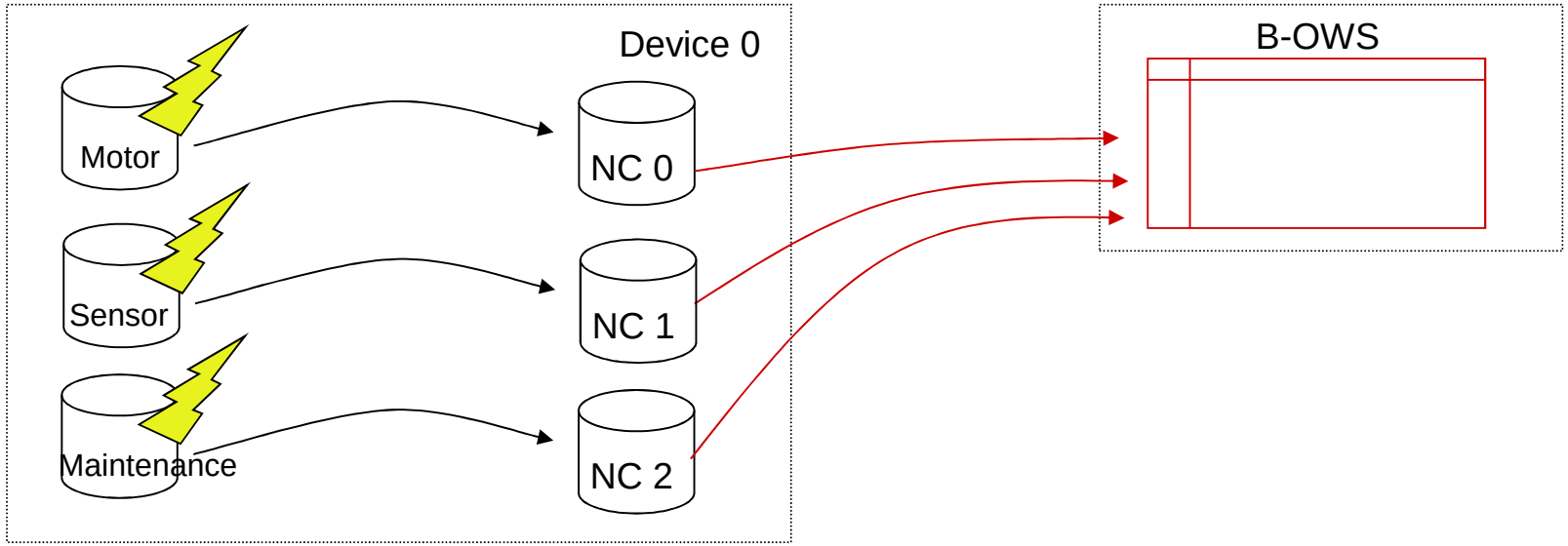
BACnet - Alarming

Alarming in BACnet functions via **Notification classes**. A notification class (NC) is an alarm group.

A BACnet object that represents an alarm, e.g. motor fault, must be allocated to an NC. The object informs the NC of its status. Only the NC itself then sends an alarm message to a receiver, the BMS.

All alarms administered via an NC therefore have the same messaging behaviour, such as priority, acknowledgement, whether they are sent as an alarm or just as an event, etc.

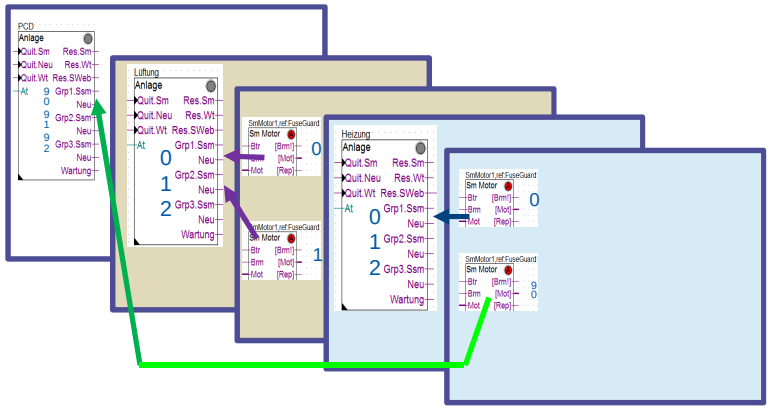
We cannot see in Fupla whether an NC has an active alarm or to which system it belongs, however.



DDC Suite 2.7 / Functions

BACnet - Alarming

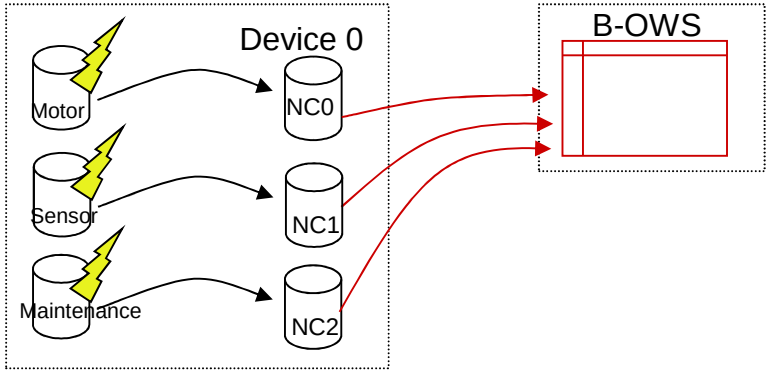
As we have seen in alarming, we use the set NC in Fupla with the **Anlagenalarm** FBox (system alarm) to automatically detect whether an alarm is pending and the system to which it belongs.



We therefore use the **Notification class** parameter in BACnet and in Fupla, giving us double the benefit.

If BACnet is not used, we can still use the NC parameter in the alarm FBoxes and can name these as we wish.

If we use BACnet in the PCD, the NCs must be numbered according to the specifications of the planner/end customer. The correct NC must then be allocated in the **Anlagenalarm** FBox (system alarm).



But: Fupla gives us no information from the BACnet NC, and this cannot be influenced.

DDC Suite 2.7 / Functions

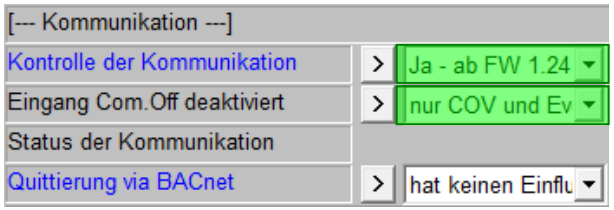
BACnet - Alarming

Customers often want the alarm messages sent automatically by the PCD as a BACnet device to be suppressed during maintenance work.

This can be activated via the parameter **Kontrolle der Kommunikation** (communication control).

-**No** → The PCD always responds to queries, value changes or alarm/event messages are also automatically sent (always unlimited communication)

-**Yes** → The PCD can suppress the sending of responses or the automatic transmission of value changes and alarm/event messages. Depending on the setting in the parameter **Eingang Com.Off deaktiviert** (Input Com.Off deactivated):



- **Komplette Kommunikation** (complete communication): The PCD completely deactivates the BACnet communication and does not respond to any queries

- **Nur COV und Events** (only COV and events): The PCD always responds to queries, the automatic transmission of value changes and alarm/event messages is suppressed

The communication can be suppressed if required via the input **Com.Off**. The alarms in the FBoxes continue to be active however, meaning that maintenance works can be performed extensively without sending unnecessary alarm messages to the BMS.

DDC Suite 2.7 / Functions

BACnet - Alarming

The alarms that run on a BACnet BMS can of course be acknowledged there. In BACnet, this acknowledgement is fed back to the BACnet device = PCD (depending on configuration).

Occasionally, the end customer want to use the acknowledgement not only to register the alarm in the alarm list of the BMS, but also to reset the alarm in the PCD.

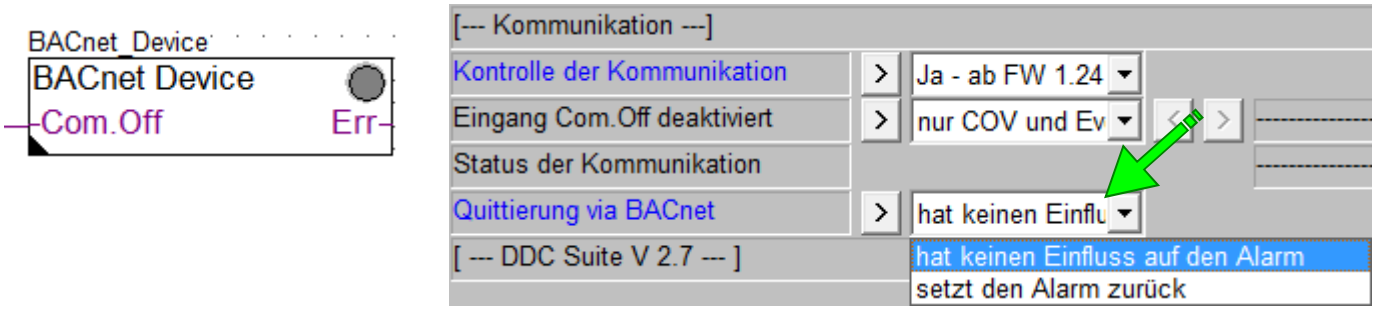
This behaviour can be set in the **BACnet Device** in exactly the same way as the **Alarm Hdr** FBox. Both work independently of each other and have no feedback.

If the alarm acknowledgement is fed back to the FBox, the FBox will automatically reset the alarm if possible. This would be the case if:

-the input status is already the normal status

-as soon as the input status reaches the normal status

-immediately in time-delayed alarms (operating feedback absent)



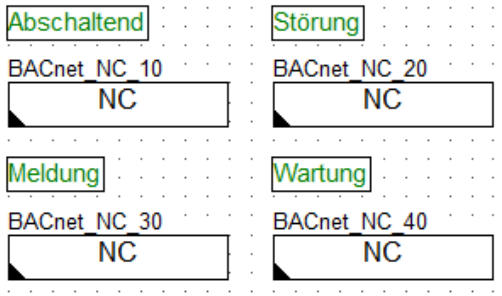
DDC Suite 2.7 / Functions

BACnet - Alarming

Now we come to the **Notification classes** themselves. An **NC** FBox must be positioned for each **NC** specified by the planner/end customer.

In this example, we have set up 4 NCs. The names of the FBoxes are only displayed to aid clarity. As you can see, we have numbered the NCs as we did a few chapters ago.

Although these NCs are set as default in the FBoxes, they do not correspond with the specifications in most cases.



The parameters must then be defined in the properties according to the specification. The following must be defined, for example:

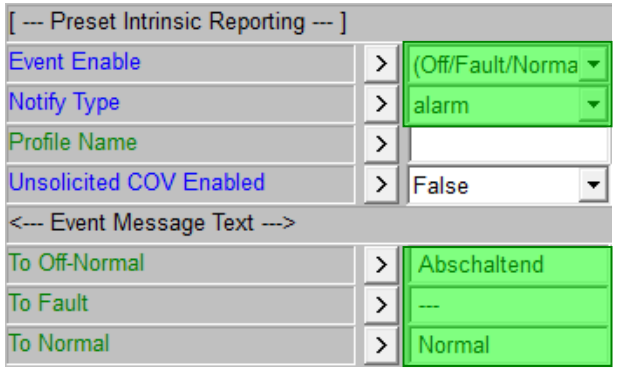
- **Object name&Description** → The AKS is not used here!
- **Notification class** → The number of the NC
- **Priority** → For the message "Comes", "Goes" and "Acknowledged"
- **Ack Required** → Which of the aforementioned messages must be acknowledged

DDC Suite 2.7 / Functions

BACnet - Alarming

All alarms or events that are notified via an NC should usually have the same construction of the message text.

To stop you having to parametrise this for every alarm individually, the **NC** FBox contains these parameters as a default setting.



All alarm FBoxes that are then connected to this NC will use these settings automatically. These are:

-**Event Enable** → Determines which of the events "Comes", "Fault" or "Goes" should be sent

-**Notify Type** → Send as "Alarm", "Event" or only as "Message with confirmation"

-Group **Event Message Text** → The texts of the respective status. You can also incorporate dynamic values here:

- **%D** → The description of the object
- **%N** → The object name of the object
- **%V** → The object's current value
- **%%** → If you want to show a % in the text

This would parametrise an NC, i.e. a BACnet alarm group. The settings for each NC can therefore vary greatly.

DDC Suite 2.7 / Functions

BACnet - Alarming

Back to our **Messwert** FBox (measurement value). The setting parameters contain either only a parameter

-**Notification class** → if it is a simple object, e.g. a motor fault, which is the case for most digital messages

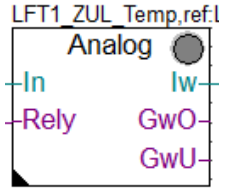
-A group **<--- Intrinsic Reporting --->** → if the object has to calculate the alarm status and additional parameters are present.

This is a measurement value that supports an alarm message for exceed and fall below.

We set the NC at 30. This means that the alarm is sent in BACnet via NC 30, which may report this to the BMS only as an event.

Beyond this, NC 30 was listed in alarm group 3 in the **Anlagenalarm** FBox (system alarm). This also registers a limit breach in group 3, and we can trigger an additional reaction in the program if required.

Limit Enable can be used to pre-select the limit value to be monitored. This could be: no monitoring, only exceeds, only falls below, or both.



DDC Suite 2.7 / Functions

BACnet - Alarming

Occasionally, an alarm may need to be reported differently to its default setting in the **NC** FBox.

The **Properties** FBox can help here too.

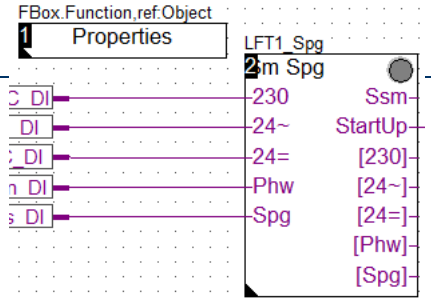
Next, the name of the Properties FBox must match that of the target FBox (**Sm Spg**): **LFT1_Spg**

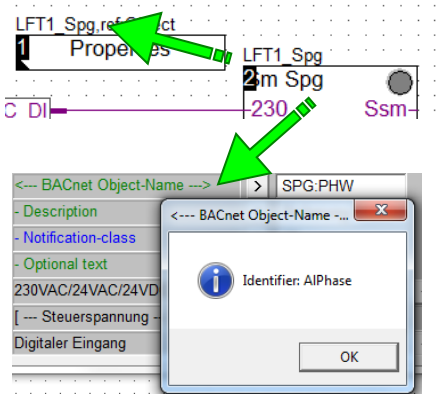
As the FBox creates (up to) 5 objects, the object that is to use the modified alarm attributes should be identified. To do this, click on the property **<--- BACnet Object-Name --->** of the phase monitor. In the following dialogue, the internal object designator **AlPhase**.

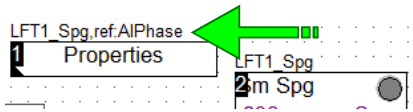
This object designator must now be entered as reference in the Properties FBox. This means it is exactly defined which object should be adjusted in which FBox.

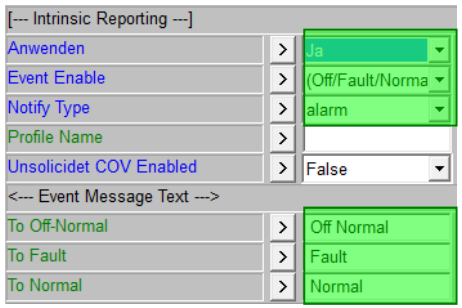
Attention! The status texts here are always used from the **Properties** FBox. Do not leave them empty!

If you also want to apply the alarm attributes from the **Properties** FBox for the referenced object, the parameter **Anwenden** (apply) should be set to **Yes**. Make sure that you then correctly set all the following parameters.









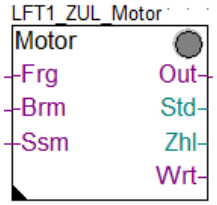
DDC Suite 2.7 / Functions

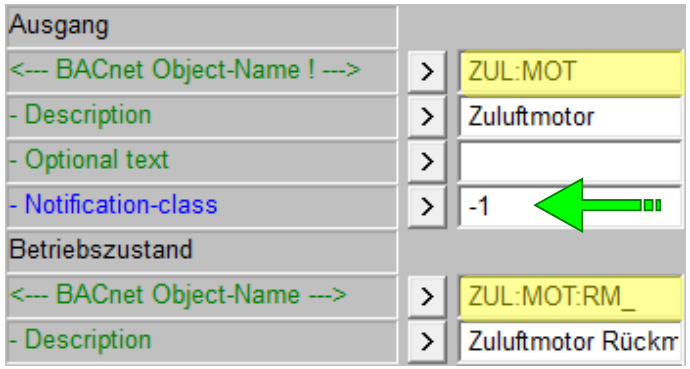
BACnet - Alarming

Some FBoxes appear to behave strangely once an NC is connected. These are above all FBoxes from the Control family.

Often, two objects are parametrised by for the output and the feedback respectively. Everything so far is correct. As this is an activation, the NC is set to -1 = not used (default setting).

A BACnet BMS will find both objects as expected.

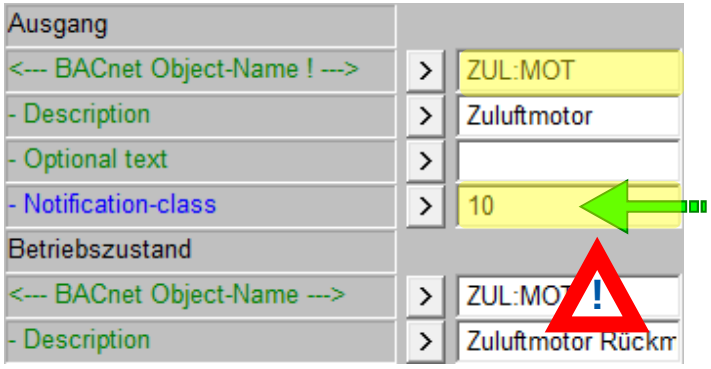




Some planners/end customers prescribe an NC for the output object. This is quick to set, but using a BACnet browser you will notice that only the output object is found.

The reason for this is simple. As soon as you create an NC in the output object, the operating feedback is automatically incorporated in the output object, as it now automatically activates feedback monitoring.

This means the feedback object is no longer necessary and the FBox will no longer create it.



DDC Suite 2.7 / Functions

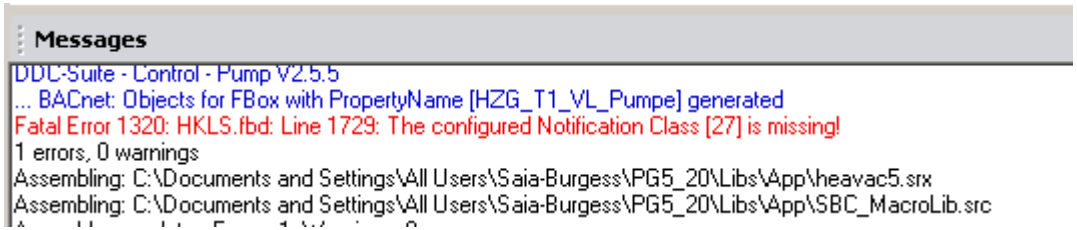
BACnet - Alarming

Familiarise yourself with the alarming (intrinsic reporting). Test the various NCs and the message suppression too, if necessary.

When switching to a BMS, this may not work once or twice as there are different implementation versions.

If you can see everything with the BACnet browser, you have done everything right.

If you have accidentally allocated an NC that does not exist to an alarm FBox (typo, NC not created), you will receive a fault message during compilation. This ensures that all alarms are matched with their corresponding alarm groups.



6. BACnet - Trending

SAIA BURGESS CONTROLS

DDC Suite 2.7 / Functions

BACnet - Trending

BACnet makes it possible to record historical data locally in the PCD. This uses a dedicated object, knows as the Trendlog object.

The amount of historical data that can be recorded using a Trendlog is limited by memory space and the number of Trendlog objects.

A long-term history over many weeks, months and years is uncommon. As a BACnet BMS is usually always present, the long-term history will run on the BMS, as it is possible here to back up the data.

This means that usually, only 200-2000 records are saved in the BACnet device = PCD per trend line. Trending can be controlled by time or events (COV = change of value). The buffer (number of records) therefore fills up at different speeds depending on the record type and recording rate.

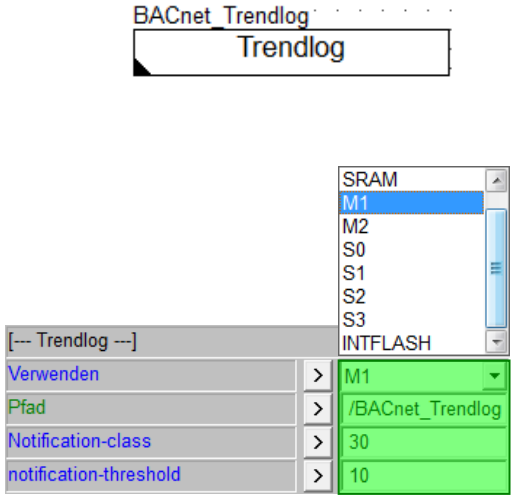
To prevent data from being lost, the BACnet device sends an event to the BMS for each Trendlog object depending on x records. This should also read the trend data to prevent data loss.

DDC Suite 2.7 / Functions

BACnet - Trending

If you want to create Trendlog objects, you first need to define a few basic settings.

You can do this using the **Trendlog** FBox. This should be placed after the **BACnet Device** FBox and after the **NC** FBoxes. The name is also only displayed here for the sake of clarity. Overview of the most important settings:



-**Verwenden** (use) → No deactivates trending completely, with no Trendlog objects created. If you want to create Trendlog objects, you only need to select a "FileSystem". Tip: Always use the BACnet module with 128 MB flash memory, and set up the corresponding slot for this.

-**Pfad** (path) → Folder in which the BACnet Trendlog objects create the historical data. Tip: Retain the default setting

-**Notification-class** → The NC with which the events "Buffer filling up" should be sent.

-**Notification-threshold** → Threshold, which triggers a message if exceeded. 10 means that a message is sent after 10 records each, even if the buffer is parametrised for 2000 records. Do not set the value too high. The BMS may be busy and read the trend data too late. This results in data loss.

DDC Suite 2.7 / Functions

BACnet - Trending

Normally, the size of the buffer is defined for each Trendlog object (=number of records).

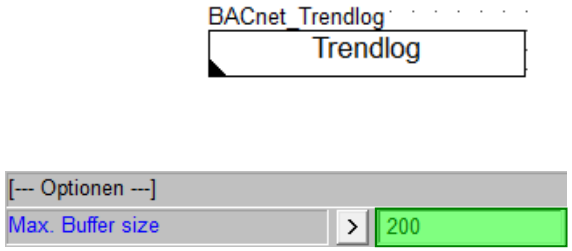
This means that a Trendlog for an on/off flap controller and recording with 100 records if the value changes is fully sufficient. If the system is switched on and off once per day, this results in 2 records. This means you will have the last 50 days in the buffer.

You can allocate a larger buffer to other Trendlogs, such as process data, and better subdivide the memory.

In rare cases however, a BMS may have problems when reading larger buffers. Instead of adjusting all Trendlog objects, the **Trendlog** FBox in the group **[--- Options ---]** offers the parameter **Max. Buffer size**.

If the **Buffer size** in a Trendlog object is smaller than the **Max. Buffer size**, the Trendlog object retains its own buffer size. If it is larger however, the Trendlog object will use the max. buffer size as its own buffer size.

It is then very easy to subsequently limit the number of maximum records.



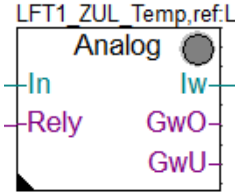
DDC Suite 2.7 / Functions

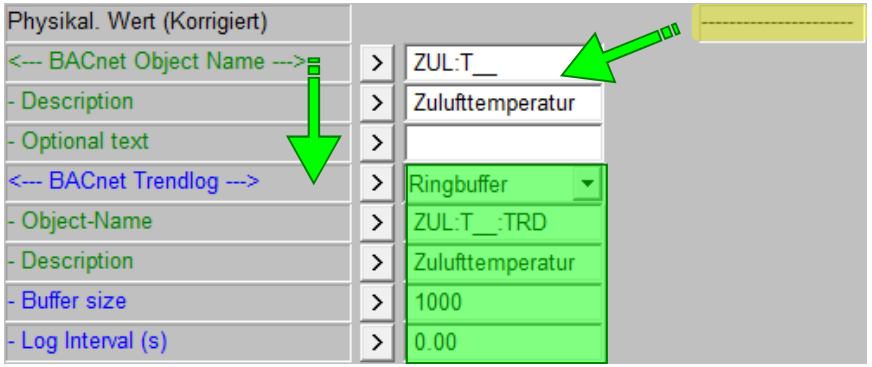
BACnet - Trending

Let's take another look at our **Messwert** FBox (measurement value) for the supply air temperature. We have already created a BACnet object for the parameter **Physikal. Wert (Korrigiert)** (physical value (corrected)).

The group **<--- BACnet Trendlog --->** comes immediately after the BACnet object. This means that the previous object before this Trendlog object is recorded. The parameters to be set are as follows:

-**BACnet Trendlog**
- **No** → No Trendlog object is created
-- **Fill&Stop** → The buffer is filled and remains. Restart via BMS
- **Ringbuffer** (ring buffer) -> Older data is overwritten (default setting)
-**Object-Name/Description** → As with all other BACnet objects
-**Buffer size** → The number of records stored in the buffer (see also **Trendlog** FBox)
-**Log Interval (s)** → 0 = no cyclical recording, only upon value change. The parameter COV of the BACnet object to be recorded is used here. A value > 0 activates cyclical recording. No COV records are registered.





DDC Suite 2.7 / Functions

BACnet - Trending

As you can see, the trend configuration in BACnet is very quick to parametrise. Don't forget the maximum number of BACnet objects (Mxx60 = max. 800 objects). This is because a Trendlog object is an independent object!

Familiarise yourself with trending (Trendlog). Also test event messages if a buffer has made enough records.

When switching to a BMS, this may not work once or twice as there are different implementation versions.

If you can see everything with the BACnet browser, you have done everything right.

7. BACnet – Trending and web

SAIA BURGESS CONTROLS

DDC Suite 2.7 / Functions

BACnet - Trending and web

Previously, the BACnet trend data could only be read and displayed by a BMS or a BACnet-compatible control panel.

If a planner/customer has requested a trend display for a local web server, this must also be programmed in Fupla. This was mostly implemented with HDLog File FBoxes.

It is now possible to use high-speed type Mxx60 controllers to also display BACnet trend data on a conventional web panel.

All that is required for this is to set the parameter **Option HDLog für Sweb** (option HDLog for Sweb) in the Trendlog FBox to **Ja – nur mit with Mxx60** (yes – only with Mxx60).

DDC Suite 2.7 / Functions

BACnet - Trending and web

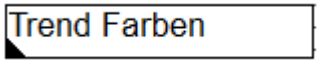
The configuration of trend data display should be as automated as possible. We already have a lot of information in the FBoxes, such as the unit and how the values should be recorded.

Ideally, you would set the colour of the trend curve and the axis values when defining the trend object.

There are already 16 pre-set colours. You can easily set these in the FBoxes with BACnet Trendlog objects. Sometimes, you may want deviating colour values.

These can be adjusted using the **Trend Farben** (trend colours) FBox. This FBox is optional, the 16 pre-set colours are also available at all times without this FBox.

The colour code must be given in R,G,B. Each value is between 0 and 255. Unfortunately, it is not possible to check syntax during entry.



[--- Farbwerte für Trendkurven ---]		
(1) Weiß	>	255,255,255
(2) Schwarz	>	0,0,0
(3) Grau	>	75,75,75
(4) Hellgrau	>	175,175,175
(5) Gelb	>	255,255,0
(6) Hellgelb	>	255,255,128
(7) Rot	>	255,0,0
(8) Hellrot	>	255,128,128
(9) Grün	>	0,64,0
(10) Hellgrün	>	0,255,0
(11) Blau	>	0,0,128
(12) Hellblau	>	0,0,255
(13) Lila	>	128,0,128
(14) Rosa	>	255,128,192
(15) Türkis	>	0,255,255
(16) Eisblau	>	0,128,255

DDC Suite 2.7 / Functions

BACnet - Trending and web

Back to our **Messwert** FBox (measurement value) for supply air temperature. In principle, the parametrisation for alarming and BACnet is complete.

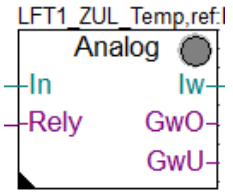
But we want a web visualisation of the BACnet Trendlog objects, which means we have to set the visualisation options. This only takes a few mouse clicks, but these settings are saved if a template is created from this system. This makes preparing the template much easier.

The unit has already been parametrised for the BACnet object. All that remains is to visualise the trend curve in a web server. Each FBox with Trendlog has a group **[--- Trend / Graph Farbe ---]** (Trend/graph colour). The following items must be set here:

-**Trend / Graph Farbe** (trend/graph colour) → Colour of the trend visualisation

-**Y-Achse Skala Wert min.** (Y axis scale value min.) → Minimum value of the Y scale (vertical axis)

-**Y-Achse Skala Wert max.** (Y axis scale value max.) → Maximum value of the Y scale (vertical axis)



[--- Definitionen ---]		
Einheit	>	Grad Celsius
COV Hysterese	>	1.0
Skalierung (Nachkomma)	>	1
[--- Trend / Graph Farbe ---]	>	(7) Rot
Y-Achse Skala Wert min.	>	40.0
Y-Achse Skala Wert max.	>	0.0

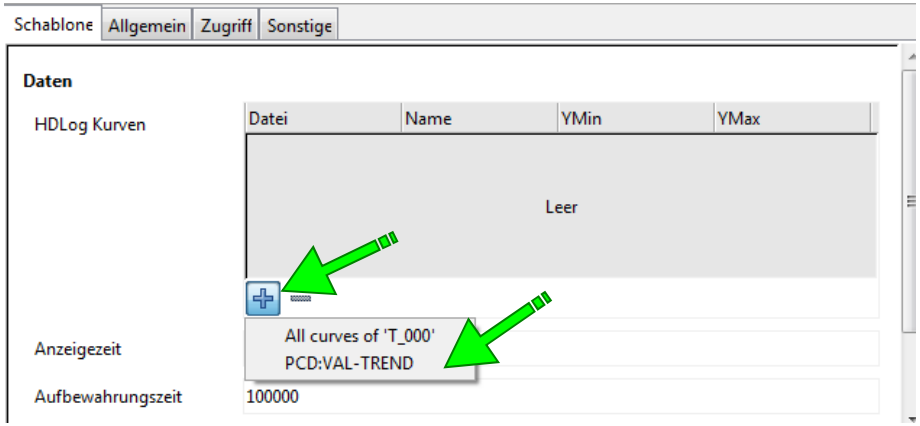
DDC Suite 2.7 / Functions

BACnet - Trending and web

Subsequently in the WebEditor project, only use the standard template to visualise HDLog (trend data).

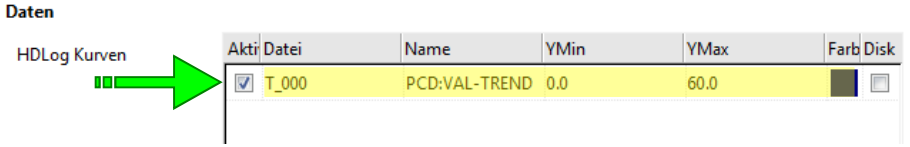
The BACnet Trendlog objects are also automatically detected by the WebEditor.

All that remains is to add the required Trendlog objects in this visualisation.



You can then find the settings that we have parametrised in the FBox in the table with the HDLog curves.

This removes the need to rework in the WebEditor project. It is therefore also possible to use Trendlog objects and the web server for commissioning.

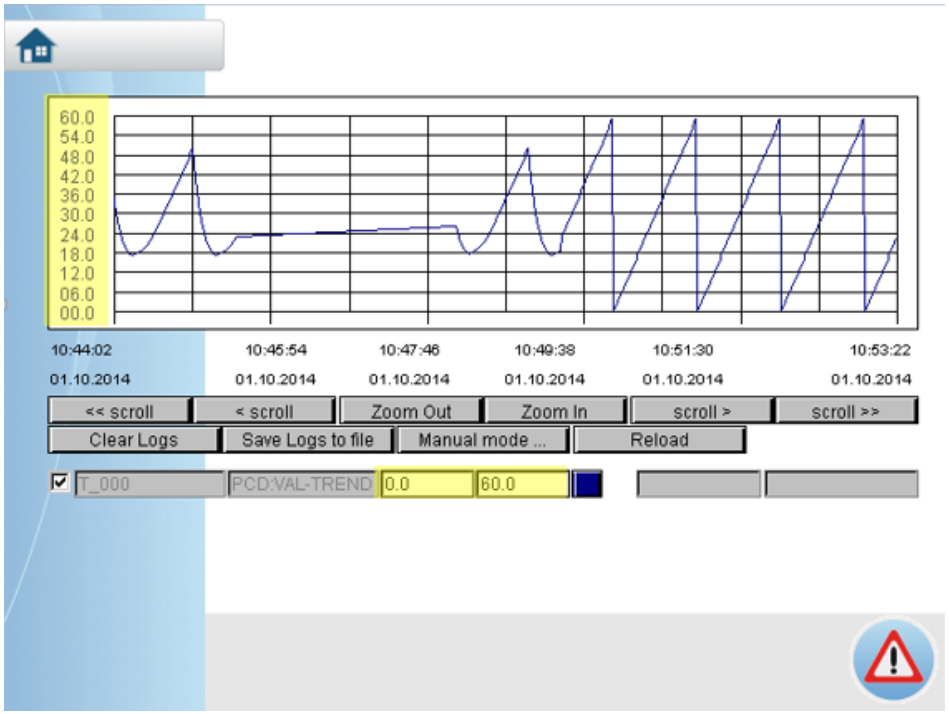


DDC Suite 2.7 / Functions

BACnet - Trending and web

Please note: NEVER mix BACnet with Trendlog and web applications on a file system. We recommend an R562 for BACnet and Trendlog, INTFLASH (xx60 systems) or SL0..SL3 for web applications.

But: You do not need a file system with PG5 2.2 for the web application if it is only used during commissioning. The WebEditor runs locally on the PC and only reads the data from the PCD. A file system is required for BACnet Trendlog, we recommend a BACnet card with 128 MB flash file system (R562).



8. BACnet – Loop and EventEnrollment

SAIA BURGESS CONTROLS

DDC Suite 2.7 / Functions

BACnet – Loop and EventEnrollment

The Loop Object is a PID controller in BACnet. In DDC Suite 2.6, we have different controller FBoxes, e.g. for a cooler, a heat recovery system or a simple controller.

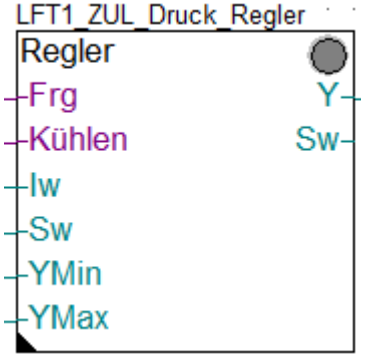
BACnet can also be activated here as with all other FBoxes.

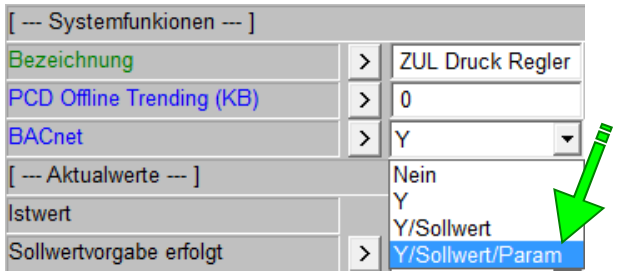
As mentioned previously, you should select the bottom option in the parameter **BACnet**, as this provides all the objects that the FBox supports.

The problem is this – we receive many individual objects, i.e. one for P, one for I, one for D, as well as signal min., signal max. and an object for the calculated signal.

But this has nothing to do with a BACnet loop (controller or control loop)!

If the planner/end customer requires a BACnet loop, the configuration must be performed differently.





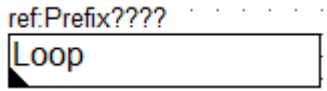
DDC Suite 2.7 / Functions

BACnet – Loop and EventEnrollment

The loop object is an independent object. It requires references to three objects for the control algorithm:

-**Sollwert** (setpoint)→ The object with the setpoint
-**Istwert**(measured value) → The object with the measured value
-**Signal** → The object that receives the calculated signal

The loop object also has certain parameters such as PID or Signal Min/Max.



We first use the Loop FBox. This is the final link in the control chain, and must be placed once after the aforementioned FBoxes.

We now have to create the connections between the Loop FBox and the other FBoxes from which the Loop FBox requires values. This is normally done via the reference.

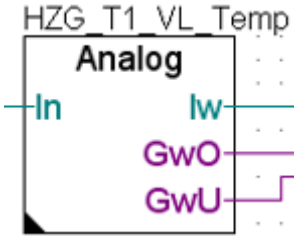
The difficulty lies in defining more than one FBox as a reference.

DDC Suite 2.7 / Functions

BACnet – Loop and EventEnrollment

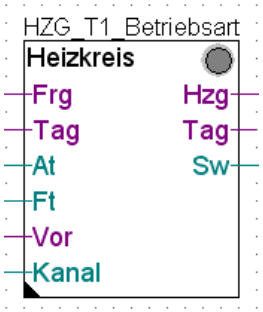
Example using a heating circuit. Let's begin by determining the FBoxes that create the respective BACnet object for setpoint, measured value and the calculated signal.

The setpoint comes from the **Messwert** FBox (measurement value) with the name **HZG_T1_VL_Temp**.



This FBox provides the measured value

The measured value comes from the **Heizkreis** FBox (heating circuit) with the name **HZG_T1_Betriebsart** (operating mode).

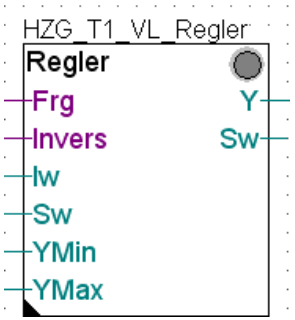


This FBox provides the setpoint

The calculated signal comes from the **Regler** FBox (controller) with the name **HZG_T1_VL_Regler**.

Note that all FBoxes have a common prefix in the FBox name: **HZG_T1**

This is a prerequisite for successfully using the **Loop** FBox.



This FBox calculated the PID parameter and writes the output

DDC Suite 2.7 / Functions

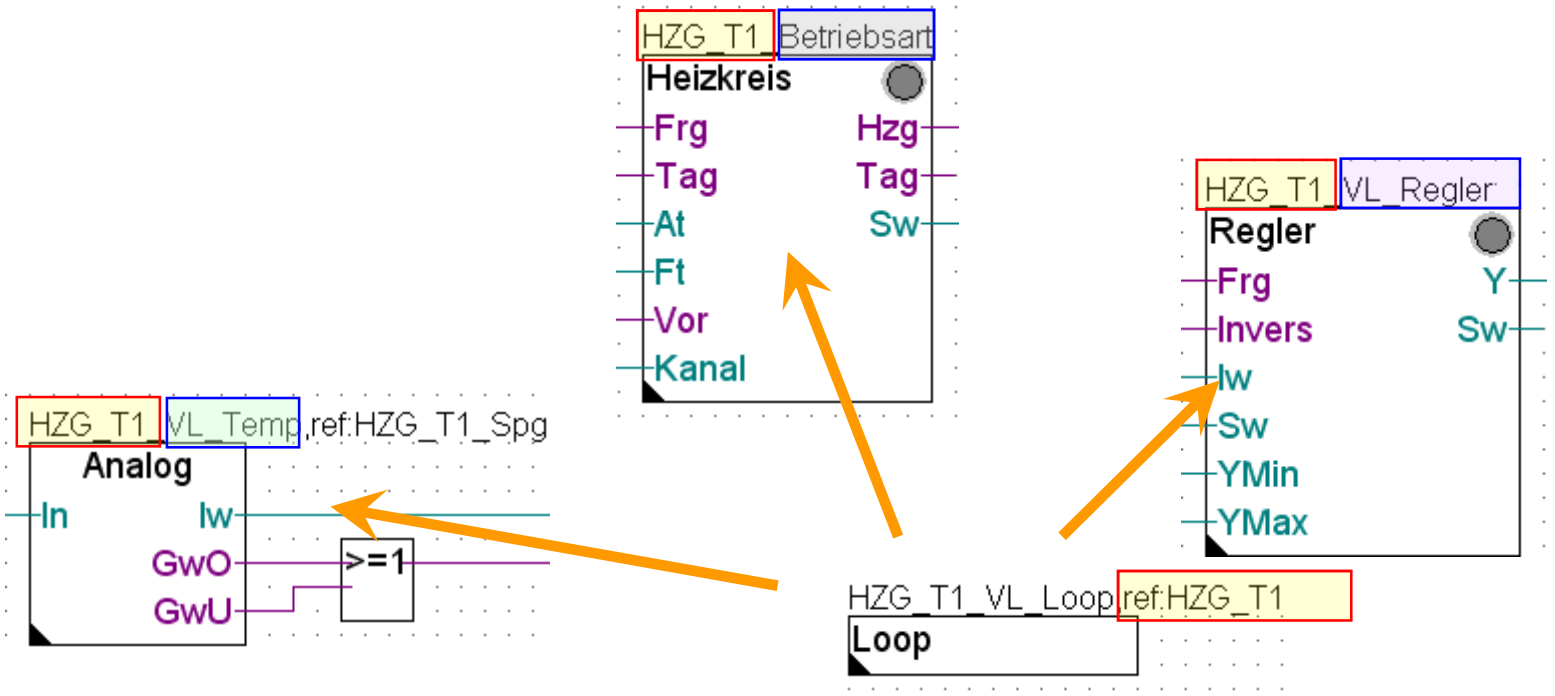
BACnet – Loop and EventEnrollment

How do we now define the reference to three different FBoxes if we can only write one reference to the FBox?

The solution is to use the longest common name as the reference for the Loop FBox.

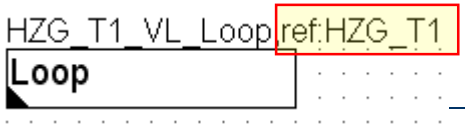
HZG_T1.

The name of the Loop FBox is only a description and has no special task in connection with the references.



DDC Suite 2.7 / Functions

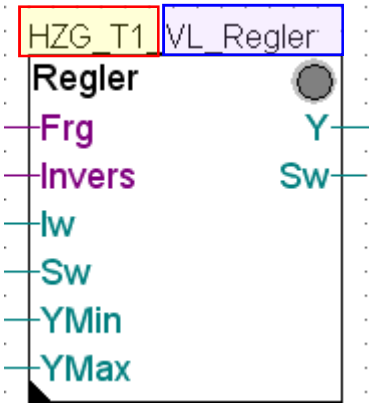
BACnet – Loop and EventEnrollment

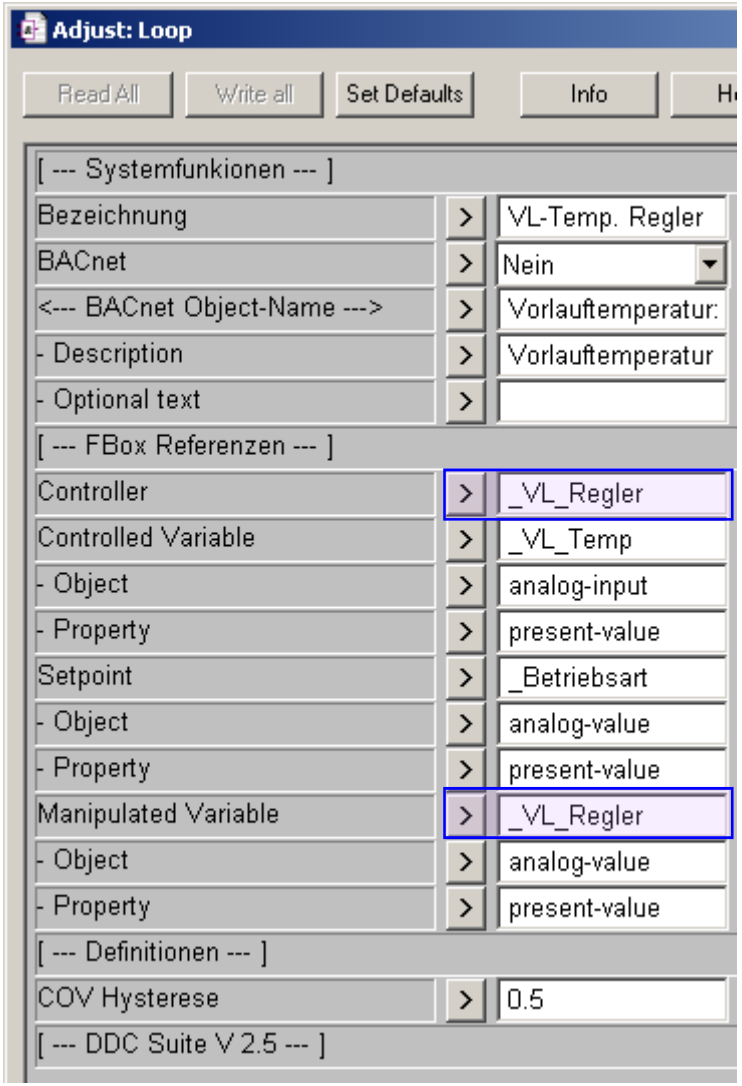


In the FBox itself, the reference to the 3 other FBoxes must be extended with the additional texts, i.e. the respective different suffix.

The first extension comes from the Regler FBox (controller) and defines the controller name: “_VL_Regler" (controller)

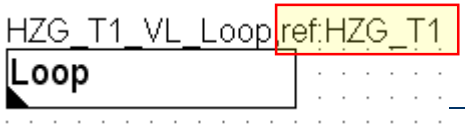
The "Loop" FBox takes all control parameters such as P, I and D from this FBox, as well as the matching units and the min. and max. values of the control signal.





DDC Suite 2.7 / Functions

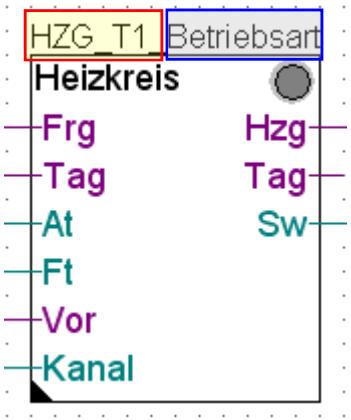
BACnet – Loop and EventEnrollment

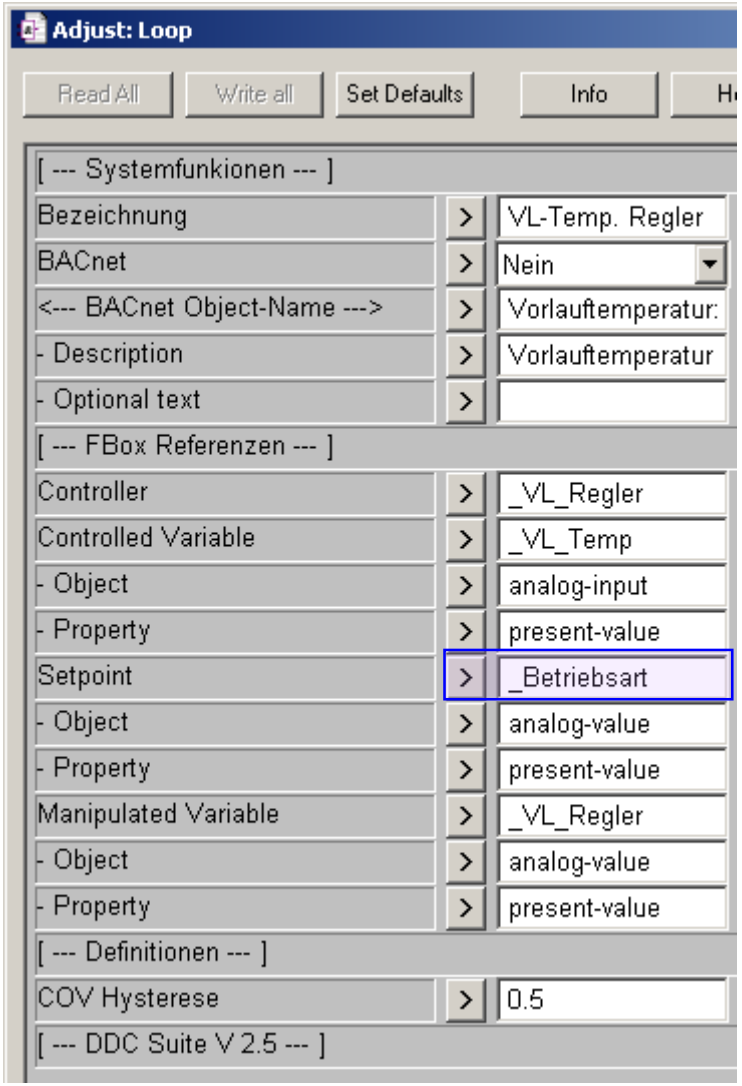


Next we enter the extension of the reference to the FBox that generates the setpoint "_Betriebsart" (operating mode).

This is normally an "analog value" object.

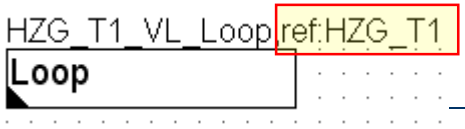
We use the "present value" to save the setpoint. This allows it to be adjusted if necessary.





DDC Suite 2.7 / Functions

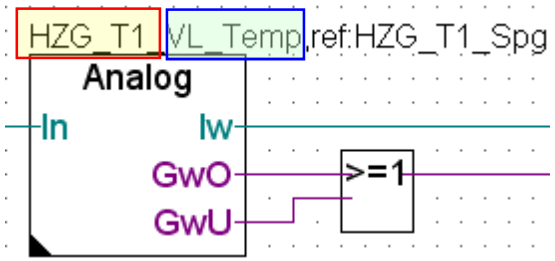
BACnet – Loop and EventEnrollment

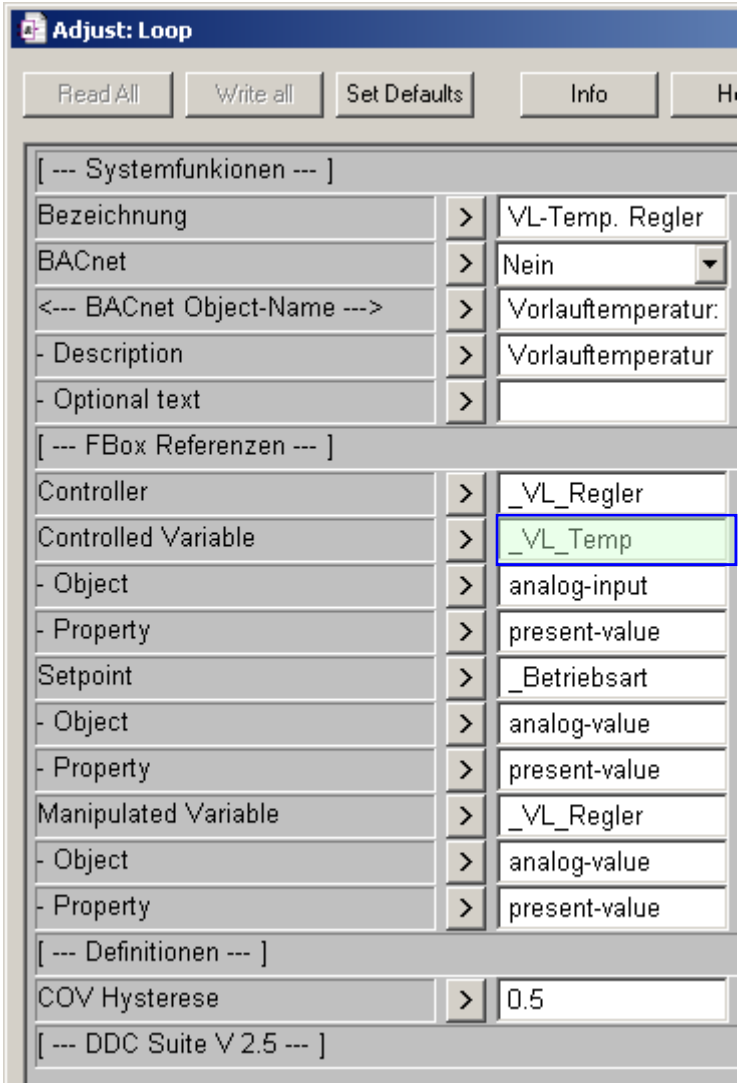


Finally, we enter the extension of the reference to the FBox that generates the measured value "_Betriebsart" (operating mode).

This is normally an "analog value" object.

We use the "present value" to save the setpoint. This allows it to be adjusted if necessary.

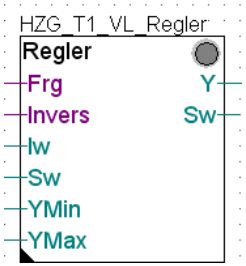




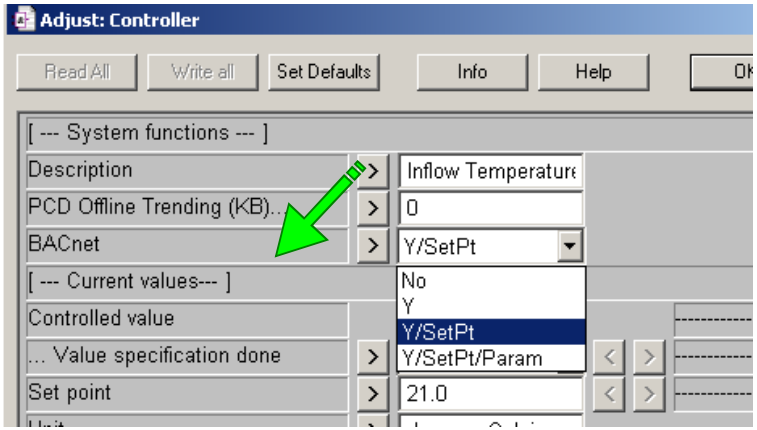
DDC Suite 2.7 / Functions

BACnet – Loop and EventEnrollment

As with the FBoxes from the control family, in which the feedback object was not created despite the definition, you may find that the loop object is not created.



This is caused by the parameter **BACnet** in the Regler FBox (controller)



- **No** → BACnet is generally not supported in the Controller FBox, meaning that the loop object cannot function

- **Y** > The object for the calculated signal is created, the loop object can be referenced to "Controller" (for the PID/min/max parameter and "Manipulated Variable" (calculated signal)

- **Y/SetPt** → As with option **Y**, although an additional object is created for the set point. The loop object can then be referenced to "setpoint"

- **Y/SetPt/Param** → The loop object is not created, as the PID/min/max values are created as a separate object.

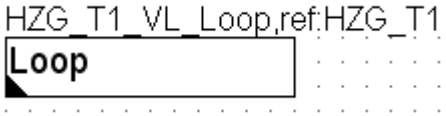
DDC Suite 2.7 / Functions

BACnet – Loop and EventEnrollment

From a technical perspective, the BACnet loop object is simply a shell. The PID is still calculated in the **Regler** FBox (controller).

This is permitted, as the BACnet standard prescribes only the parameters the loop object must support.

The control algorithm can be implemented by the manufacturer, depending on requirements.



DDC Suite 2.7 / Functions

BACnet – Loop and EventEnrollment

Let's go back to our **Messwert** FBox (measurement value) for the supply air temperature.

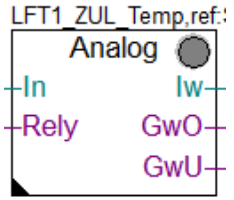
We have already parametrised so many of the functions in the FBox that it would seem we have all of the customer's wishes covered.

Almost! That's because this FBox and the resulting object can only generate 2 limit values or limit value alarms.

Some applications require a kind of pre-alarm, however, before the actual, critical limit value is reached.

This is a typical H-H-L-L limit value monitoring with 4 values.

This is the FBox with its abilities at the end. We also do not require the H-H-L-L function very much, so we can save some resources.



[--- Grenzwerte ---]		
Hysterese	>	2.0
Verzögerung	>	10
Grenzwert überschreiten	>	100.0
... Meldung		
... Alarmtext	>	Limit High
Grenzwert unterschreiten	>	0.0
... Meldung		
... Alarmtext	>	Limit Low
<--- Intrinsic Reporting --->		
- Notification-class	>	20
- Limit Enable	>	(Low/High)

DDC Suite 2.7 / Functions

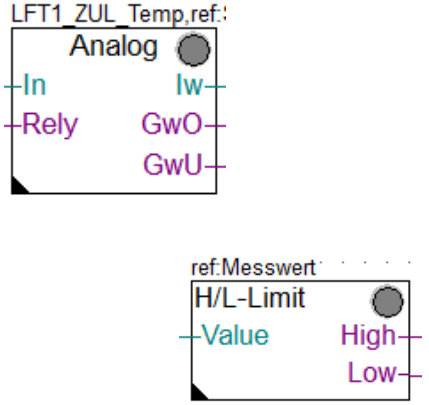
BACnet – Loop and EventEnrollment

BACnet includes an EventEnrollment object for such cases. This is an independent object, which is capable of monitoring other objects.

An EventEnrollment is like a kind of rucksack strapped to another object.

There is an **EventEnrollment** FBox for this, too. This must be placed after the FBox that creates the object to be monitored.

Currently, it is only possible to connect the **Messwert** FBox (measurement value) with the **EventEnrollment** FBox.



DDC Suite 2.7 / Functions

BACnet – Loop and EventEnrollment

First, we have to set the reference of the **EventEnrollment** FBox to the name of the **Messwert** FBox (measurement value) to be monitored. In this example, the **LFT1_ZUL_Temp** (supply air temperature).

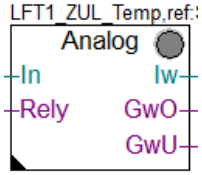
As before, the name of the EventEnrollment FBox is displayed only for clarity.

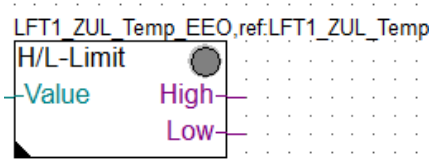
2 more limit values are provided. The BACnet object name and description are generated slightly differently to before:

AKS BACnet – ON Level 1..10 + Prefix object name + Object name

AKS BACnet – D Level 1..10 + Prefix description + Description

Note: This division into prefix and object name will be used in future FBoxes and offers more flexibility.





BACnet	>	Ja
[--- Überwachung ---]		
Grenzwert oben	>	100.0
Grenzwert unten	>	0.0
Hysterese	>	2.0
Verzögerung	>	10
Überschreitung		
Unterschreitung		
[--- Alarmtexte ---]		
Überwachung Überschreitung	>	Limit High
Überwachung Unterschreitung	>	Limit Low
[--- BACnet objects ---]		
Prefix Object-name	>	EE:
Prefix Description	>	Out of range
[--- Überwachung (EE) ---]		
- Object-mame	>	LIM
- Description	>	H/L-Limit
- Optional text	>	
- Notification-class	>	10

DDC Suite 2.7 / Functions

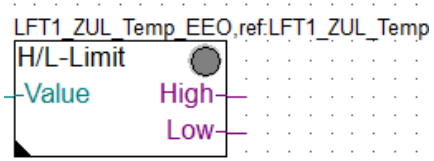
BACnet – Loop and EventEnrollment

The **EventEnrollment** FBox can also be used without BACnet.

The alarms can also be connected to a **Anlagenalarm** FBox (system alarm) using NC as normal.

The alarms can also appear in an alarm list. The parameters Designation and Alarm texts are provided with text for this purpose.

Or all functions together.



[--- Systemfunkionen ---]		
Bezeichnung	>	Sensor
PCD Alarmverwaltung (Index)...	>	-1
BACnet	>	Nein
[--- Überwachung ---]		
Grenzwert oben	>	100.0
Grenzwert unten	>	0.0
Hysterese	>	2.0
Verzögerung	>	10
Überschreitung		
Unterschreitung		
[--- Alarmtexte ---]		
Überwachung Überschreitung	>	Limit High
Überwachung Unterschreitung	>	Limit Low
[--- BACnet objects ---]		
Prefix Object-name	>	EE:
Prefix Description	>	Out of range
[--- Überwachung (EE) ---]		
- Object-mame	>	LIM
- Description	>	H/L-Limit
- Optional text	>	
- Notification-class	>	10

DDC Suite 2.7 / Functions

BACnet – Loop and EventEnrollment

When using the EventEnrollment objects, bear in mind that they are fully fledged BACnet objects.

If you parametrise a **Messwert** FBox (measurement value) with Trendlog object, and then always use an EventEnrollment FBox, you will get 3 objects per measurement value.

For 30 measurement values, this would therefore be 90 objects. Take the requirement into account when configuring the hardware. A PCD of the type Mxx60 can administer a max. of 800 objects, with the slower controllers administering a max. of 600 objects.

9. BACnet – Client

SAIA BURGESS CONTROLS

DDC Suite 2.7 / Functions

BACnet – Client

Communication between BACnet devices is also possible via BACnet communication. Here, the device to provide the data is the server (or remote device). The device that requests the data is the client.

Often, only a small amount of data is exchanged between controllers. This is usually weather data, setpoints for boiler systems or requirements.

Each BACnet server provides so-called EDE files. These contain all information concerning the objects present in the server. These EDE files can be easily imported in the device configurator, but remember to then delete all non-required objects afterwards. A further configuration is necessary in order to be able to use the data in Fupla.

Without additional intervention, we have no information about whether the communication to the server is active or interrupted.

To simplify all of these steps, DDC Suite 2.7 includes a small FBox family called **BACnet Client**.

DDC Suite 2.7 / Functions

BACnet – Client

First of all, we have two connect ourselves as a client to a BACnet server. This is done using the **BACnet Client** FBox.

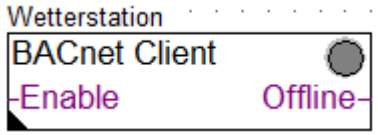
Use the input **Enable** to start or stop the connection establishment to the server. The output **Offline** becomes High if communication to the server was enabled (input Enable), but either no connection could be established or an existing communication was interrupted.

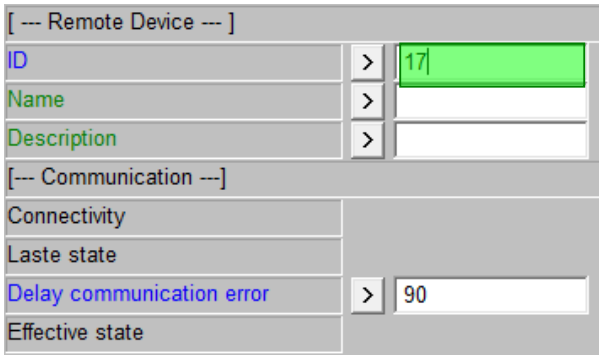
The BACnet Client FBox checks the status of the server at regular intervals. If there is no response, an interruption of communication is registered.

The name of the FBox helps connect FBoxes that the actual objects communicate.

Only be ID of the server must be set. Please refer to the EDE file for this.

The parameters **Name** and **Description** are optional. If no name is given here, the FBox will use its own name.





DDC Suite 2.7 / Functions

BACnet – Client

The values can be communicated in 3 different ways

-**Polling** → The PCD queries the value every 60 seconds at the server (default setting)

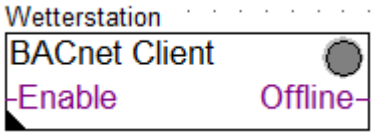
-**COV-Re/Subscription** → The PCD informs the server that a value change should automatically be sent to the client (i.e. the PCD). This reduces the communication workload (default setting)

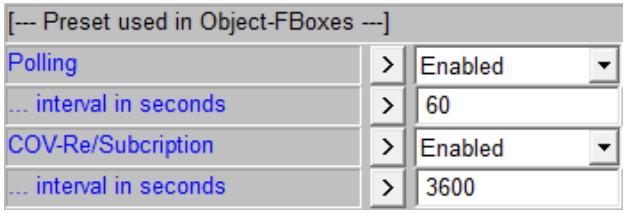
Both methods are pre-set. The PCD asks the server whether it supports COV. If this is the case, the PCD is automatically informed in the event of a value change. Polling is no longer used.

If the server does not support COV, the PCD will cyclically query the value at the server (=polling).

If neither method is activated, the PCD will listen to **Broadcast** telegrams and take the current value from these.

The type of communication is pre-set in the **BACnet Client** FBox, and is used by the object FBoxes.





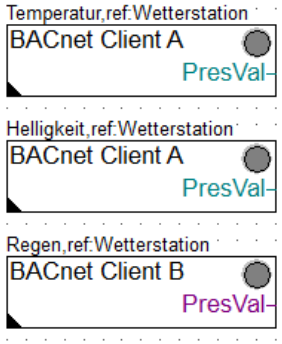
DDC Suite 2.7 / Functions

BACnet – Client

All we need to access the server data are a few object FBoxes. These are available for analogue, binary and multi-state objects.

The reference should be set to the name of the **BACnet Client** FBox, which determines the server from which the value effectively comes.

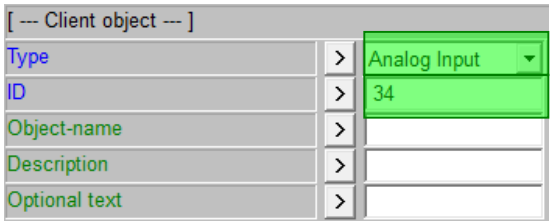
The name serves only the purposes of clarity, but should be given.



The object FBoxes can be used for input, value or output objects respectively.

In the parameter **Type**, select the object type and enter the **ID**. Please refer to the EDE file for this.

The parameters **Name** and **Description** are optional. If no name is given here, the FBox will use its own name.

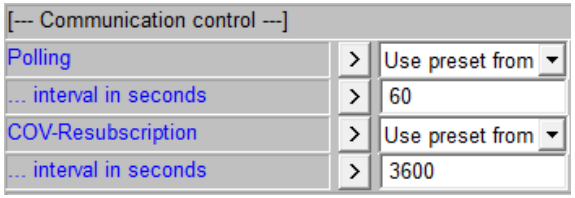


DDC Suite 2.7 / Functions

BACnet – Client

The method with which the values are communicated is defined in the **BACnet Client** FBox as a default setting.

It may be necessary however for a value to be read cyclically every 15 seconds, for example. This could be because the COV increment, i.e. the threshold when the value is resent, is set too large in the server.



Communication does not run immediately after a program download or PCD power-on. The communication may also be interrupted during operation.

In these cases, you can determine what should be output as a value at the output **PresVal**. The following options are possible:

-Output the value from the parameter **default value** only after a program download. Retain the most recently received value after power-on or in the event of a communication interruption.

-Output the value from the parameter **default value** after a program download, power-on or in the event of a communication interruption.

10. BACnet – Finalising works

SAIA BURGESS CONTROLS

DDC Suite 2.7 / Functions

BACnet – Finalising works

The BACnet objects are configured in the FBoxes. This means that you should always perform a build after adjusting parameters.

During the build, a configuration file is created that contains the entire BACnet configuration. This also includes the BACnet parameters that cannot be seen in the FBoxes. The FBoxes automatically fill many parameters.

This configuration file - DDC_BACnet.bnt – is first deleted in every build and then completely regenerated. This means it makes no sense to perform manual modifications in this file.

The same problem arises if BACnet values are read from the PCD which the customer has changed. This applies above all to switching times. These would then be lost in the event of a download.

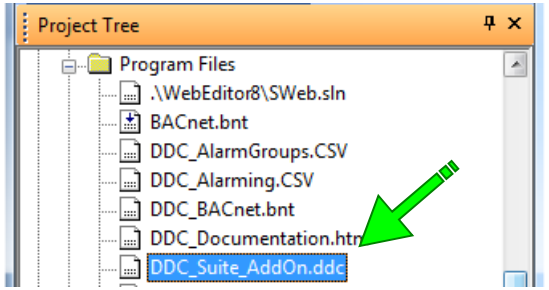
This is why the automatically generated configuration file is only a stepping stone to the final BACnet configuration.

This step is performed using the AddOn Tool of the DDC Suite. This tool is an integral component of PG5 2.2.

DDC Suite 2.7 / Functions

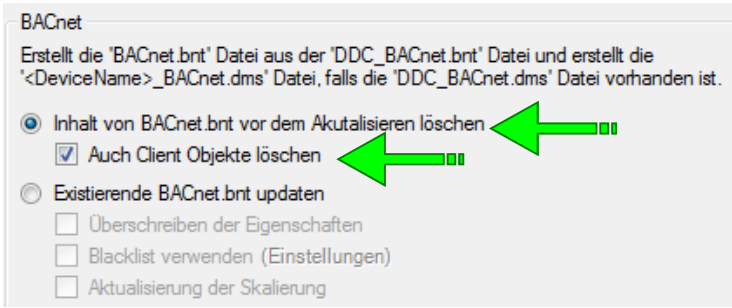
BACnet – Finalising works

Starting the **DDC Suite AddOn Tool** from the Program Files folder of the current device.



As long as you:
-have not yet switched to a BMS
-have not forwarded any EDE files
-have created BACnet client only using FBoxes

Select the option
✓Inhalt von BACnet.bnt vor dem Aktualisieren löschen (Delete content from BACnet.bnt before updating)
✓Auch Client Objekte löschen (Also delete client objects)



This ensures that the resulting BACnet configuration is tidied up and any superfluous elements are removed.

Please note: This can also cause the object ID to change.

This option must be selected the very first time.

Then start the process with **Erstellen** (create). The program is compiled. You will then be able to load the program in the PCD. BACnet is functioning.

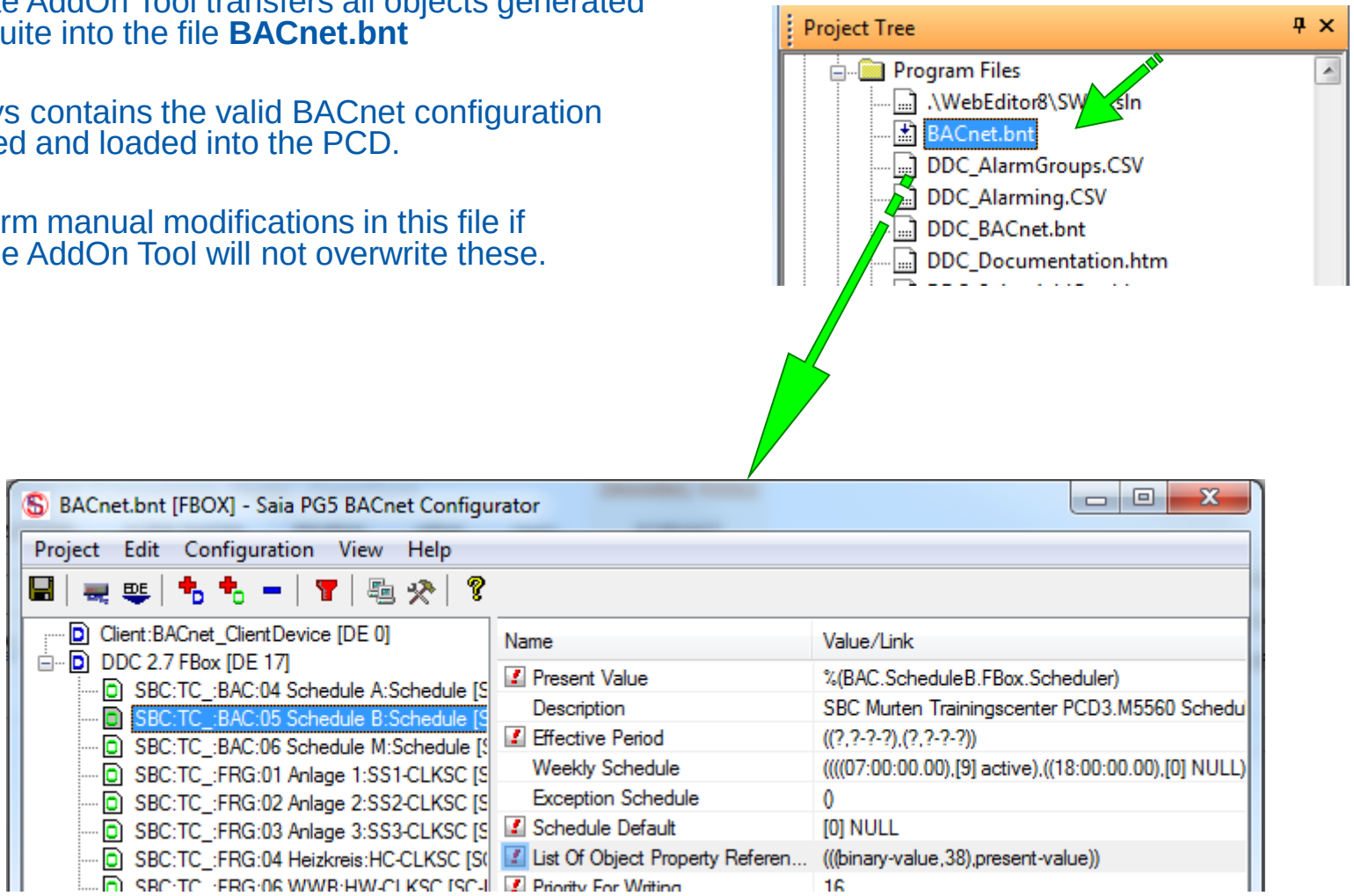
DDC Suite 2.7 / Functions

BACnet – Finalising works

The DDC Suite AddOn Tool transfers all objects generated by the DDC Suite into the file **BACnet.bnt**

This file always contains the valid BACnet configuration that is compiled and loaded into the PCD.

You can perform manual modifications in this file if necessary. The AddOn Tool will not overwrite these.



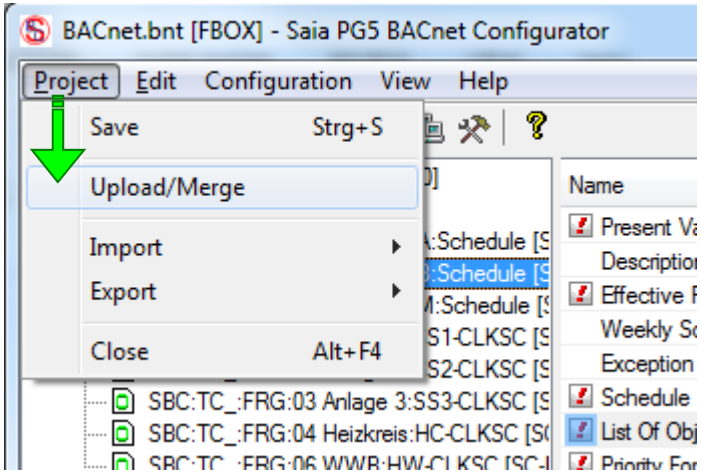
DDC Suite 2.7 / Functions

BACnet – Finalising works

Values that the operator has since changed, such as switching times, limit values, etc., must be saved prior to a download.

Failing to do this will result in the BACnet objects being loaded into the controller with their original values, with all operator settings being lost.

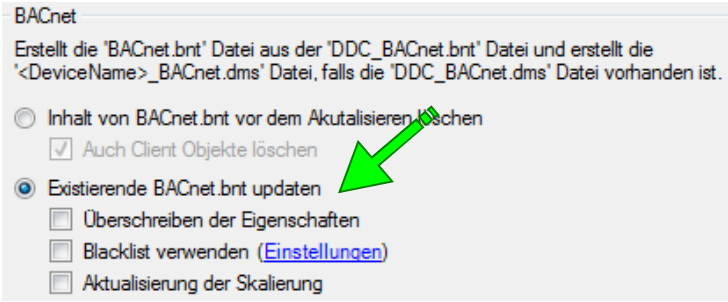
Using **Upload/Merge** in the BACnet configurator, you can read all relevant values from the PCD and save them in this configuration file.



DDC Suite 2.7 / Functions

BACnet – Finalising works

As mentioned previously, you must start the **DDC Suite AddOn Tool** after every change in Fupla that affects BACnet.



If you
-are switched to a BMS
-have forwarded any EDE files
-have manually created BACnet clients
-have performed Upload/Merge
-have "brought the system to life"

Select the option
✓Existierende BACnet.bnt updaten (Update existing BACnet.bnt)

This ensures that the resulting BACnet configuration is only extended with new BACnet objects. All existing objects are not changed with this method!

If you have deactivated a BACnet object in FBoxes, you must then manually delete it from the BACnet configuration!

DDC Suite 2.7 / Functions

BACnet – Finalising works

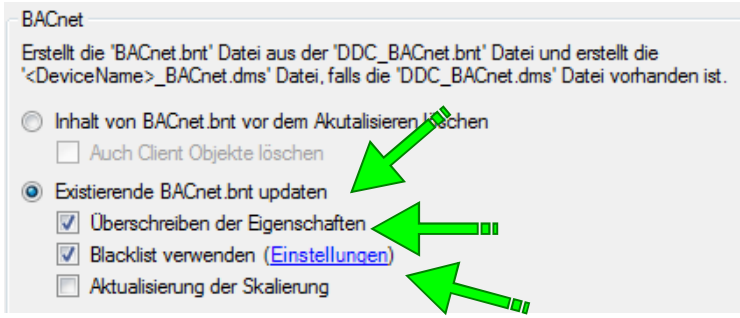
You may occasionally find that basic settings change during after commissioning.

It could be that the customer requires changes to the plant identification system or a different subdivision of the notification classes.

Adjustments like these can usually be made very quickly in the FBoxes. But how can you get these in the existing configuration? We only want to update the objects, and only the individual properties such as the notification class.

Everything else should stay as it is. This can also be done with the DDC Suite AddOn Tool. As the system is already operating, you should choose the option **Existierende BACnet.bnt updaten**
(update existing BACnet.bnt). In addition:

-**Überschreiben der Eigenschaften** (overwrite properties) → all properties are overwritten by the FBoxes in the BACnet configuration
-**Blacklist verwenden** (use blacklist) → Limits the properties to be overwritten (recommended)



DDC Suite 2.7 / Functions

BACnet – Finalising works

The dialogue for creating the **Blacklist** lists all the object types and properties.

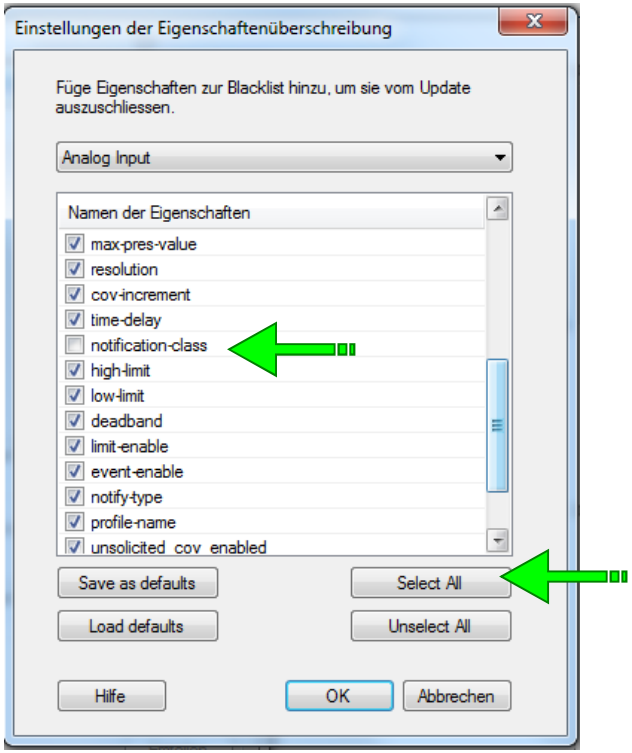
The simplest way to proceed is as follows:

-**Select All** → This excludes all object types and properties from the update (recommended)

-Afterwards, remove the property **notification-class** selectively from the exclusion list in all object types

Ticked = excluded from update

Sometimes the customer often changes their mind. You can save and read the settings so that you don't have to reconfigure this **Blacklist** every time.



DDC Suite 2.7 / Functions

BACnet – Finalising works

Please familiarise yourself with the DC Suite AddOn, as well as the Upload/Merge function of the BACnet configurator.

It is extremely important to ensure that the object ID of an active system does not change, as otherwise the allocations in the BMS (or other clients) will no longer be correct.

Also ensure that value changes made by the customer are not lost as a result of a download.