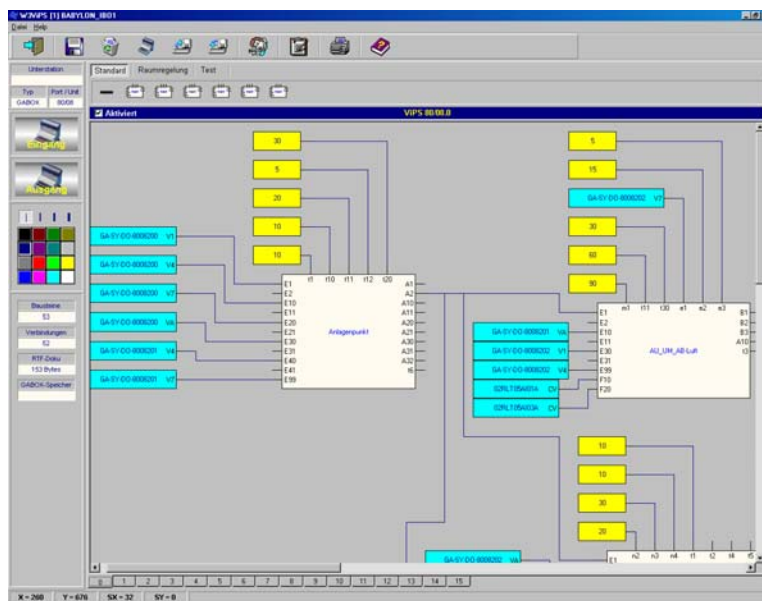


Building-Automation
Babylon Uranus – Software
ViPS



**The easiest way to construct
a complex system via drag & drop!**

Graphical Programming

- Graphical surface for the building automation system URANUS. Developed for Windows
- More than 40 ViPS-components.

Examples:

- Strobe of controlling and regulating an installation
- Controlling und regulating of flaps
- Preheater
- PID-control

Advantages:

- Minimisation of services

Juni 2004
Version 0.06

Table of contents

1	General informations	4
2	Program-Versions	14
3	Generation of the necessary download-files and data banks	15
4	ViPS-element-application.....	16
4.1	Icon-menu-bar.....	21
4.2	Function-element group menu-bar:.....	22
4.3	Menu-bar on the left screen-mask-side.....	23
4.4	Construction of a ViPS-programme	25
4.4.1	Selecting and positioning of a function-element	26
4.4.2	Delete a function-element on the work-surface.....	26
4.4.3	Calling the function-element-documentation.....	26
4.4.4	Selecting and positioning of inputs and outputs.....	27
4.4.4.1	Punktnamen/Attribut- bzw. Fixwertzuweisung eines Eingangs / Ausgangs.....	28
4.4.4.1.1	Datapoint-name-/attribute-assignment of an input / output	28
4.4.4.1.2	Fixvalue- assignment of an input / output.....	30
4.4.5	Remove an input / output from the work-surface	30
4.4.6	Drawing of connection-lines.....	30
4.4.7	Delete connectionlines.....	32
4.4.8	Moving the function-elements and connectionlines	32
4.4.9	Creating of documentations for ViPS-programmes.....	32
4.4.10	Using the debug-mode	33
4.4.11	Saving the created ViPS-programmes.....	33
4.4.12	Copying created ViPS-programmes	34
4.4.13	Export and import of ViPS-programmes	34
4.4.13.1	Import a ViPS-programme-file.....	34
4.4.13.2	Export ViPS-programmes into a file	35
4.4.14	Download of a ViPS-programme into a GA-Box	36
5	ViPS-element-definition	37
5.1	Icon-menu-bar.....	38
5.2	Function-element-list.....	39
5.3	Function-elements-detail-characteristics/diagrammes	40
5.4	Creation of a new function-element	43
5.4.1	Start a new function-element	43
5.4.2	Generation of function-element source-code.....	44
5.4.2.1	Commands for the GA-language	45
5.5	Export and import of function-elements	49
5.5.1	Import of a function-element	49
5.5.2	Export of a function-element.....	50
6	Starting up a ViPS-programme.....	51
	AUTEC Gesellschaft für Automationstechnik mbH.....	51

1 General informations

ViPS is a graphical programming-tool you can use to construct user- and regulations-programms:

Visuell

Programming-

System

The regulation-programms can be generated without any programming-knowledges via „drag-and-drop“. For this purpose the services can be reduced significantly.

For the generation of the regulation-programms/systems you can use preconfigured function-elements; these function-elements are able to do special jobs on their own (for example a double-pump-regulation etc.).

Further on own function-elements can be created to handel special jobs.

For this purpose a simple programming-language –comparable to C- can be used.

The following regulation-elements are planned for the moment: (most of them are already finished and work)

Name of the function module:

0. Installation point

0.1 Strobe of controlling and regulating an installation

Room air technique

1. Outdoor air preparation

1.1 Calculation of the enthalpy from humidity and temperature

2. Heat recovery

- 2.1 Controlling und regulating of a circulation network system
- 2.2 Controlling und regulating of controllable flaps
- 2.3 Controlling und regulating cross current heat exchanger with barrier flaps
- 2.4 Controlling und regulating heating wheel

3. Barrier flaps

- 3.1 Controlling of the flaps
- 3.2 Hygiene circuit for external air filters

4. Air heater

- 4.1 PWW – Preheater without pump(s), with antifreeze (air and /or water side)
- 4.2 PWW - Preheater with 1 – n pumps, with antifreeze (air and /or water side)
- 4.3 PWW – Secondary heater without pumps
- 4.4 PWW – Secondary heater with pumps
- 4.5 Electro air heater (staggered), with safety device
- 4.6 Electro air heater (staggered), without safety device
- 4.7 Electro air heater (continous), with safety device
- 4.8 Electro air heater (continous), without safety device

5. cooler

- 5.1 KW – Cooler without pump(s)
- 5.2 KW – Cooler with pump(s)
- 5.3 Evaporator control (2 point control)
- 5.4 Evaporator control (continous control)

6. Ventilators

- 6.1 Controlling of unregulated ventilators with 1 – n speeds
- 6.2 Controlling of regulated ventilators
- 6.3 Controlling of flue gas ventilators
- 6.4 Controlling of fast ventilation installations
- 6.5 Controlling of frequency converter with disturbance processing

7. Air humidifier

- 7.1 Air washer with pump, 2 point control, refilling and automatical water blow off
- 7.2 Air washer with pump, continuous control, refilling and automatical water blow off
- 7.3 Controlling of direct vapor humidifier, 2 point control
- 7.4 Controlling of direct vapor humidifier, continuous control

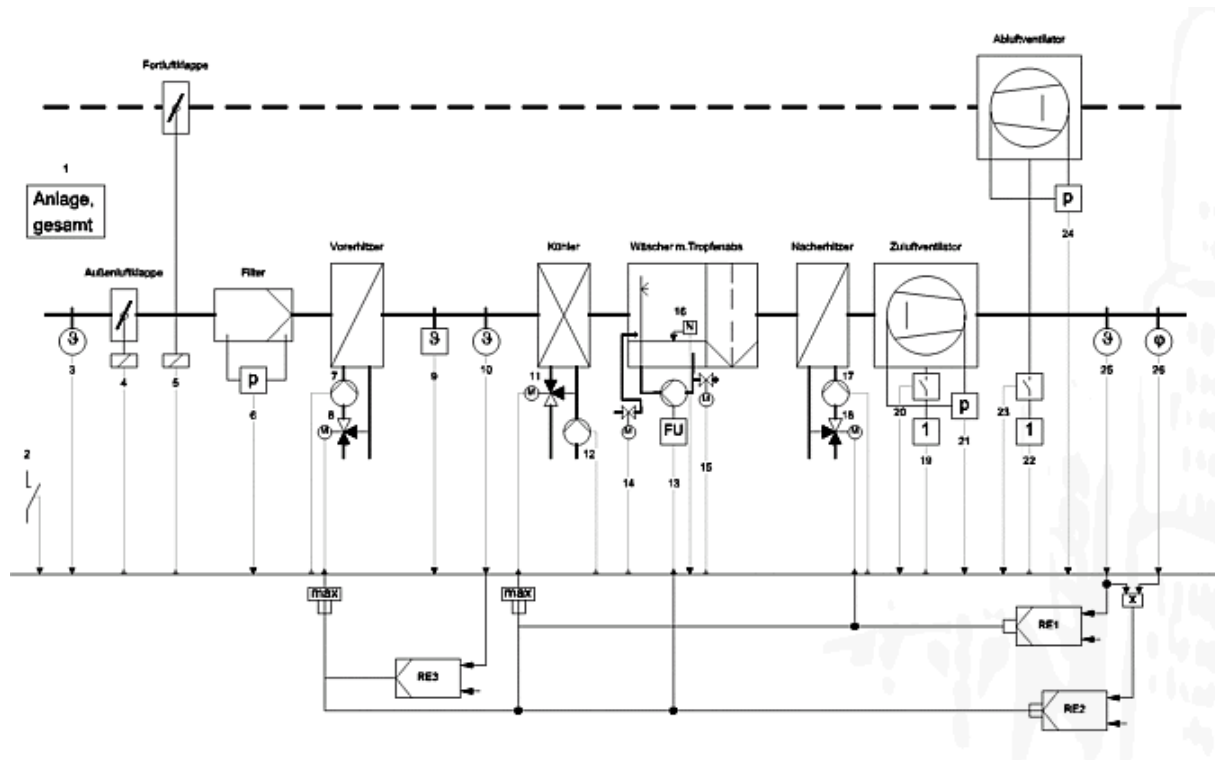
8. Motor driven fire protection / smoke extracting flaps

- 8.1 Fire protection flaps
- 8.2 Smoke extracting flaps

9. Controllers

- 9.1 Controller with one output, heating or cooling / humidifying or dehumidifying, analog outputs
- 9.2 Set-point controller with 2 sequence controllers, heating or cooling / humidifying or dehumidifying, analog outputs
- 9.3 Set-point controller with 3 sequence controllers, heating or cooling / humidifying or dehumidifying, analog outputs
- 9.4 Set-point controller with 4 sequence controllers, heating or cooling / humidifying or dehumidifying, analog outputs
- 9.5 Cascade controller, room /incoming air cascade for temperature or humidity control, analog outputs
- 9.6 Controller with one output, heating or cooling / humidifying or dehumidifying, binary outputs
- 9.7 Set-point shift in dependence on outside temperature (DIN)
- 9.8 h,x led control
- 9.9 Hysteresis with one analog input and one binary output
- 9.10 Adaptive controller
- 9.11 Calculation of dew-point temperature in dependence on the outside temperature

Scheme of an air conditional installation:



- Switching on of the installation, either by an external switch (E1),
- or by software (e.g. time switch program),
- Switch command for an outdoor air flap and a exhaust air flap,
- Antifreeze by freeze guard,
- In case of freeze alarm switching-off of ventilators, closing of the outdoor air and the exhaust air flap, opening of the preheater valve and switching-on of the preheater pump,
- Before activating of the ventilators → swilling of the pre-heater,
- Differential pressure supervision (van belt) for the outdoor air and the exhaust air ventilator,
- Fan belt supervision also possible bei a $\cos\phi$ guarder (stress guarder) or rotation guarder instead of an differential pressure supervision,
- Filter supervision (exceeding of a maximum pressure) with maintenance message,
- Regulation of the outdoor air temperature (RE1) on fix value affected in sequenz on the secondary heater and the cooler,
- Regulation of the absolute outdoor air humidity (RE2) on fix value affected in sequenz on the speeds of the washer pump, on the pre-heater and on the cooler,
- Antifreeze regulation (RE3) affecting the pre-heater in maximum selection to the set-point signal of the humidity controller,
- Heater and cooler circulation pumps switched demand dependent (after-run time and jam protection),
- Instead of admixing valves in the flow of the heating transmitter also distribution valves in the return flow can be used,
- Water refilling of the washer via magnetic valve, controlled by a level guarder,
- Desludging possibility of the guarder via magnetic valve,
- Instead of admixing valves in the flow of the heating transmitter also distribution valves in the return flow can be used,
- Total installation with start and net return programm and accumulative disturbance display.

Heating technique

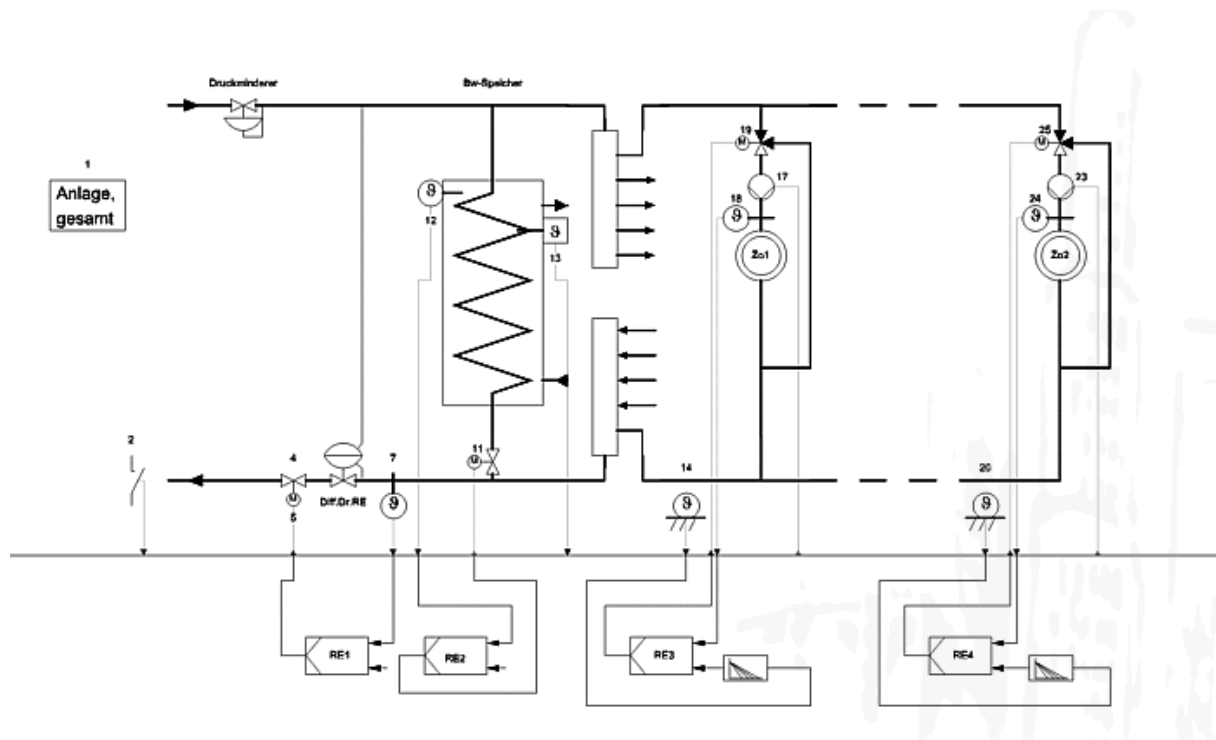
1. Heat generation with warm water boilers

- 1.1 Controlling of a warm water boiler
- 1.2 Controlling of 2-n warm water boilers
- 1.3 District heat delivery warm water (without condensate control)
- 1.4 District heat delivery hot water / vapor (with condensate control)
- 1.5 Controlling of heating pump
- 1.6 Controlling of electric water heater with legional circuit
- 1.7 Controlling of a solar installation
- 1.8 Controlling of 1- n supplying pumps

2. Controlling

- 2.1 Room temperature control with thermostat
- 2.2 Room temperature control as cascade (room/ flow temperature)
- 2.3 Weather led flow temperature control
- 2.4 Weather led flow temperature control with room temperature compensation
- 2.4 Weather led flow temperature control underfloor heating
- 2.4 Weather led flow temperature control underfloor heating with room temperature compensation
- 2.5 Adaptive controller
- 2.6 Controller for return stroke increasing of the boiler temperature
- 2.7 Storage recirculation
- 2.8 Pressure balancing

Scheme of a heating installation:



- Switching-on of the installation, either by an external switch (E1),
- or by software (e.g. time switch program),
- Individual room control via thermostate valves,
- Air-pressure reducing valve and differential pressure controller without auxiliary energy (conventionally analog),
- Return flow temperature limitation (regulation of the return flow temperature via controller RE1), to prevent the exceeding at the maximum allowed value,
- Weather led flow temperature control for heating circulations (controller RE3 and RE4), if necessary with various outside temperature / weather sensors and/or heating curve adaption,
- Overnight reduction of the flow temperature set-point values, if necessary with optimization of the switching times,
- industrial water temperature control via opening and closing of the barrier valve (2 point controller RE2),
- priority switching of the industrial water load,
- demand dependent pump controls with after-run time and jam protection for heating circuits,
- integrated speed control of the heating circuit pumps,
- error messages,
- if necessary industrial water circulation pump with time controlled switching on.
- Instead of the zone admixing valves in the flow also distribution valves in the return flow can be used.

Refrigerating technique

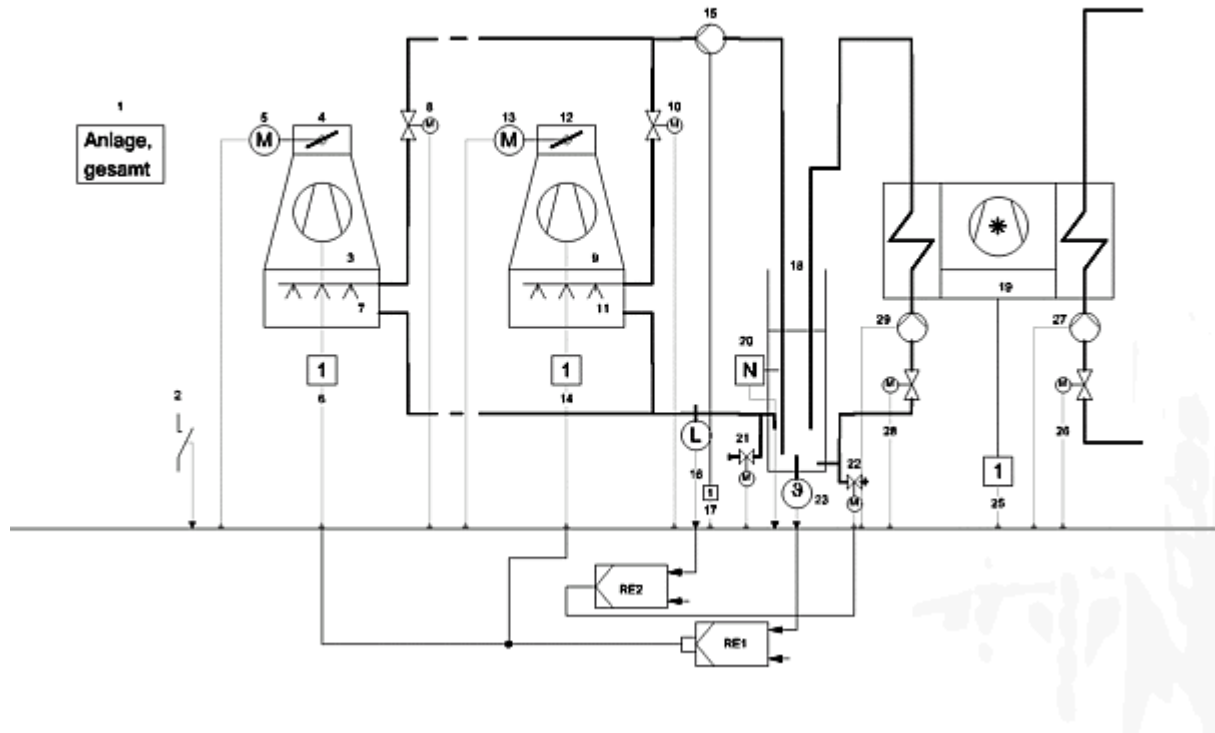
1. Refrigerating machines and re-cooler

- 1.1 Controlling of a refrigerating machine with pump(s)
- 1.2 Controlling of 2 - n refrigerating machines with pumps
- 1.3 Controlling of a cooling tower (dry) with 1 - n speeds of the ventilator
(return temperature)
- 1.4 Controlling of a cooling tower (wet) with 1 - n speeds of the ventilator
(return temperature)
- 1.5 Controlling of 1 - n condensor ventilators (return temperature)
- 1.6 Controlling of 1 - n supplying pumps
- 1.7 Free cooling

2. Control

- 1.1 Flow temperature control
- 1.2 Return temperature control
- 1.3 Pressure balancing

Scheme of a refrigerating installation:



- Switching-on of the installation, either by an external switch (E1),
- or by software (e.g. time switch program),
- Switching on of the cooling water pump for the refrigerating machine circuit during start-up of the installation,
- Regulation of the cooling water temperature in the buffer (PI controller RE1) in sequenz via the cooling water pump of the cooling tower circuit and n cooling tower ventilators (staggered),
- Sequenz switching of cooling tower pump and cooling tower ventilators in this way, that the continous output signal of RE1 switches e.g., two cooling towers in the range of 0 ... 33% the pump, in the range 33 ... 66% the 1. ventilator and in the range 66 ... 100% the 2. ventilator,
- Cooling water pumps after-run time and jam protection,
- Opening of the cooling tower flaps before activating the ventilators,
- Water lever supervision in the buffer, if necessary refilling,

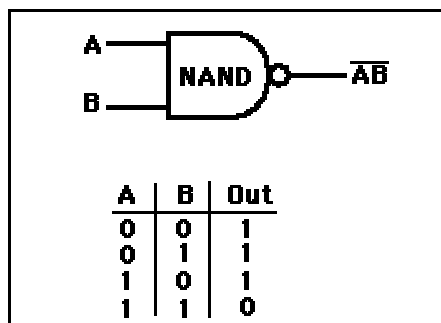
Miscellaneous

1. Selection of the leading aggregat from operating hours and error messages
2. average value of 2 – n (measurement) values
3. Minimum value of 2 – n (measurement) values
4. Maximum value of 2 – n (measurement) values
5. Efficiency optimization of 2 – n aggregats
6. Sequence circuit of 1 – n aggregats
7. Averaging of the outside temperature over the last 24 hours
8. Or – linkup
9. And – linkup
10. Not – level
11. Nor – level
12. Nand – level

$$T : M_n(X) \rightarrow C$$

$$[x_{ij}] \mapsto \sum_{i,j} T_{ij} x_{ij}$$

$$r(\varphi) = \frac{a}{\cos \varphi} \pm k$$



$$U_{eff} = \frac{\hat{U}}{\sqrt{2}}$$

$$I_{eff} = \frac{\hat{I}}{\sqrt{2}}$$

$$(r(\varphi))^2 = 2 e^2 \cos(2 \varphi)$$

The **number of regulation-elements is continuously extended**, to satisfy the demands of the market and of our partners.

In addition a **comfortable tool** is offered to construct **own elements** (see section 5).

2 Program-Versions

The documentation in hand refers to the following program-versions:

W3block	Version 1.2 (Jun 1 2004)
W3ViPS	Version 1.5 (Jun 2 2004)

3 Generation of the necessary download-files and data banks

For every automation-station XMP-GA-Box a download-file has to be generated. This file later on contains all data which are saved on the XMP-GA-Box (routines, timetables, attribute-parameters, ViPS-programmes, usw.). Therefore the download-file is also a safety-file. After a reset all data can be saved again in the XMP-GA-Box.

The file has the following name: \$\$X2bbcc.386

bb: „Mbox“-address, the XMP-GA-Box is connected to
cc: Hardware-address of the XMP-GA-Box

The file has to be generated with a size of 2048 Records (1Record = 512Byte) .

If not automatically occurred during the system-intallation, the following data banks has to be generated and activated:

\$\$BLOCK.386	Programmnummer: 150	Größe: 128 Records
\$\$BLOCKS.386	Programmnummer: 151	Größe: 128 Records
\$\$BLOCKC.386	Programmnummer: 152	Größe: 128 Records

The file \$\$BLOCK.386 contains the indices of the function-elements;
the file \$\$BLOCKS.386 contains the source-code of the function-elements
and in file \$\$BLOCKC.386 the documentations and comments for the individual function-elements are.

Further software-informations:

Necessary Babylon-programmes:

W3BLOCK for the generation of the function-elements
W3VIPS for the generation of the ViPS-programmes/systems
W3PORT for the port-konfiguration

Firmware for the GA-Box:

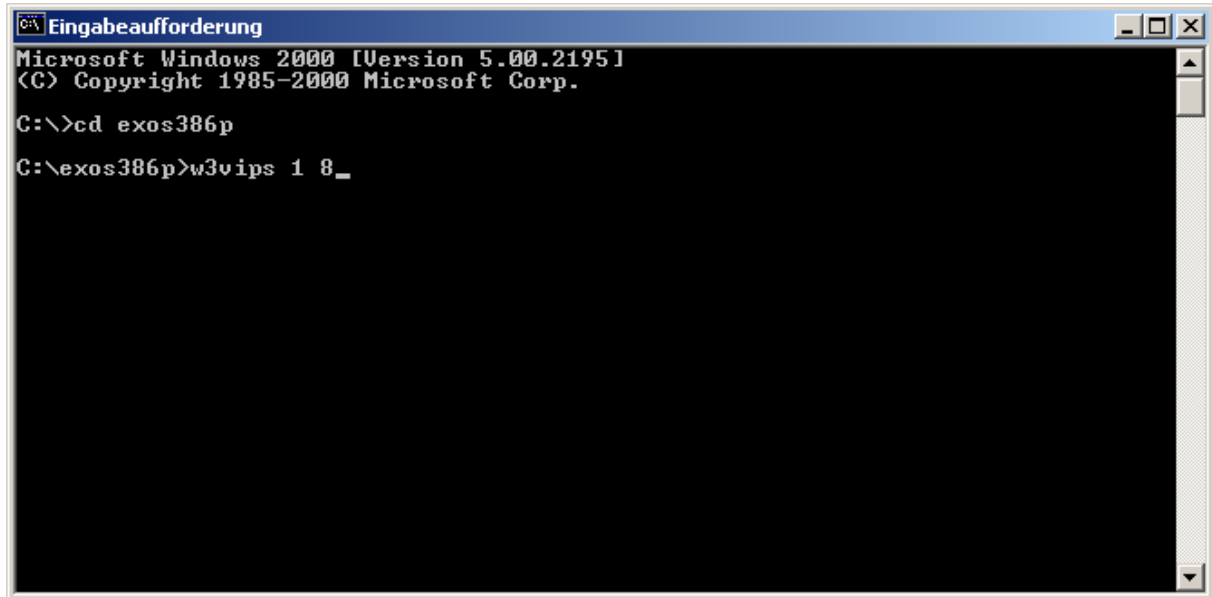
XMP2E.BIN
XMP2F.BIN

(**Remark!** First you have to save the E-segment in the GA-Box, then the F-segment!)

4 ViPS-element-application

Opening the program W3ViPS for the desired GA-Box:

Herefor you have first to start the IBO-surface; then you have to open the DOS-window and change to the register 'exos386p'. If you want to start the program for the GA-Box with address 8 –for example- you have to enter the command 'w3vips 1' with parameter 8 as shown in the following mask:

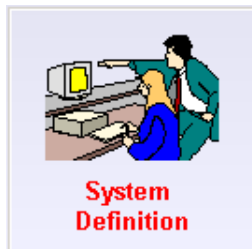


```
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\>cd exos386p
C:\exos386p>w3vips 1 8_
```

After pressing the enter-Button the mask, shown in chapter 4.1 is opened:

Attention: Not yet available!!!

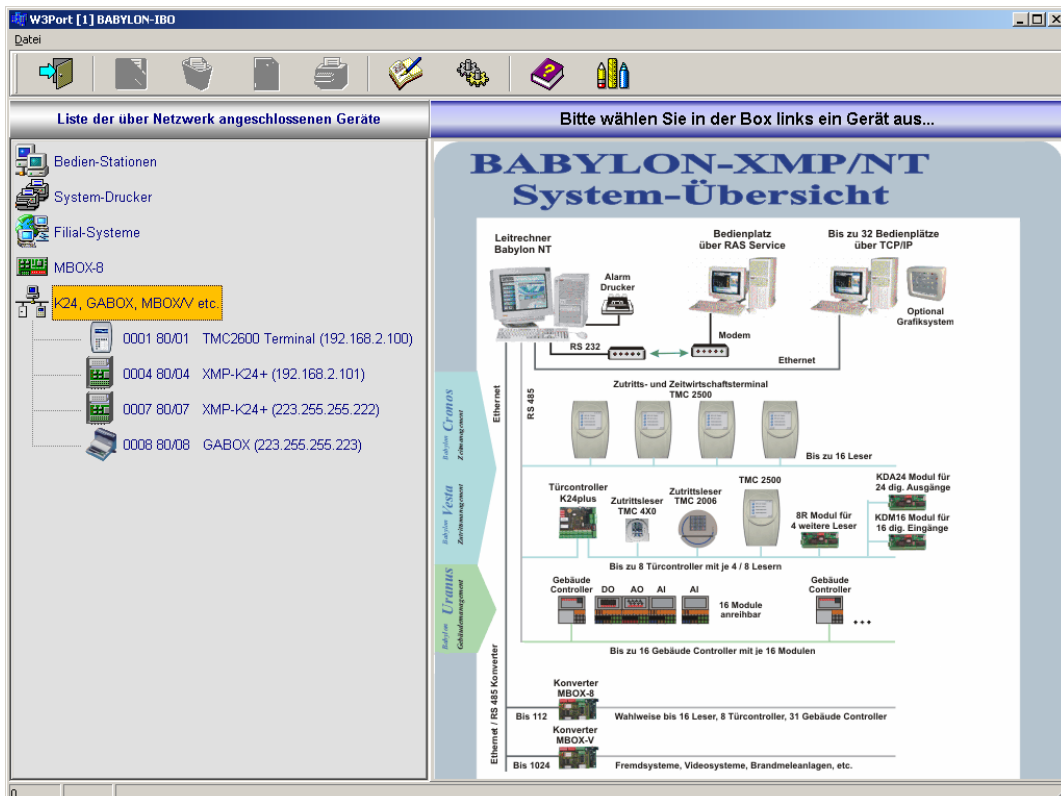


By clicking the icon „System Definition“ you are reaching the programm for choosing the GA-Box on which the ViPS-programms shall run:

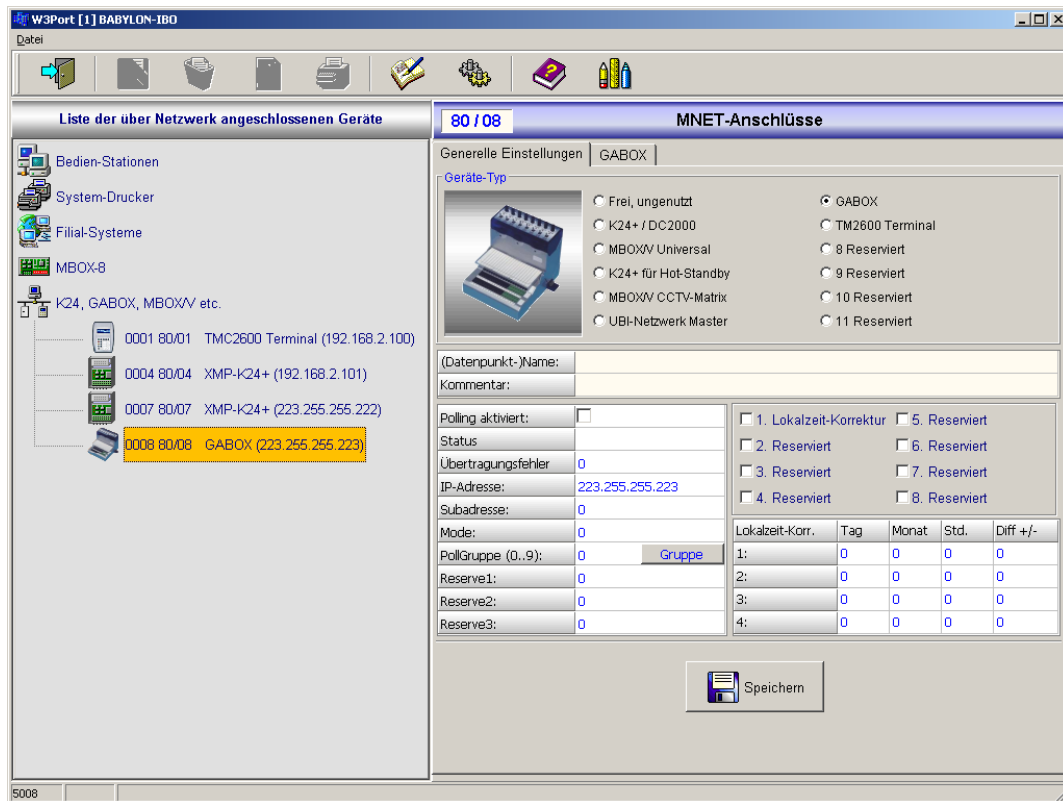


K24, GABOX, MBOXV etc.

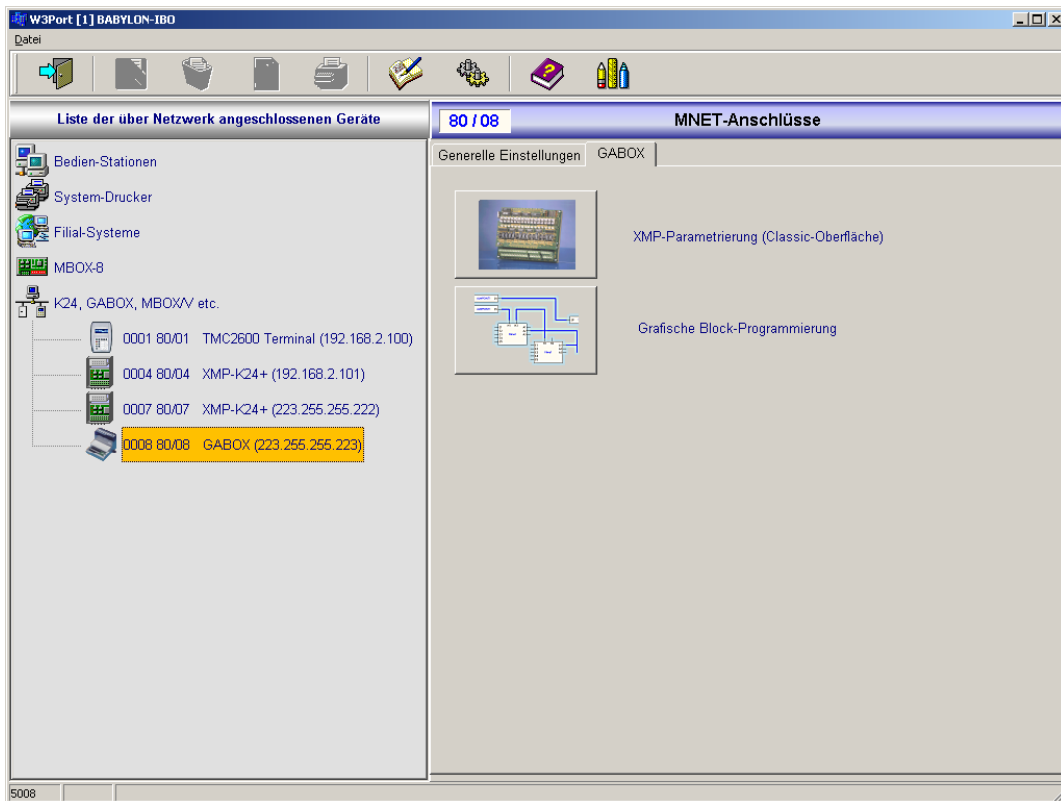
By clicking the field „K24. GABOX,MBOX/V etc.“ you get the selection of the XMP-modules connected to the system:



After clicking the desired GA-Box the scen-mask looks like followed:

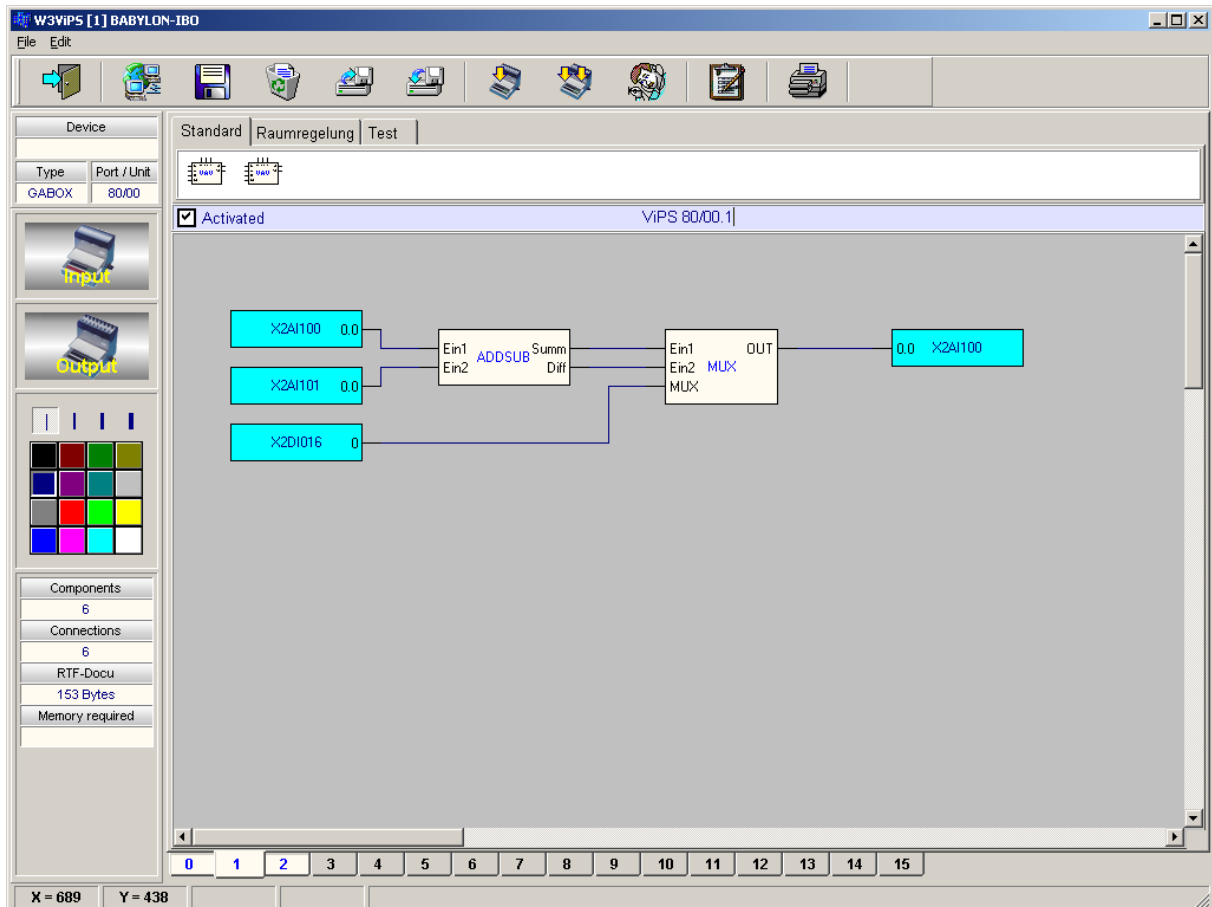


After clicking the field GABOX the following screen-mask is opened:



By clicking the button “Grafische Block-Programmierung” the programme for creating ViPS-programmes is opened:

4.1 Icon-menu-bar



Exit Program: Serves to leave the programme.



Refresh list of components: For loading all the used regulation-elements new.



Save actual ViPS-Program in database: Saving a new constructed ViPS-programme/system in a data bank.



Delete actual ViPS-Program: Deleting the actual ViPS-programme/system.



Export ViPS-Program(s): Opens the dialog for the exportation of ViPS-programmes/systems in special data files with the ending .vip .



Import ViPS-Program(s): Opens the dialog for the importation of ViPS-programmes/systems from already existing system-files with the ending .vip .



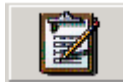
Download actual ViPS-Program into GABOX/K32: Serves to store the actual ViPS-programme/system in the desired GA-Box.



Download all 16 ViPS-Programs into GABOX/K32: Serves to store all 16 ViPS-programmes in the desired GA-Box.



Switch Debug-Mode on/off: While running the actual attribut-values are shown at the pins of the regulation-elements. The values are secondly brought up to date.



Toggle: Connection Diagram <-> documentation: Serves to swich ober between the element-diagram of the system and the documentation of the system. The documentation should explain the job, characteristic and function of the system.



Print actual ViPS-Program: Opens the printer-menu for printing an already created ViPS-programme/system.

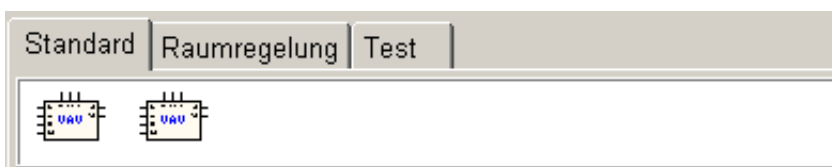


Help: Opens the help-dialogue.

The same icons and corresponding functions can be found in the top-menu by clicking on File or Edit.

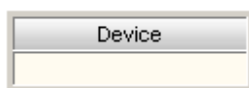


4.2 Function-element group menu-bar:

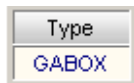


Below the icon-menü-bar you can see the names of the function-element groups. These names correspond to the term, filled in the field “Palette” while creating a function-element. By clicking on one of these function-element group-names, all function-element belonging to these group are notified below the writing.

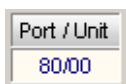
4.3 Menu-bar on the left screen-mask-side



Device: Shows the name of the substation



Type: In this field you can find the type-name of the unit selected for the ViPS-element-application.



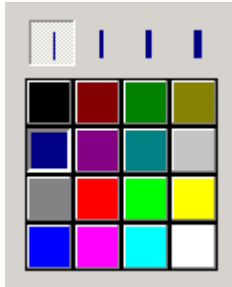
Port/Unit: In this field you can find the port-number and the unit-number (hardware address) of the unit selected for the ViPS-element-application.



Input: If an input should be positioned on the work-surface you have to click on this icon and bull it via drag-and-drop to the desired position on the work-surface.



Output: If an output should be positioned on the work-surface you have to click on this icon and bull it via drag-and-drop to the desired position on the work-surface.



The choice of colour and size of the connectionlines has to be done via the left mouse-key.



Components: Shows the number of the regulation-elements on the work-surface. The maximum number of regulation-elements per system is 100.



Connections: Shows the number of the connectionlines on the work-surface. The maximum number of connectionlines per system is 300.



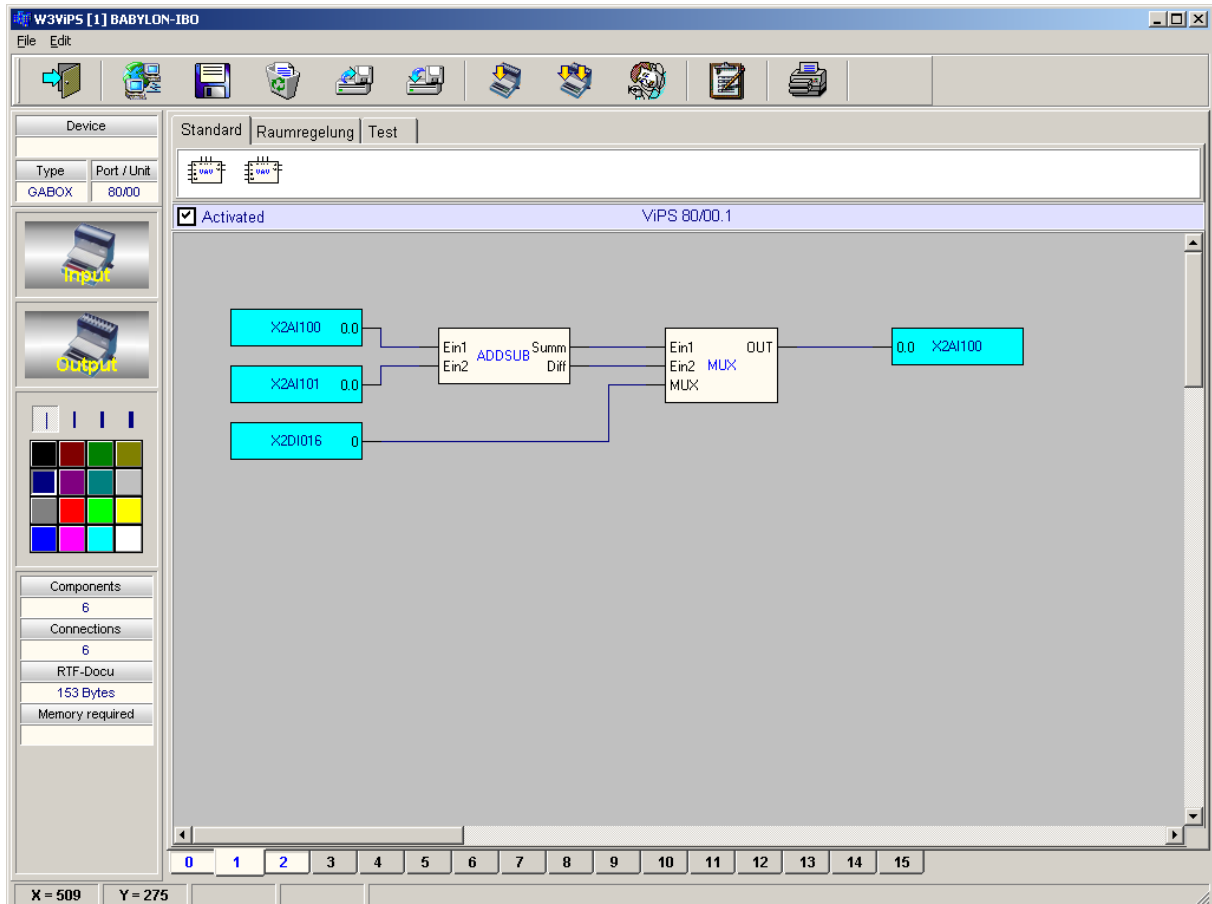
RTF-Doku: Shows the size of the RTF-document of the current shown ViPS-programme-documentation.



Memory required:

4.4 Construction of a ViPS-programme

For the construction of a ViPS-programme the right-hand work-surface has to be used:



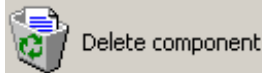
4.4.1 Selecting and positioning of a function-element

After choosing a special function-element by clicking first onto the function-element group-name and then onto the function-element itself, the function-element-diagramme can be positioned on the work-surface per drag-and-drop.

If you want to move a function-element on the work-surface, you have to bring the mouse-cursor into the middle of the function-element-diagramm (onto the name of the function-element) until the arrow-cursor changes into a hand; now its possible to move the function-element-diagramme per drag-and-drop (**Important!** Left mouse-key).

4.4.2 Delete a function-element on the work-surface

If you want to delete a function-element on the work-surface, you have to bring the mouse-cursor into the middle of the function-element-diagramm (onto the name of the function-element) until the arrow-cursor changes into a hand; by clicking the right mouse-key a menu is opened where you can chose “Delete component”



if you want to delete the function-element-diagramm from the work-surface.

4.4.3 Calling the function-element-documentation

If you want to open the documentation of a function-element on the work-surface, you have to bring the mouse-cursor into the middle of the function-element-diagramm (onto the name of the function-element) until the arrow-cursor changes into a hand; by clicking the right mouse-key a menu is opened where you can chose “Show component documentation”



4.4.4 Selecting and positioning of inputs and outputs

Depending on choosing to position an input or an output on the work-surface, you have to click on the icon

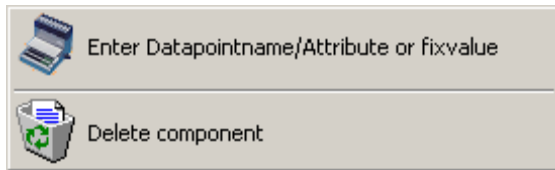


„Input“ or „Output“. Now its possible to position an input-diagramm or an output-diagramm an the work-surface via drag-and-drop.

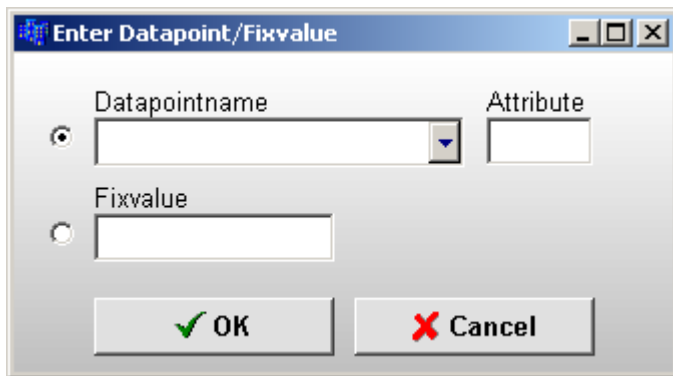
If you want to move an input- or an output-diagramm on the work-surface, you have to bring the mouse-cursor into the middle of the function-element-diagramm (onto the name of the diagramm) until the arrow-cursor changes into a hand; now its possible to move the input- / output-diagramm per drag-and-drop (**Important!** Left mouse-key).

4.4.4.1 Punktnamen/Attribut- bzw. Fixwertzuweisung eines Eingangs / Ausgangs

If you want to give a datapoint-name or a fixed value to an input or an output on the work-surface, you have to bring the mouse-cursor into the middle of the function-element-diagramm (onto the name of the diagramm) until the arrow-cursor changes into a hand; by clicking the right mouse-key a menu is opened where you can chose **“Enter Datapointname/Attribute or fixvalue”**;



the following window is opened:

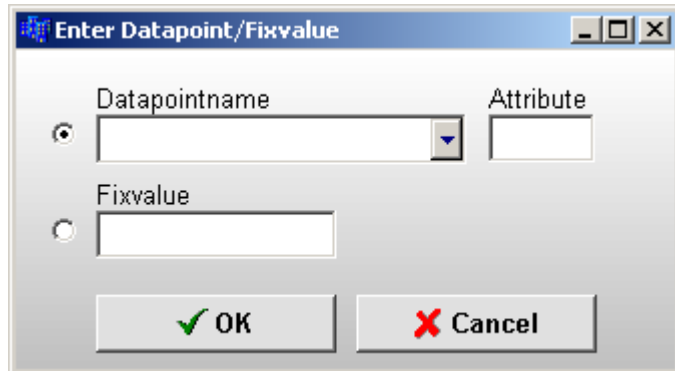


4.4.4.1.1 Datapoint-name-/attribute-assignment of an input / output

If you want to assign a datapoint-name and an attribute, you can use the therefor intended fields. If you don't know the datapoint-name from memory the system offers the possibility to search for the name in the datapoint-name-list of Babylon (Attention! The datapoint has to be assigned befor in the “datapoint-definition” of Babylon/NT)); when clicking on the field with the small black triangle, the following window is opened:

4.4.4.1.2 Fixvalue- assignment of an input / output

If you want to assign a fixed value to an input respectively output, you have first to activate the fix-value-input by clicking onto the small white Button beneath the fix-value-field:

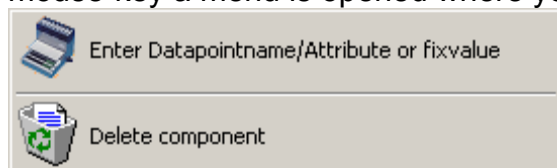


Now you can fill in the desired fix-value.

By clicking the button „OK“  the value will be confirmed and you are back on the work-surface.

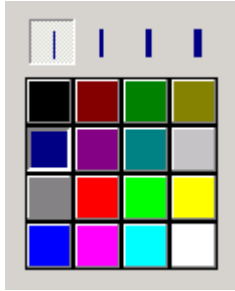
4.4.5 Remove an input / output from the work-surface

If you want to delete an input respectively output on the work-surface, you have to bring the mouse-cursor into the middle of the input-/output-diagramm (onto the name of the diagramm) until the arrow-cursor changes into a hand; by clicking the right mouse-key a menu is opened where you can chose “Delete component”:



4.4.6 Drawing of connection-lines

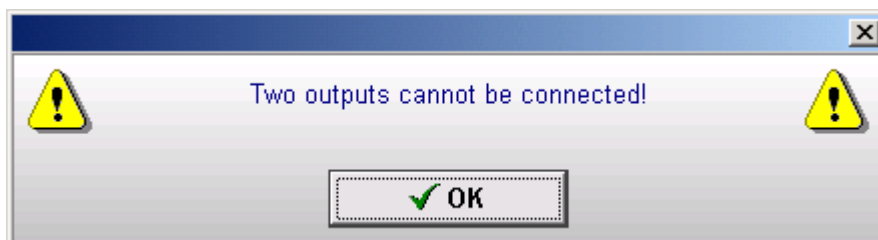
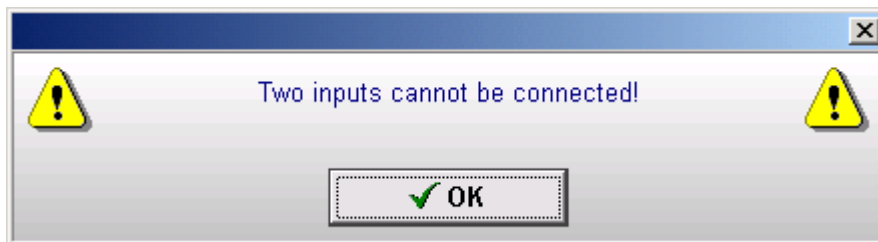
The choice of colour and size of the connectionlines has to be done via the menu-bar on the right side of the screen-mask:



For connecting –for example- an input-element with the input of a function-element, you have to move the mouse-cursor onto the pin of the input-element until it appears red. Now, while pressing the “**shift-key**”, you have to click on the left mouse-key and pull the cursor into the desired direction. An elastic band appears which connects the tip of the pin with the mouse-cursor. By pulling the mouse-cursor to the desired input-pin of the function-element -until also this pin appears red- you can create a solid connection between this pins by clicking the input-pin of the function-element with the left mouse-key. The connection-lines way is automatically created by the programme. If you want to influence it, you can create kinks into the “elastic band” –while pulling the mouse from one to the other pin- by clicking the left mouse-key.

(Attention! It is possible, that the connectionline comes to lie behind a function-element-diagramm. Even so the connection exists; but the diagramme overview will be more difficult.)

Attempts to connect two inputs respectively two outputs to each other are restricted by the system; the following error-message appears:



4.4.7 Delete connectionlines

For removing an existing you have to move the mouse-cursor onto the pin of the input-element until it appears red. Now, while pressing the “**control-key**”, you have to click on the left mouse-key and pull the cursor to the other pin of the connectionline until also this pin appears red. By repeated pressing of the left mouse-key (the “control”-key still has to be pressed) the connectionline will be deleted.

4.4.8 Moving the function-elements and connectionlines

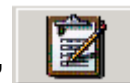
In spite of solid connectionlines it is possible to move the function-elements without problem:

If you want to move an already connected function-element on the work-surface, you have to bring the mouse-cursor into the middle of the function-element-diagramm (onto the name of the function-element) until the arrow-cursor changes into a hand; now its possible to move the function-element-diagramme per drag-and-drop (**Important!** Left mouse-key). The connectionlines will be fitted automatically to the new position of the function-element-diagramm on the work-surface.

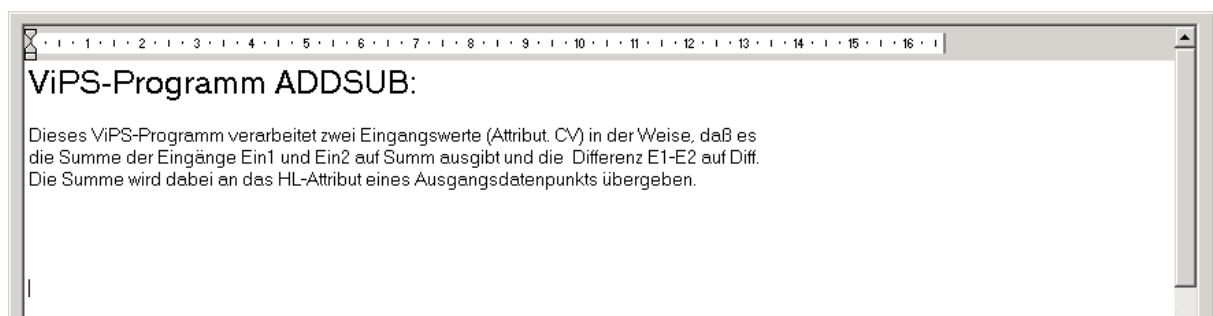
(**Attention!** It is possible, that the connectionline comes to lie behind a function-element-diagramm. Even so the connection existes; but the diagramme overview will be more difficult.)


4.4.9 Creating of documentations for ViPS-programmes

By clicking the Icon



„**Toggle: Connection Diagram <-> documentation**” you can switch between the function-element-circuit-diagramm of the opend system and a documentation of this system; this documentation should be created by the erector of the system as good and informative as possible.



In the field „RTF-Docu“ on the left menu-bar  the size of the momentary shown documentation is notified.

4.4.10 Using the debug-mode

The debug-mode offers the possibility to show the momentary values of the inputs, outputs, auxiliary figures and parameters.
To switch on the debug-mode of the ViPS-function-element application-programm, you have to click on the icon



„**Switch Debug-Mode on/off**“ .

If the debug-mode runs, the pin-descriptions at the function-elements are replaced by the actual attribute-values of the pins. The values are renewed approximately every second. The function-elements are still running when the debug-mode is switched on; the system is still be handled.

For leaving the debug-mode you have to click one more time on the icon „ **Switch Debug-Mode on/off** “ .

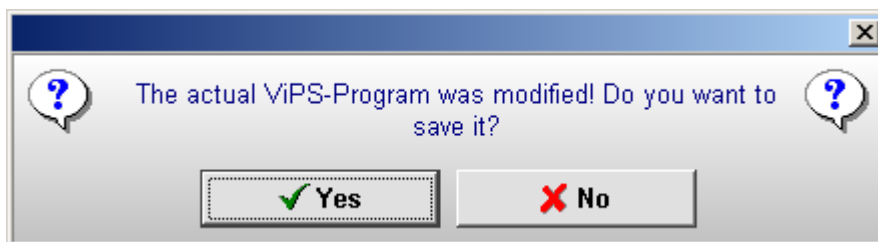
4.4.11 Saving the created ViPS-programmes

After generating a new ViPS-programme respectively changing an already existing one, you can save the programme in a data-bank by clicking on the icon



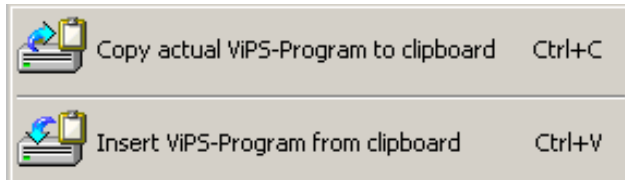
„**Save actual ViPS-Program in database**“-Buttons

If you want to leave the „Component-Application-Program“ after a change, a window opens where you are asked whether you want to save the changes or not:



4.4.12 Copying created ViPS-programmes

ViPS-programmes can be copied into and from the memory via CTRL-C / CTRL-V respectively via the menu, opened by clicking the right mouse-key.



4.4.13 Export and import of ViPS-programmes

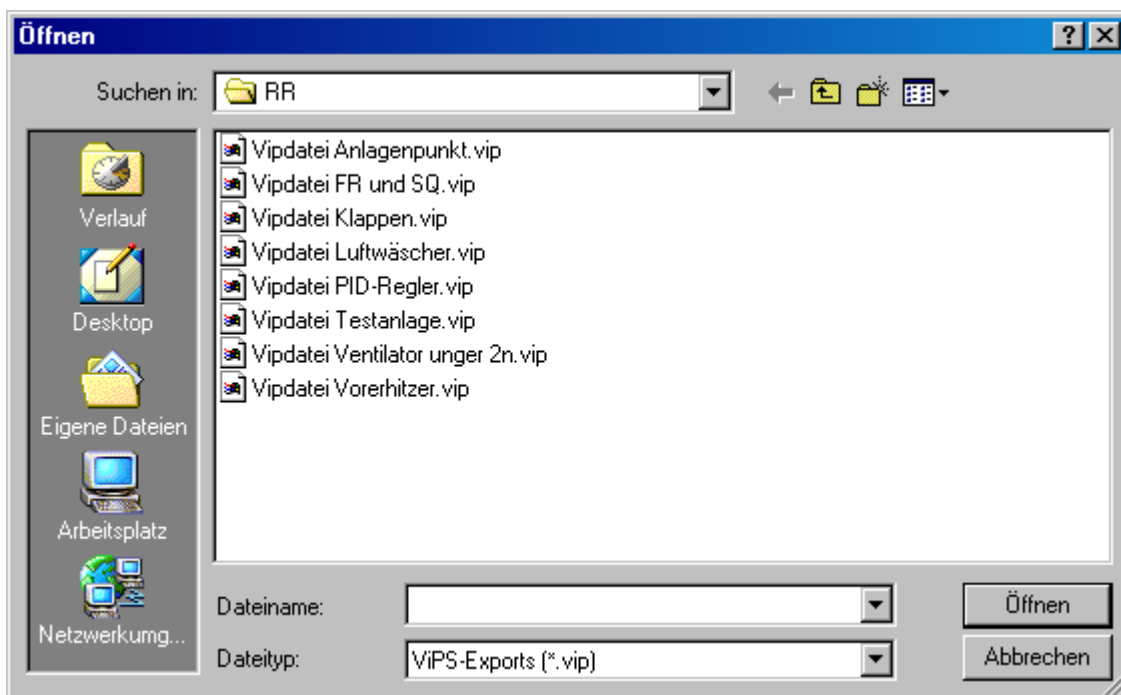
ViPS-programmes can be imported from files respectively exported into files. The herefore necessary functions you can find in the icon-menu-bar; the functions are described in the following sections.

4.4.13.1 Import a ViPS-programme-file

If you want to import a ViPS-programm from a file, you have to click on the icon



„Import ViPS-Program(s)“. A window is opened where you can chose the desired ViPS-programme-file:



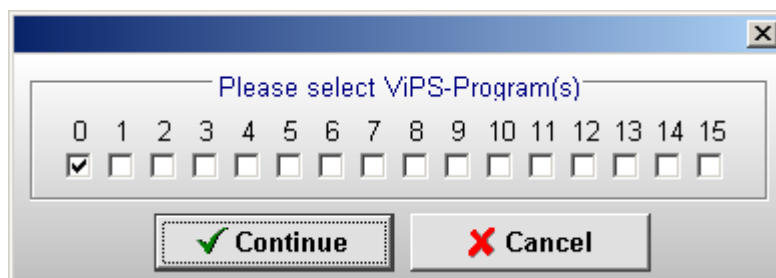
By clicking the button „**Öffnen**“  the content of the desired ViPS-programme-file will be opened in the ViPS-work-surface.

4.4.13.2 Export ViPS-programmes into a file

If you want to export a ViPS-programm into a file, you have to click on the icon

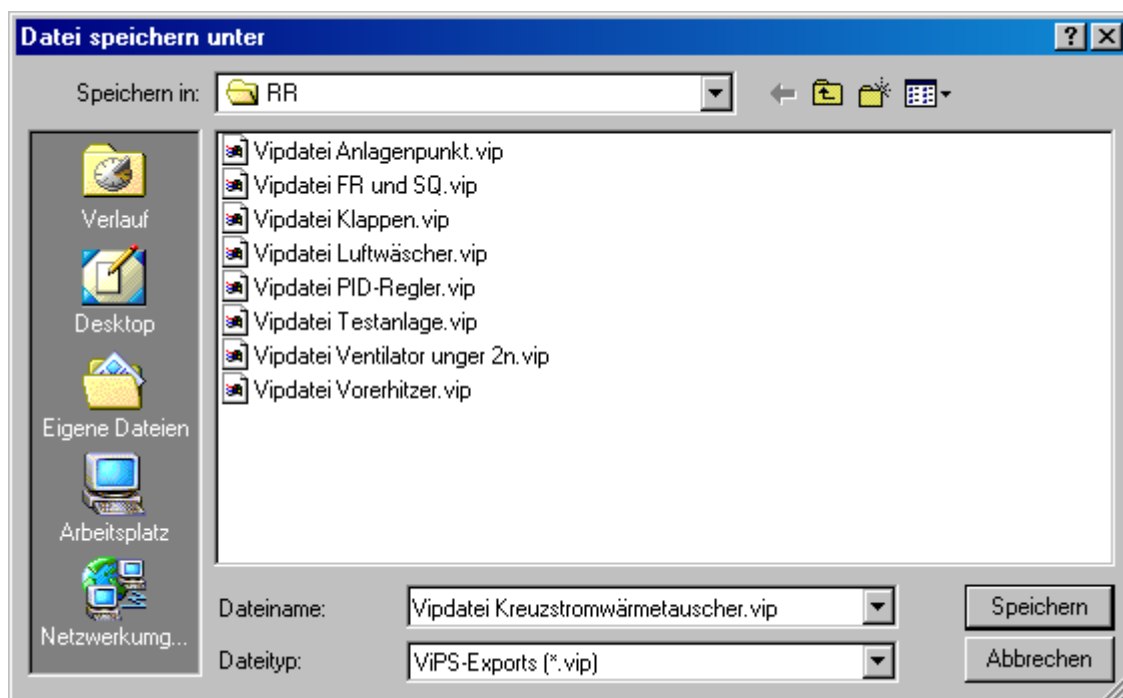


„**Export ViPS-Program(s)**“. A dialogue-window is opened where the user can chose, which ViPS-programme he wants to export; here it is also possible to chose several programmes at the same time:




After choosing the desired ViPS-programmes and clicking on

„**Continue**“ , the following dialogue-window is opened:



Here, the desired file-name for the exported ViPS-programme can be filled in.

By clicking the button „**Speichern**“ , the programm will be saved.

4.4.14 Download of a ViPS-programme into a GA-Box

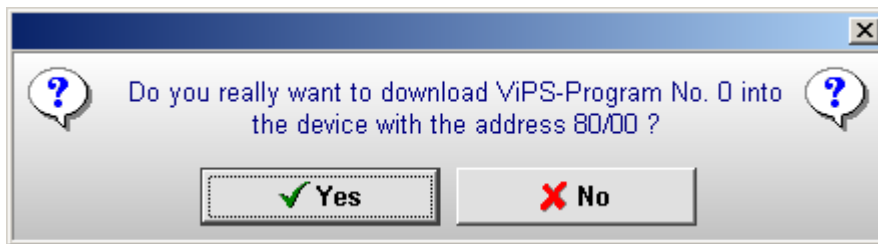
The ViPS-programmes, which shall be downloaded into a GA-Box have to be activated; this has to be done by clicking into the box in front of the writing “Activated”


Activated

After clicking on the icon „Download actual ViPS-Program into GABOX/K32“



the following dialogue-window is opened:



If now clicking on „Yes“ , a download of the open ViPS-programme into the desired GA-Box takes place.

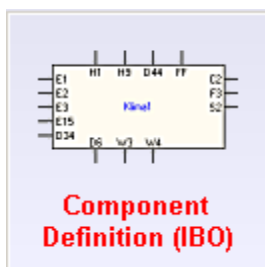
Alternatively it is possible to download all 16 ViPS-programmes simultaneously into the GA-Box by clicking the icon

„Download all 16 ViPS-Programs into GABOX/K32“

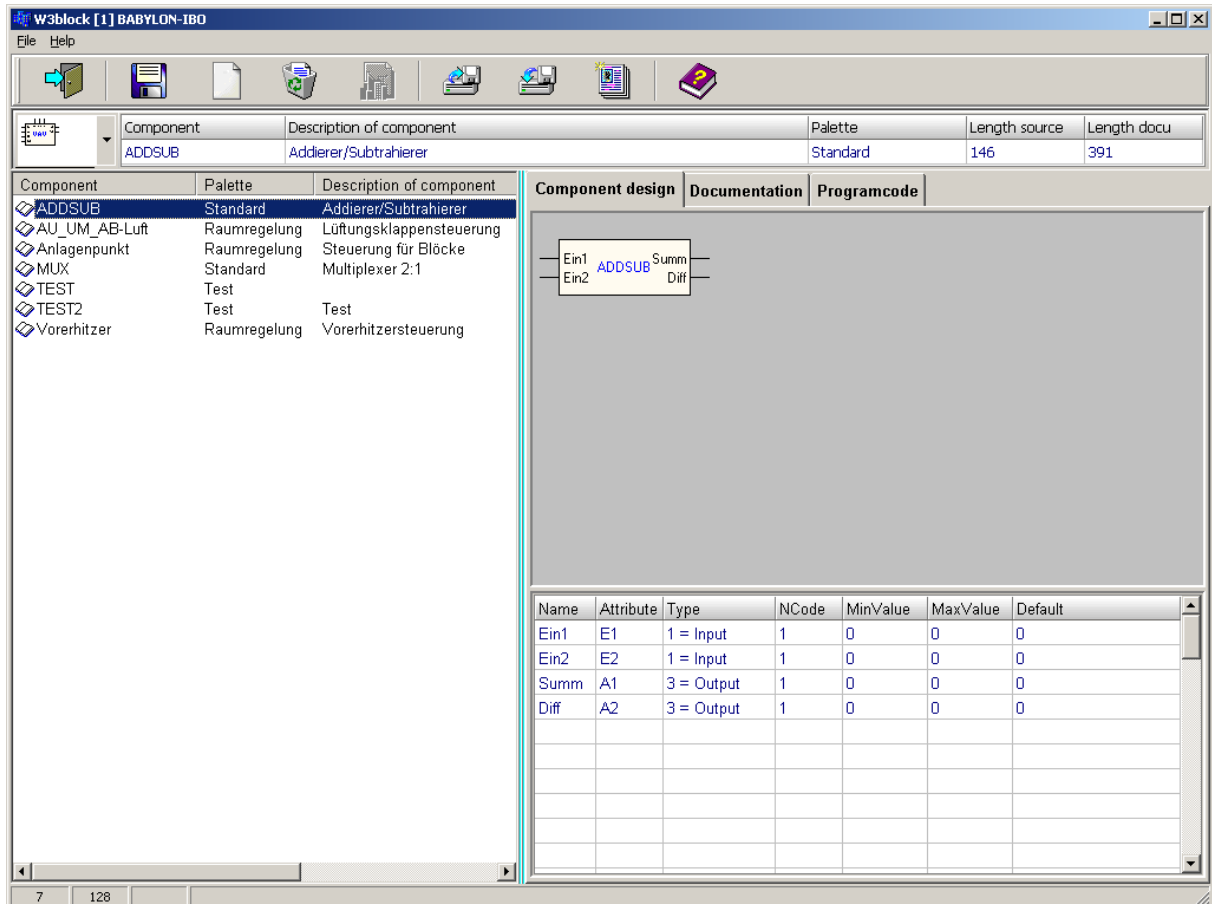


5 ViPS-element-definition

Babylon/NT building-automation-mask (URANUS):



By clicking the icon „Component Definition“ you are reaching the programm for the creating of function-elements:



5.1 Icon-menu-bar



Exit Program: Serves to leave the programme.



Save component in database: For saving a new created function-element in the data bank.



Add new component: Serves to create a new function-element



Delete component: Removes an existing – in the left column of the mask blue marked– function-element from the function-element-list.



Export component(s): Opens the export-dialogue for the function-elements; the exported files have a special ending: .blx .



Import component(s): Opens the import-dialogue for the function-elements; the importable files have a special ending: .blx .



Build RTF-File with component description:



Help: Starts a help-file, in which the functions of the mask are declared.

The same icons and corresponding functions can be found in the top-menu by clicking on File or Help.



5.2 Function-element-list

In the left column of the screen mask all available function-elements are listed with **Component**, **Palette** und **Description of component**.

Component	Palette	Description of component
ADDSUB	Standard	Addierer/Subtrahierer
AU_UM_AB-Luft	Raumregelung	Lüftungsklappensteuerung
Anlagenpunkt	Raumregelung	Steuerung für Blöcke
MUX	Standard	Multiplexer 2:1
TEST	Test	
TEST2	Test	Test
Vorerhitzer	Raumregelung	Vorerhitzersteuerung



5.3 Function-elements-detail-characteristics/diagrammes

Bellow the icons some characteristics of the blue marked function-element (see left column of the screen mask) are listed:

Component	Description of component	Palette	Length source	Length docu
ADDSUB	Addierer/Subtrahierer	Standard	146	391



This menu-point enables the user to assign special icons to the function-elements. When clicking on the field with the small black triangle, a palette of icons is opened. From these palette an icon can be selected by simply clicking; in this way the icon is connected to the at the time open function-element; by clicking the



„Save component in database“-Icon the assignment ist saved.

If no icon is selected a fixed standard-icon will be used.

Remark! The selectable function-element-icons are saved in folder \EXOS386D\BMP\w3block ; new one can be added by saving new bmp-files in this path.

Component:

In this field the function-element-name (up to 16 characters) can be filled in.

Description of component:

This field can be used for a function-element-description (up to 48 characters).

Palette:

In this field the assignment of a funktion-element to a spezial function-element-group can be made. For the name of the palette up to 16 characters are possible.

Length source:

Number of characters of the function-element source-code.

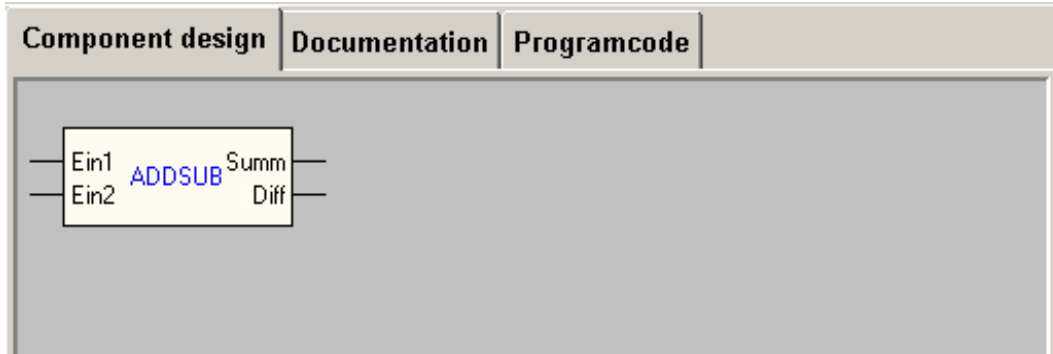
Length docu:

Number of characters of the function-element documentation.

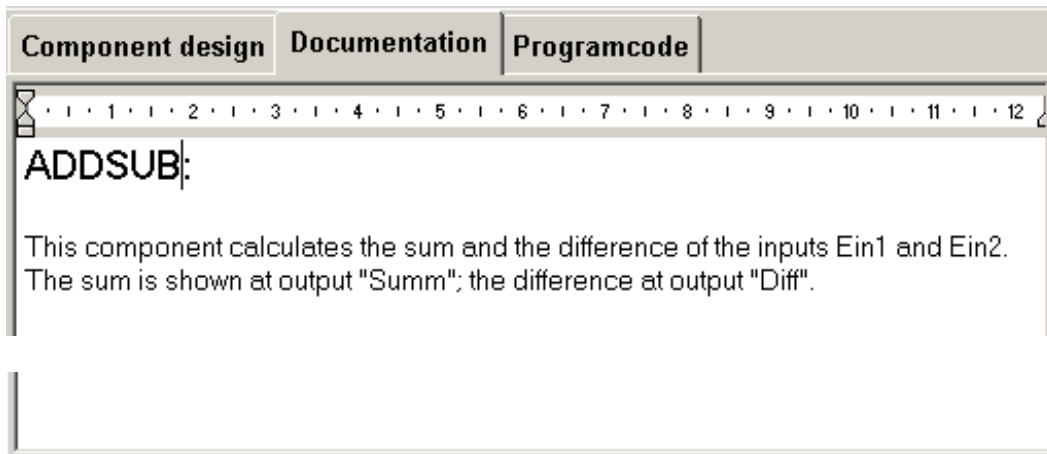
In the middle part of the right column of the screen-mask you can see the function-element-design (**Component design**), the documentation (**Documentation**) or the source-code (**Programcode**) of the blue marked function-element (see left column).
abbreviation

Component design: In the register-card „Component design“ you can see a graphical presentation of the function-element. Inputs and outputs usw. are recognizable at the function-element-edges and are marked with their corresponding name-abbreviation.

In the below part of the screen-mask the inputs, outputs, auxiliary figures and parameters are listed up. For each the name, attribute, type, ncode, min-value, max-value and default-value are listed.



Documentation: In the register-card „Documentation“ a function-description of the marked function-element can be called up.



Programcode: In the register-card „Programcode“ the source-code of the marked function-element can be called up if it is not locked up.

Component design	Documentation	Programcode
<pre>pin("Ein1", "E1", 1, 1, 0, 0, 0); pin("Ein2", "E2", 1, 1, 0, 0, 0); pin("Summ", "A1", 3, 1, 0, 0, 0); pin("Diff", "A2", 3, 1, 0, 0, 0); { A1 = E1 + E2; A2 = E1 - E2; }</pre>		
Source	Listing	Token
Errors	Code	Symbols

5.4 Creation of a new function-element

5.4.1 Start a new function-element



After clicking the icon, a dialogue-window is opened where you have to fill in element-name, element-description, and palette (that means the function-element group-affiliation). By clicking the OK-button you have to confirm your inputs.

A small box with the element-name has been created on the register-card

Component design .

The inputs, outputs, auxiliary figures and parameters have to be created in the subregister „**Source**“ of the register-card „**Programcode**“.


An input is –for example- created like followed:

```
pin("Ein1", "E1", 1, 1, 0, 0, 0) ;
```

The paranthese behind the pin-command contains seven items:

1. Longname of the pins; maximum 4 figures
2. Attribute-name (short-name of the pins); always 2 figures
3. Pin-type: 0=invisible 1=input, 2=auxiliary figure, 3=output, 4=parameter
4. NCode (see table BABYLON)
5. Minimum-value
6. Maximum-value
7. Default-value (start-value)

After entering the pin-command as shown above, the created programm-code has to

be saved by clicking the icon  :now it has to be compiled by clicking the icon



(Attention! The compiling-icon is only available if the register-card „Programm“has been opened. The by the compiler created ASCII-files are saved in the BABYLOND-path.). After opening the register-card „Baustein-Design“ you can see the generated pins with their „longnames“ at the edge of the function-element. At the same time the created pins are listed up below the presentation-field.

Name	Attribute	Type	NCode	MinValue	MaxValue	Default
Ein1	E1	1 = Input	1	0	0	0
Ein2	E2	1 = Input	1	0	0	0
Summ	A1	3 = Output	1	0	0	0
Diff	A2	3 = Output	1	0	0	0

The register-card „Documentation“ can be used for a description text for the function-element (duties, features, etc.).

(additional informations: see chapter 5.4.2)

5.4.2 Generation of function-element source-code

If you are ready with parameterising the inputs, outputs, auxiliary figures and parameters of the function-element you can start to programme the working-funktion of the function-element.

For this an own programming-language (**“GA language”**) –similar to C- is placed at your disposal. The commands and possibilities of this language are shown in section 5.4.2.1.

By using these commands you can write the necessary programme-code for the function-elements in the subregister **„Source“** of the register-card **„Programcode“**.

Component design	Documentation	Programcode
<pre> pin("Ein1", "E1", 1, 1, 0, 0, 0); pin("Ein2", "E2", 1, 1, 0, 0, 0); pin("Summ", "A1", 3, 1, 0, 0, 0); pin("Diff", "A2", 3, 1, 0, 0, 0); { A1 = E1 + E2; A2 = E1 - E2; } </pre>		
Source	Listing	Token
Errors	Code	Symbols

Beside the subregister „**Source**“ there are some more subregisters with the following features:

Listing: Source-code by the line numbered

Token: Scanned source-code; input-file for the parser (part of the compiler)

Errors: Shows the errors occurred during the compilation; if no errors where found the complete source code is shown in this window.

Code: By the compiler generated, workable code.

Symbols: List of the variables used by the programme

5.4.2.1 Commands for the GA-language

Example for the programme-structure:

```
pin ("E001", "E1", 1, 1, 0, 0, 0);
pin ("E002", "E2", 1, 1, 0, 0, 0);
pin ("A001", "A1", 3, 1, 0, 0, 0);
pin ("A002", "A2", 3, 1, 0, 0, 0);
int i, j;
{
  A1 = E1 + E2;
  A2 = E1 - E2;
  i = i+5;
  j = i+17;
}
```

Remarks:

- pin- and int-instructions have to be written before the main-programm starts { ... }.
- In the programme-code the data-points can be reached either via their longname or via their attribute-name (short-name).

Commands:

Bellow there is a list of all available commands:

```
//
/* and */
pin(...)
int
{ and }
=
+
-
*
if ...{...} else {...}
for (...) {...}
while (...) {...}
goto
wait (...)
event (...)
„system-variables“
```

Now following, the commands are described in detail:

Commands for commentaries:

```
//
All written in a line behind the command // will be interpreted as a commentary.

/* .... */
All written between the commands /* and */ will be interpreted as a commentary.
```

Pin-command: Serves to create pins (inputs, outputs,...) for the function-elements.

pin("E001", "E1", 1, 1, 0, 0, 0);

1. Long-name – up to 4 characters
 2. Attribute-name – always 2 characters
 3. PinTyp: 0=invisible 1=input, 2=auxiliary figure, 3=output, 4=parameter
 4. Ncode (see table BABYLON)
 5. Minimum value
 6. Maximum value
 7. Default value or start value
-

Int-command: Serves for the variable-declaration, if necessary.

```
int i, j;
```

Main-programm: Is included by sweeping parantheses.

```
{...  
}
```

Assignment:

```
A1 = E1 + E2;
```

If-loop:

```
If < prerequisite >  
{  
}  
else  
{  
}
```

For-loop:

```
For (start value, prerequisite, expression)  
{  
}
```

Example: for(i=0;i<20;i=i+1)

```
{  
  j = j + i*2;  
  A1 = j+3;  
}
```

While-loop:

```
while (prerequisite)  
{  
}
```

Example: while(i < 10)
 {
 i = i + 2;
 j = i * 3;
 }

Goto-command: Enables to jump to a certain point in the programme; this point has to be marked by a label (= any name you like). The command-structure:

goto label;

The point you want to jump at has to be marked as followed:

label

Wait-command: Enables to interrupt the working programme for a defined time; other element which are running parallel are not interrupted and are still working:

wait(expression);

expression = number of seconds for the interrupt

Example: wait(E1+2);

Event-command: Generates an alarm when a special event has taken place:

event(figure);

Generate an alarm (Message group 40, Message -1)

Example: event(6) -> Kontakt-alarm

System-variables:

CY = current year (2003)

CM = current month(1..12)

CD = current day(1..31)

CW = current weekday(0=Monday, 1=Tuesday... 7=Holiday1, 8=Holiday2..)

CN = tomorrow weekday

TIME = momentary time in minute of the momentary day (0..1439)

RAND = random figure

COUNT1 = 10ms system-counter
COUNT2 = 100ms system-counter

Example: $A1 = CY + 10;$

5.5 Export and import of function-elements

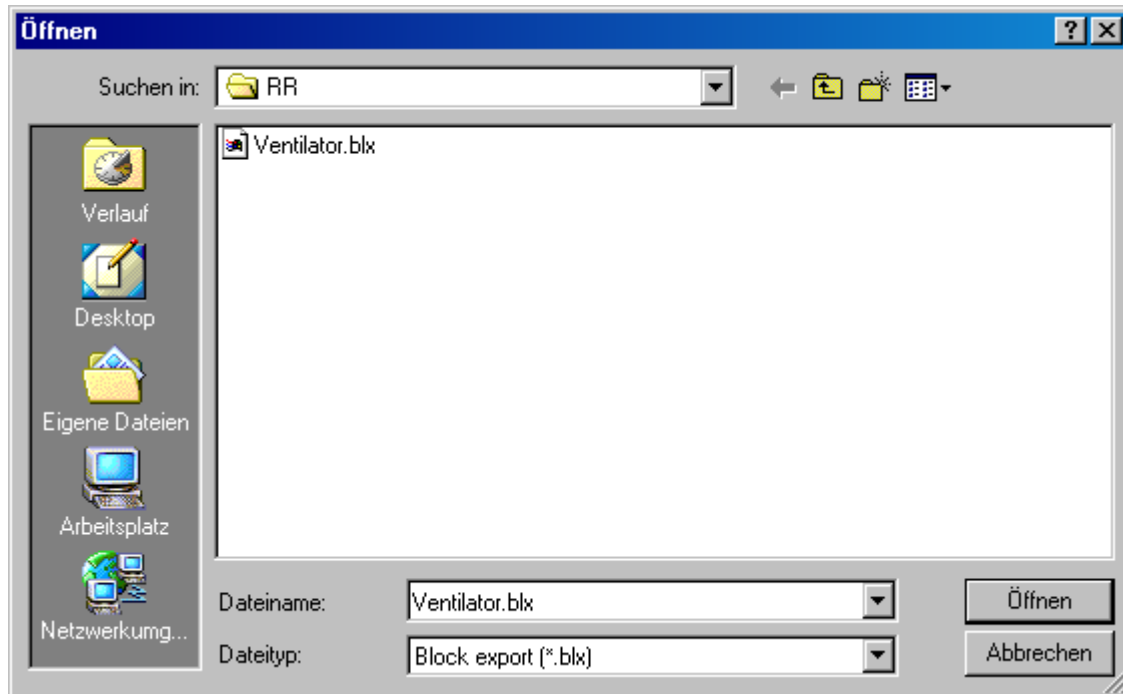
Function-elements can be imported from respectively exported into files.
The therefor necessary functions are available from the icon-menu-bar.

5.5.1 Import of a function-element

If you want to import a function-element from a file you have to click on the icon



„Import component(s)“. A dialogue-window opens where the desired function-element-file can be selected:



By clicking the button „Öffnen“  the selected function-element appears in the ViPS-work-surface.

5.5.2 Export of a function-element

If you want to export a function-element into a file you have to click on the icon

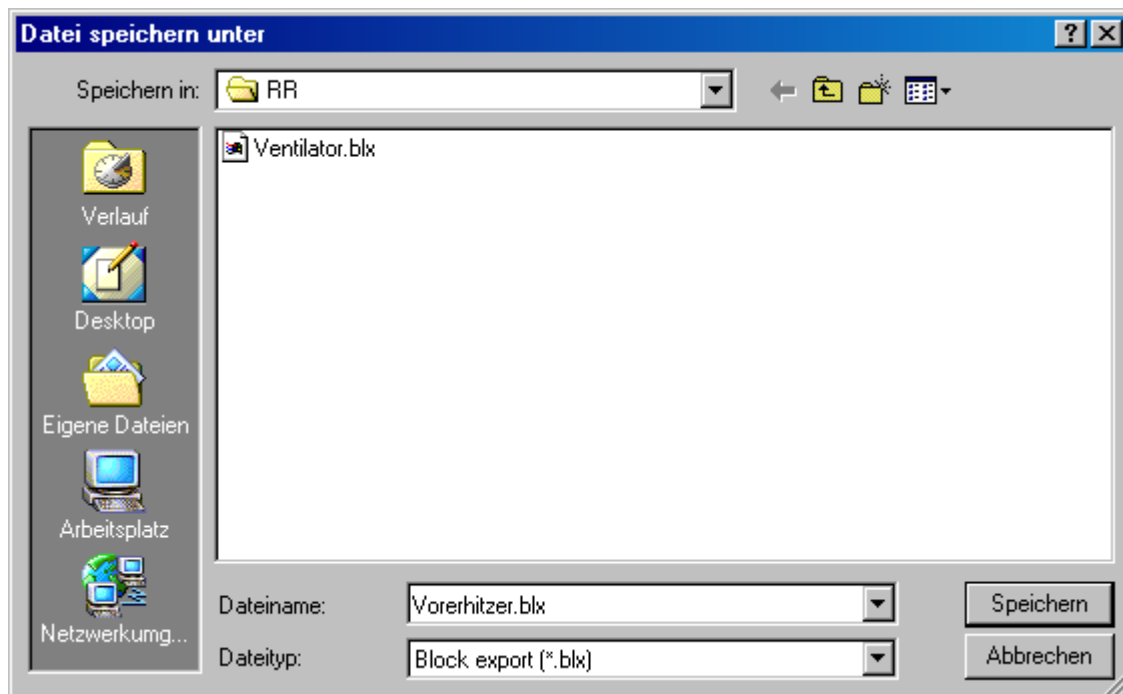


„**Export component(s)**“. A dialogue-window opens where the user is asked whether he wants to export the function-element with or without source-code. If clicking on „Yes“ the source-code will be exported into the new file, if clicking on „No“ it won't.



If the source-code will not be exported, the corresponding function-element will be marked with a small lock-icon; after a reimport there is no more access possible to the programm-source-code.

In the next step the following dialogue-window is opened:



In this window it is possible to chose the name under which the exported function-element should be saved. The saving-process starts by clicking the Button

Speichern .

6 Starting up a ViPS-programme

After the ViPS-programme has been downloaded into the desired GA-Box (analog to Part 4.4.14), the programme runs automatically.

The following point has to be noted:

- While a ViPS-programm is running the input-values of the function-elements keep their value until the programm finishes.

AUTEC Gesellschaft für Automationstechnik mbH
Bahnhofstraße 57-61b
55232 Framersheim
Tel.: +49 (0)6733-9201-0
Fax: +49 (0)6733-9201-99
Email: vk@autec-gmbh.de
Internet: www.autec-gmhb.de