ArcSuit

Documentation

KNX arcus-eds

Version:	0.1
Date:	16.06.2010

Contents

1 ArcSuit Documentation

1.1 General

This document contains information such as data, illustrations, measurements and others which are subject to change without prior notice. Additional information is available at `http:/www.arcus-eds.de`.
Technical changes preserved!
All product- descriptions used in this documentation are registered trademarks of the respective companies. No parts of this document must be copied or transmitted without the explicit permission of Arcus-edsGmbH, irrespective of the purpose or the way in which it is carried out.
All rights reserved!
©by Arcus-EDS GmbH

1.2 ArcSuite

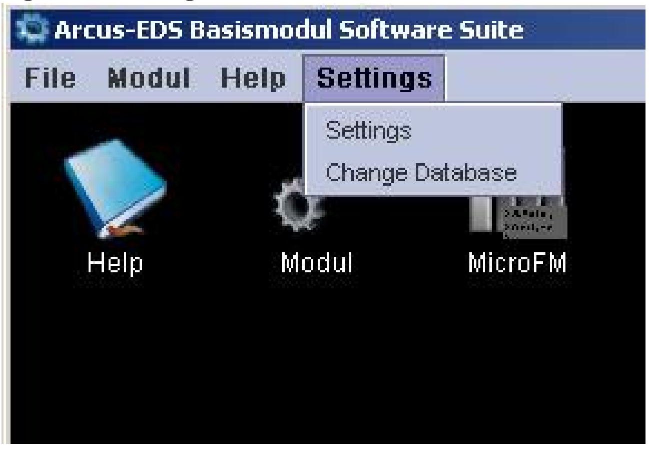
The ArcSuite is a Programm-Collection associated to the Products of the Arcus-EDS GmbH.

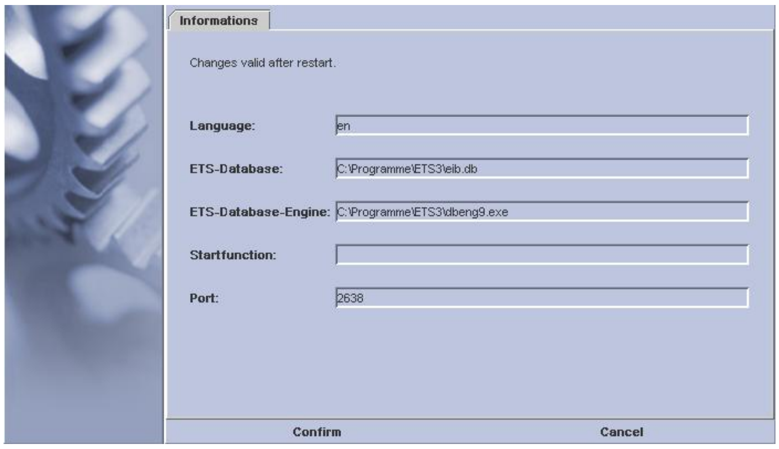
1.3 Modules

Modules are small Programs for the different Products of Arcus-EDS GmbH. Currently these Modules are implemented:
MicroVis for the Configuration of the Graphical Interface of the MicroVIS-Display
MicroFM for interactive Programming of the MicroFM Function-Module
FontEditor to generate individual Fonts for the MicroVIS
Systemmodul as an Overview over the Usage of the Systemresources by the ArcSuite
MicroVis2 for the Configuration of the Graphical Interface of the MicroVIS2-Display, which is an improved Version of the MicroVis
MicroVis2-Logic for interactive Programming of the Logic Part of the MicroVIS2

1.4 Settings

The ArcSuite Settings are global Settings for all Modules.





Changes valid after restart.

Language: en

ETS-Database: C:\Programme\ETS3\eib.db

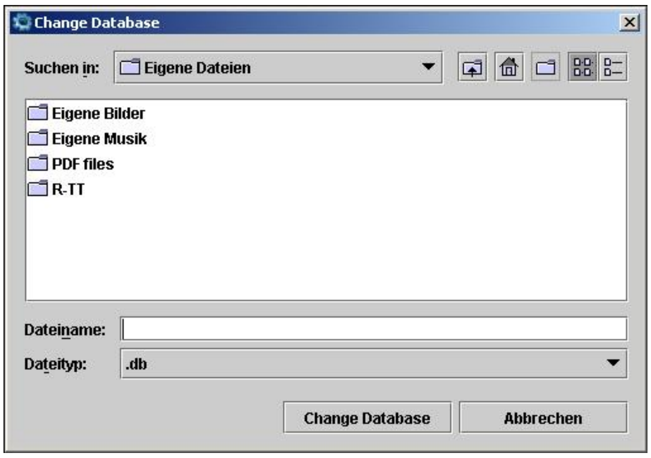
ETS-Database-Engine: C:\Programme\ETS3\dbeng9.exe

Startfunction:

Port: 2638

Here you can alter the Language of the ArcSuite. For availability of languages look at `www.arcus-eds.de`
The Database-Engine is the one that comes with your ETS3-Software. If you can not connect to your ETS-Database, you probably have an unusual Installation, locate your Database-Engine here. The ETS-Database to use can be entered in the specified Field or it can be altered by Change Database.

Change Database:


Change Database

Suchen in: Eigene Dateien

Eigene Bilder
Eigene Musik
PDF files
R-TT

Dateiname:
Dateityp: .db

Locate the Database that shall be used to import the Informations from your ETS-Project.

2 The MicroFM Module

2.1 Introduction

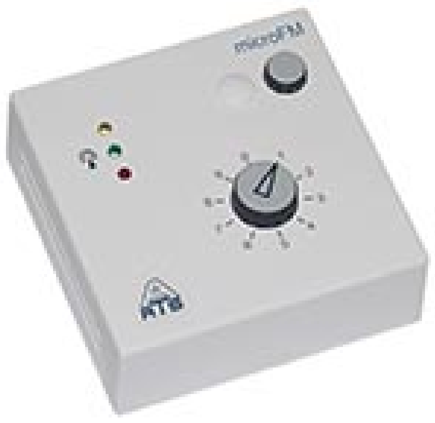
This document contains information such as data, illustrations, measurements and others which are subject to change without prior notice. Additional information is available at http://www.arcus-eds.de.
Technical changes preserved!

2.2 Systeminformation

This device is a product for the Instabus- EIB / KNX- System. Detailed knowledge in depth of the Instabus- EIB/KNX- System is essential. The functions of the device are software- depending. Detailed information, which software can be loaded and which function- capacity is then available as well as information about the software itself, have a look at the software- details of the manufacturer.
It is operated with the software tool ‘ARC Suite’ and is ready for downloading under http://www.arcus-eds.de.
This device is working with a real- time operating system FreeRTOS (www.freertos.org).

2.3 Operating Controls

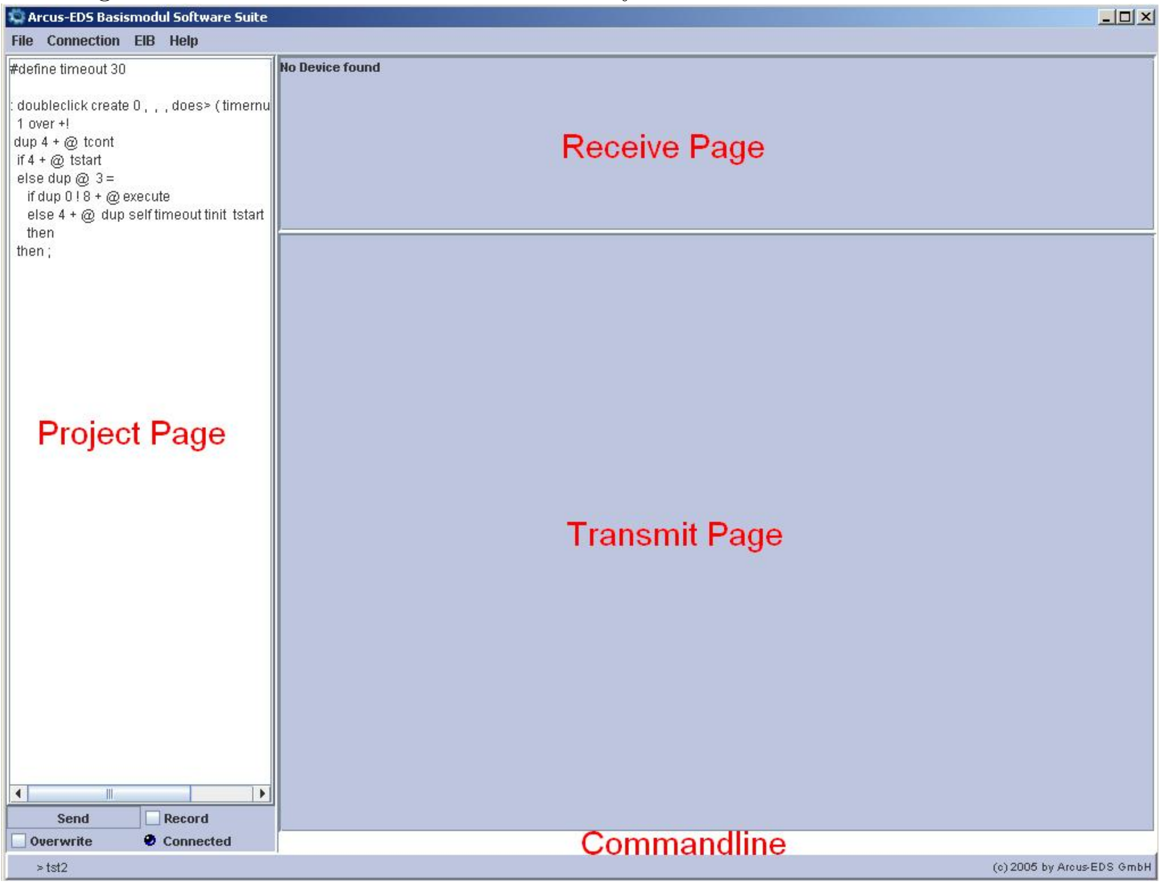
Your MicroFM has free programmable elements of **Operating & instruction control**



There are **3 Programmable LED's**, a **10 Step-Switch** and one Pushbutton. MicroFM will also be delivered optionally with an integrated DCF77 - Time - Signal Receiver.
LEDs can be switched on or off by the command: **setled**
The 10 step-switch can be polled with the command: **getsw**
When operated, the pushbutton calls up the function **onpb**,
which can be freely defined by the user.
DCF-77:
In case of the model with the DCF 77- receiver the green LED is flashing in sequence with the incoming time- signals. These have to come in up to 2 minutes without interruption to make the green LED lit constantly. If an interruption of the signal - flow occurs, the synchronisation will be repeated completely. When the synchronisation has been successful, the current time can be read by 'syst' and the current date by 'sysd'. In addition, 'syst' and 'sysd' are sending back the information 'summer-time' in Bit number 24, and 'synchronised' in Bit number 25.

2.4 Program Components

After starting the MicroFM- Module from ARCSuite you see the Userinterface.



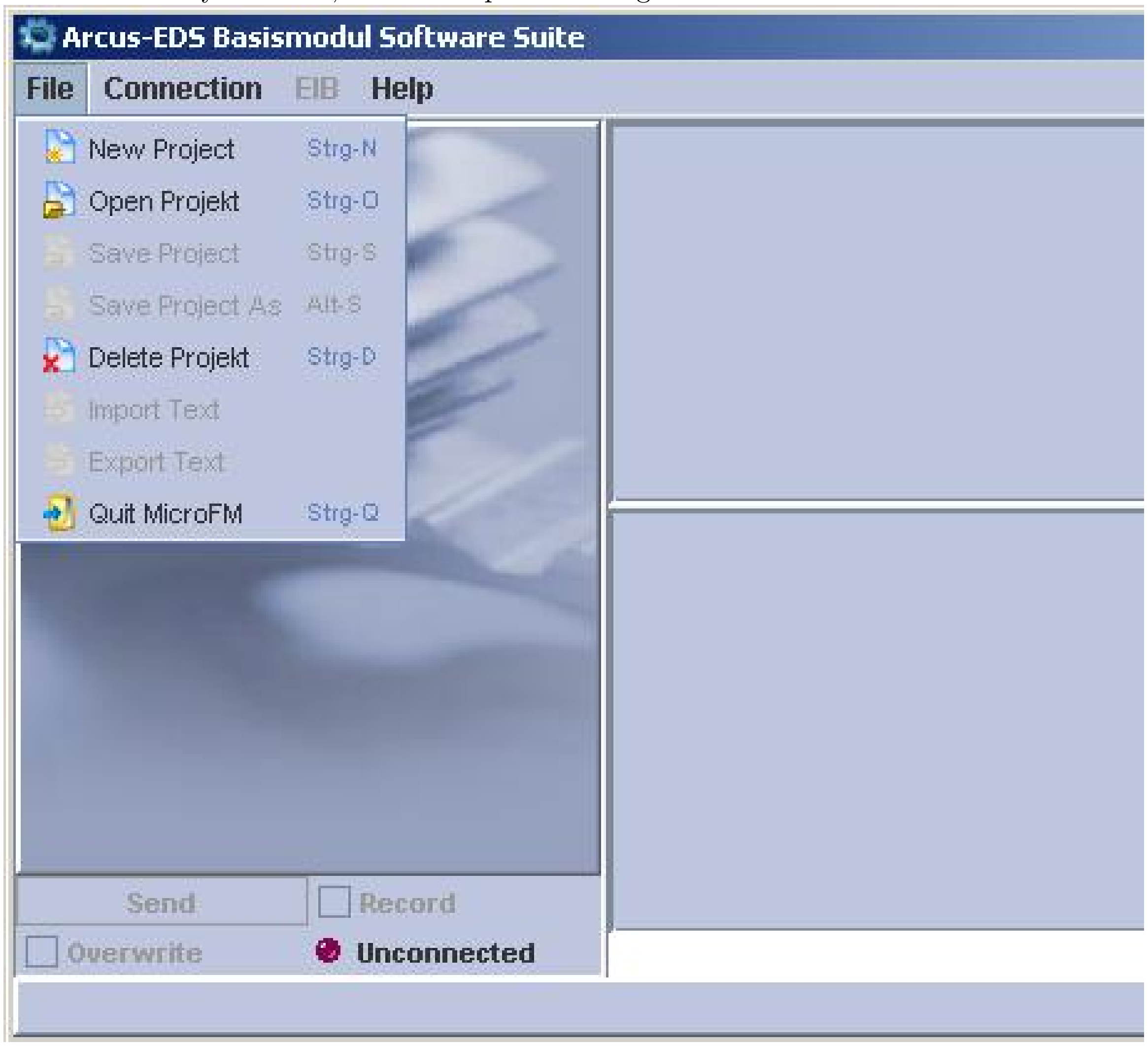
```
#define timeout 30

: doubleclick create 0 , , , does> ( timernu
 1 over +!
dup 4 + @ tcont
 if 4 + @ tstart
 else dup @ 3 =
   if dup 0 ! 8 + @ execute
   else 4 + @ dup self timeout tinit tstart
   then
 then ;
```

Inputdata executed in the **Commandline** will be transmitted to MicroFM directly.
These input data are on record at the **Send-page**.
The Reception page shows the text-output data of MicroFM.
The latest input data executed in the Commandline are recalled by **SHIFT+UP** and **SHIFT+DOWN**.
The **project-page** will be saved together with the project. If the option **Recording** is marked, all input data from the Commandline are transferred to the project-page. Through the option **Overwrite** already defined functions from the project- page will be replaced automatically by updated definitions. By operating the pushbutton **Send** the whole project page is transferred to MicroFM Module.

2.5 Programming

2.5.1 Projects

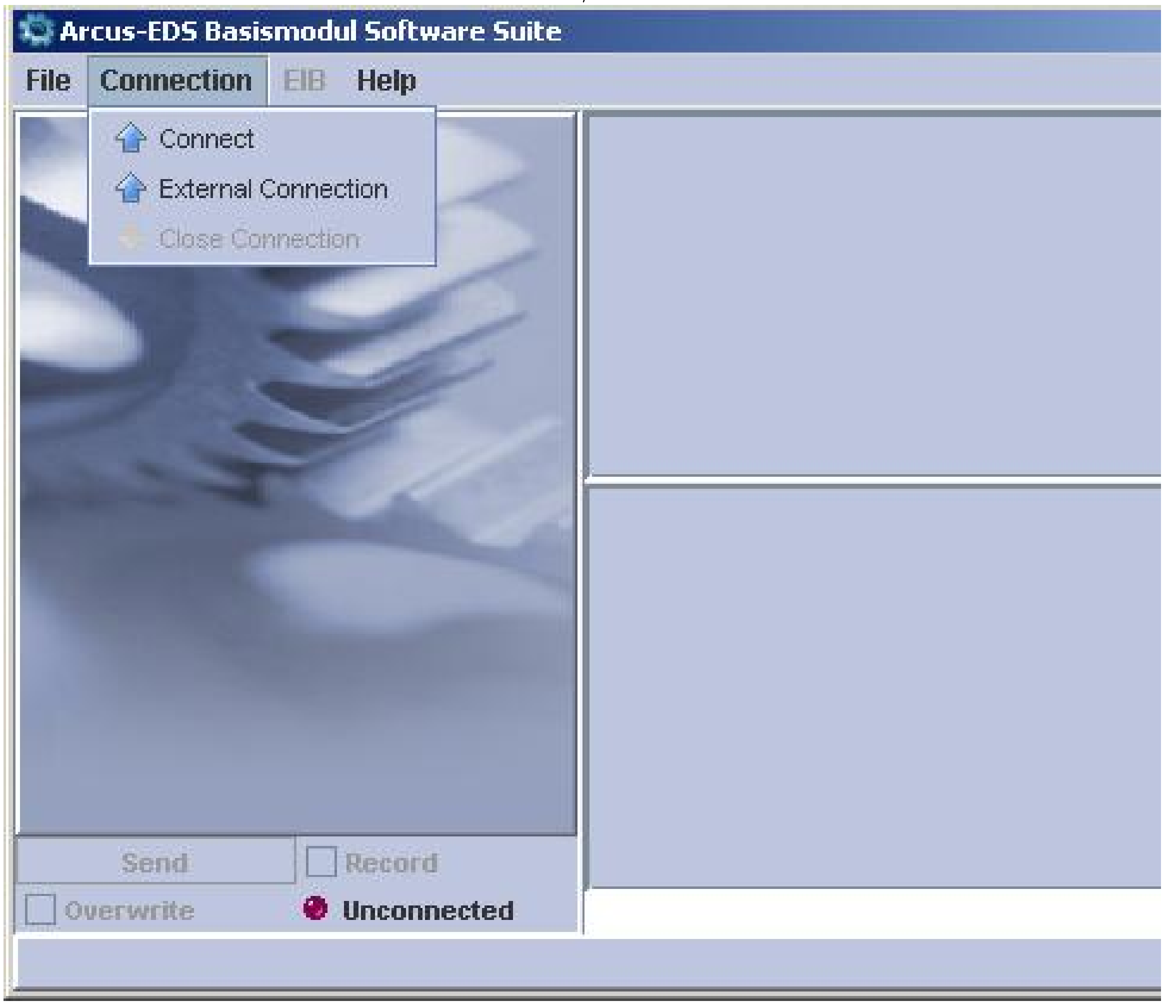
Projects can be newly installed, stored or opened through the menu-item **File**.



Additional to the **projectname**, a **description** together with the **name of the author** and a **customer** can be added. The projectname has to be unique and must not contain any special characters.

2.5.2 USB-Connection

With the menu-item **connection** your PC is connected with MicroFM, if a USB cable is plugged in. You don't need to connect MicroFM with the EIB/KNX- network.

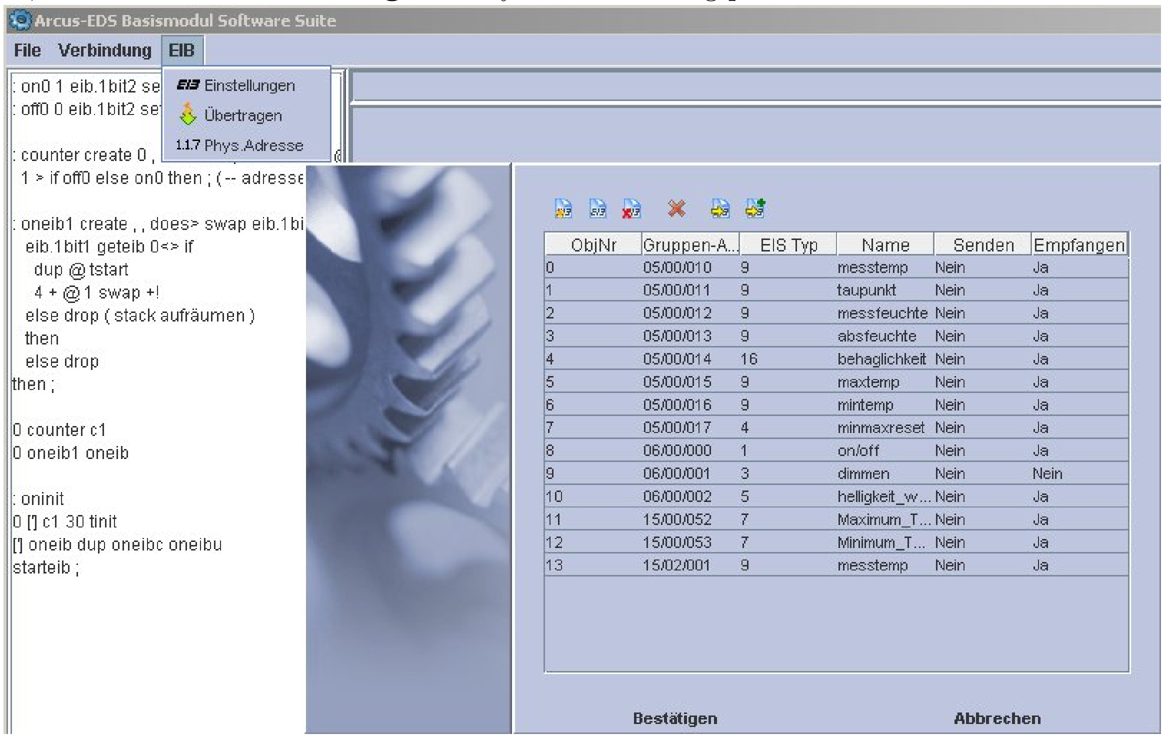


By **open connection** the connection is carried out, the MicroFM answers **'connected'** and the information connected shows up at the bottom of the display. Now you can communicate with MicroFM. When you start the also provided USB-server by a remote computer, you can have access to it and to a connected MicroFM over extern connection. You can use it as it were locally connected with your PC .

2.5.3 EIB-Settings

At this point you need knowledge about the European Installation Bus (EIB). Ask your system integrator or your EIB service-provider.
64 EIB Group Addresses can be used. The import of Group Addresses is carried out by direct selection

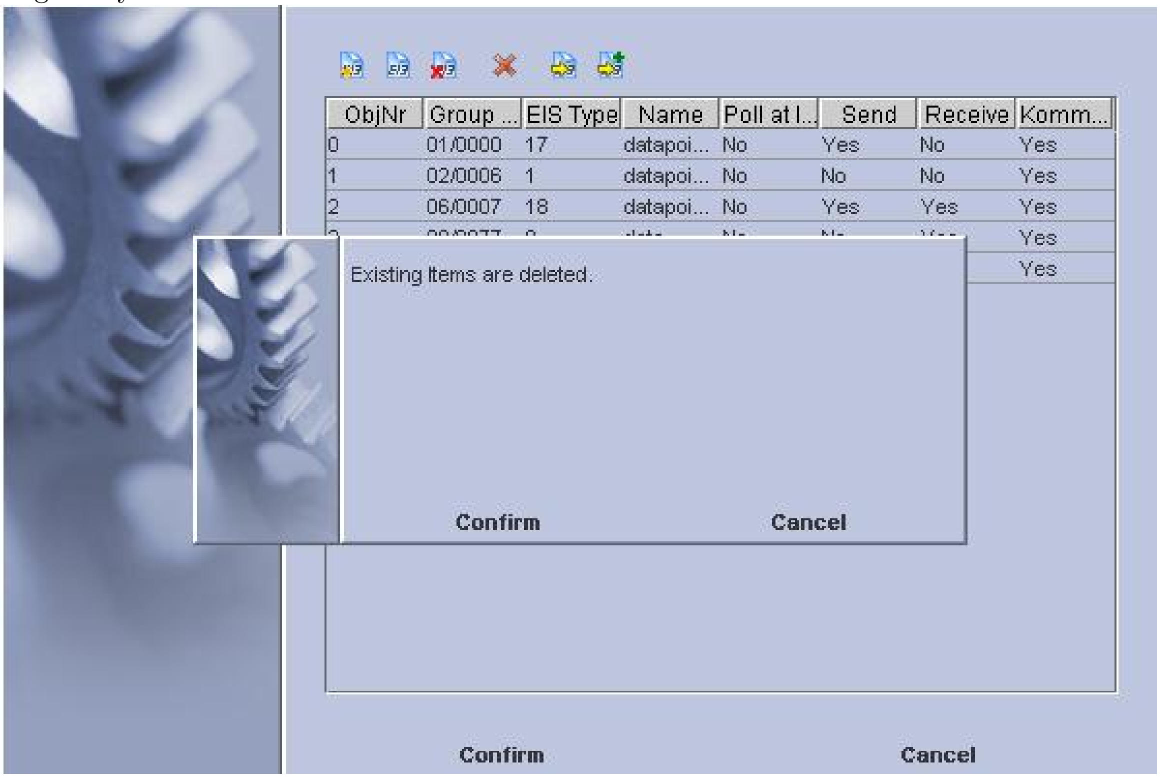
from the ETS3 database.
Each Datapoint can also be entered and processed separate manually. As soon as you have opened a project, the menuitem **EIB settings** offers you the following picture:



```
: on0 1 eib.1bit2 se
: off0 0 eib.1bit2 se

: counter create 0 ,
 1 > if off0 else on0 then ; ( -- adresse

: oneib1 create , , does> swap eib.1bi
  eib.1bit1 geteib 0<> if
   dup @ tstart
   4 + @ 1 swap +!
  else drop ( stack aufräumen )
  then
  else drop
then ;

0 counter c1
0 oneib1 oneib

: oninit
0 ['] c1 30 tinit
['] oneib dup oneibc oneibu
starteib ;
```

ObjNr	Gruppen-A...	EIS Typ	Name	Senden	Empfangen
0	05/00/010	9	messtemp	Nein	Ja
1	05/00/011	9	taupunkt	Nein	Ja
2	05/00/012	9	messfeuchte	Nein	Ja
3	05/00/013	9	absfeuchte	Nein	Ja
4	05/00/014	16	behaglichkeit	Nein	Ja
5	05/00/015	9	maxtemp	Nein	Ja
6	05/00/016	9	mintemp	Nein	Ja
7	05/00/017	4	minmaxreset	Nein	Ja
8	06/00/000	1	on/off	Nein	Ja
9	06/00/001	3	dimmen	Nein	Nein
10	06/00/002	5	helligkeit_w...	Nein	Ja
11	15/00/052	7	Maximum_T...	Nein	Ja
12	15/00/053	7	Minimum_T...	Nein	Ja
13	15/02/001	9	messtemp	Nein	Ja

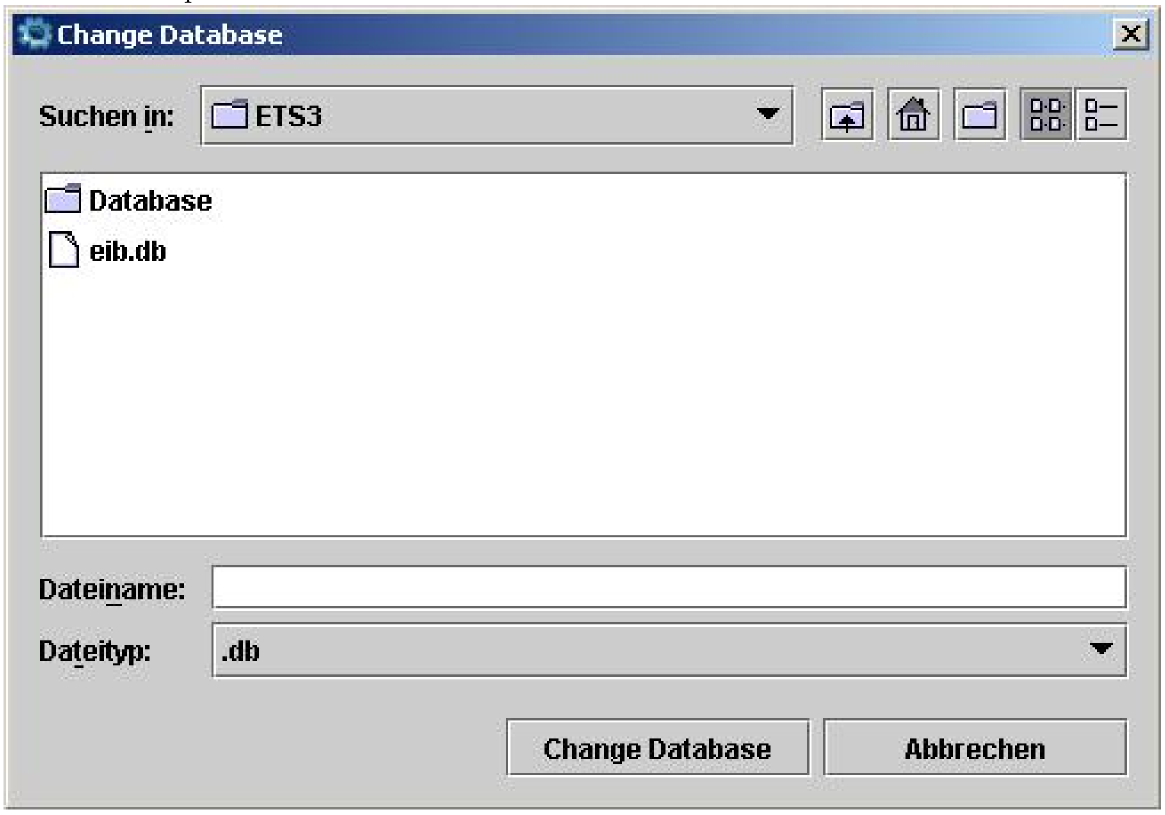
By a new import the complete EIB- Device list will be overwritten. Chose your respective ETS - Database when carrying out the first import. This entry is then stored in the settings and as well can be changed any time.



ObjNr	Group ...	EIS Type	Name	Poll at I...	Send	Receive	Komm...
0	01/0000	17	datapoi...	No	Yes	No	Yes
1	02/0006	1	datapoi...	No	No	No	Yes
2	06/0007	18	datapoi...	No	Yes	Yes	Yes

Existing Items are deleted.

Now choose the respective ETS3 database.



Change Database

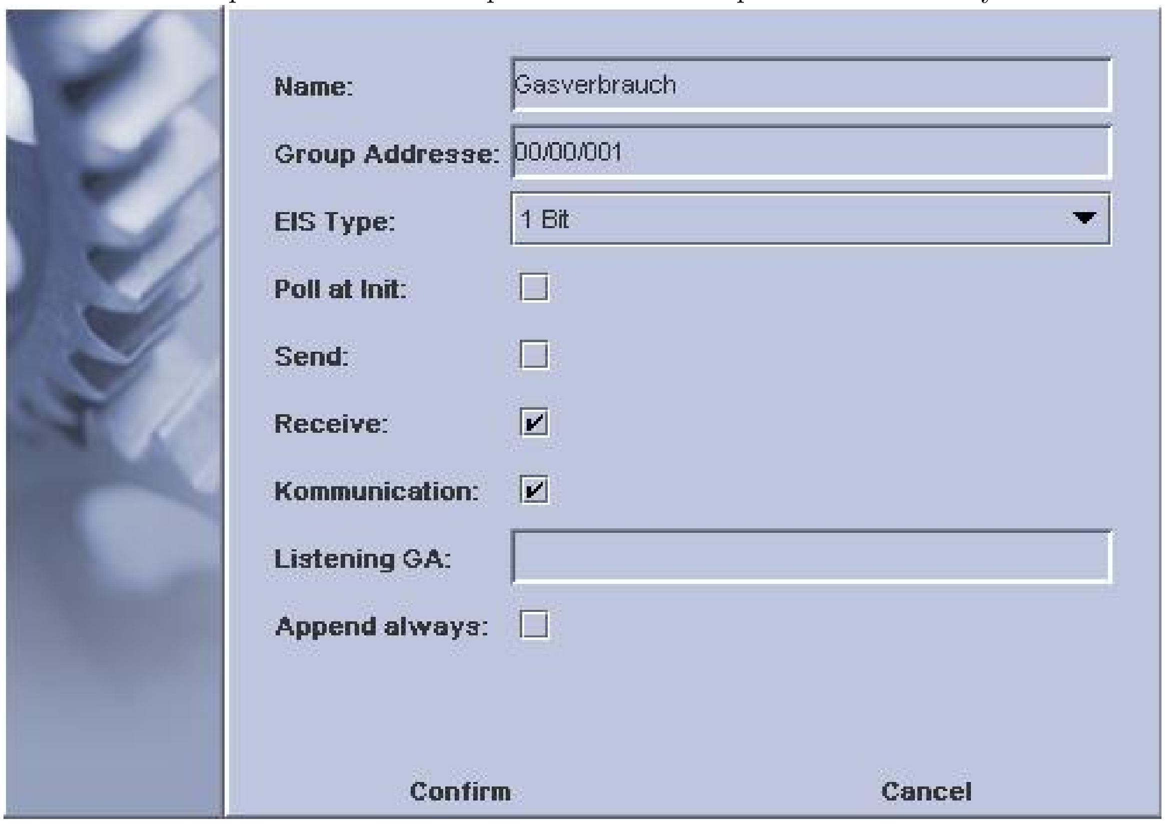
Suchen in: ETS3

Database
eib.db

Dateiname:

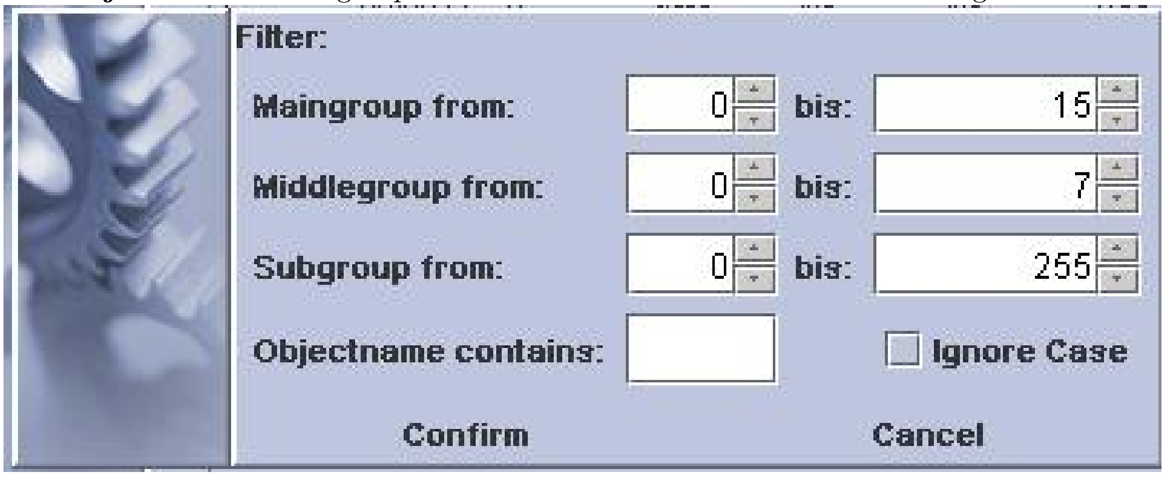
Dateityp: .db

You can confirm or skip some of the data- points as well as import all immediately.



Name:	Gasverbrauch
Group Addresse:	00/00/001
EIS Type:	1 Bit
Poll at Init:	☐
Send:	☐
Receive:	☑
Kommunication:	☑
Listening GA:	
Append always:	☐

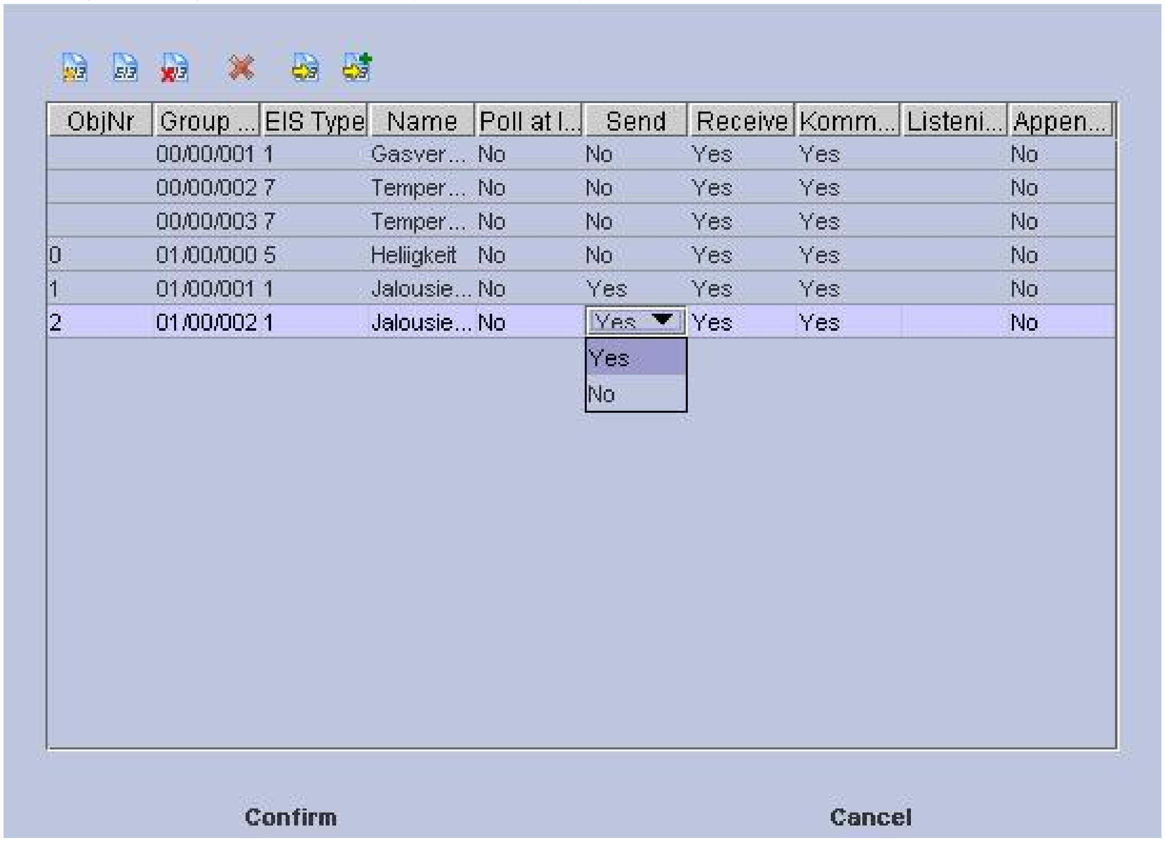
The selective data- import can be carried out by pressing the button:**‘Add EIB- objects filtered**’ out of the ETS over the primary-, secondary - or sub - groups or as well with the fulltext search **contains object- name**. The group- addresses will then be added to the existing list.



Filter:

Maingroup from:	0	bis:	15
Middlegroup from:	0	bis:	7
Subgroup from:	0	bis:	255
Objectname contains:		☐ Ignore Case	

With ‘change settings and database’ you can always choose other EIB databases for import.



ObjNr	Group ...	EIS Type	Name	Poll at I...	Send	Receive	Komm...	Listeni...	Appen...
	00/00/001	1	Gasver...	No	No	Yes	Yes		No
	00/00/002	7	Temper...	No	No	Yes	Yes		No
	00/00/003	7	Temper...	No	No	Yes	Yes		No
0	01/00/000	5	Heligkeit	No	No	Yes	Yes		No
1	01/00/001	1	Jalousie...	No	Yes	Yes	Yes		No
2	01/00/002	1	Jalousie...	No	Yes	Yes	Yes		No

With the menu-item **physical address** you can enter the **unitary** address of the MicroFM. With the item **transmit** the settings are transmitted to MicroFM. Before any access to the objects, the MicroFM has once to be rebooted (command: **reset** in the Commandline)

2.5.4 Command Interface

Enter the commands to your MicroFM in the **Commandline** . The answer of the MicroFM appears on the **Receive page**.
The programming of MicroFM is processed in a **FORTH** dialect. You can get further information about **Forth**, the syntax and about the stack- and heap- usage from literature reference and/or the internet.
In using this language for MicroFM you have the advantage of its compact, fast code and self-compiling attributes. Functions programmed and commands entered are executed by multitasking. As there exists only one common stack, every function processed in the background has to take care of an empty stack.
A short summary of the functions available you will find in the chapter: *Language Components.*
We wish you a lot of fun with your MicroFM

Your team of developer of Arcus-EDS GmbH

2.5.5 Language Components

Basic Functions					
+	n1 n2	→	n3	n3 = n1 + n2	Adds n1 and n2 and leaves sum on stack.
-	n1 n2	→	n3	n3 = n1 - n2	Subtracts n2 from n1 and leaves difference on stack.
*	n1 n2	→	n3	n3 = n1 * n2	Multiplies n1 and n2 and leaves product on stack.
/	n1 n2	→	n3	n3 = n1 / n2	Divides n1 by n2 and leaves quotient on stack.
’ word		→	caddr	Obtain compilation address	Places the compilation address of the following word on the stack.
,	n	→		Store in heap	Reserves four bytes of heap space, initialising it to n.
.	n	→		Print top of stack	Prints the number on the top of the stack.
.(str		→		Print constant string	Immediately prints the string that follows in the input stream.
.S		→		Print stack	Prints entire contents of stack.
." str		→		Print immediate string	Prints the string literal that follows in line.
: w		→		Begin definition	Begins compilation of a word named w.
;		→		End definition	Ends compilation of word.
<	n1 n2	→	flag	Less than	Returns -1 if n1<n2, 0 otherwise.
<=	n1 n2	→	flag	Less than or equal	Returns -1 if n1 $\leq$ n2, 0 otherwise.
<>	n1 n2	→	flag	Not equal	Returns -1 if n1 $\neq$ n2, 0 otherwise.

=	n1 n2	→	flag	Equal	Returns -1 if n1=n2, 0 otherwise.
>	n1 n2	→	flag	Greater	Returns -1 if n1>n2, 0 otherwise.
>=	n1 n2	→	flag	Greater than or equal	Returns -1 if n1 $\geq$ n2, 0 otherwise.
?	addr	→		Print indirect	Prints the value at the address at the top of the stack.
!	n addr	→		Store into address	Stores the value n into the address addr.
+!	n addr	→		Add indirect	Adds n to the word at address addr.
@	addr	→	n	Load	Loads the value at addr and leaves it at the top of the stack.
[		→		Set interpretive state	Within a compilation, returns to the interpretive state.
['] word		→	caddr	Push next word	Places the compile address of the following word in a definition onto the stack.
]		→		End interpretive state	Restore compile state after temporary interpretive state.
0<	n1	→	flag	Less than zero	Returns -1 if n1 less than zero, 0 otherwise.
0<>	n1	→	flag	Nonzero	Returns -1 if n1 is nonzero, 0 otherwise.
0=	n1	→	flag	Equal to zero	Returns -1 if n1 is zero, 0 otherwise.
0>	n1	→	flag	Greater than zero	Returns -1 if n1 greater than zero, 0 otherwise.
1+	n1	→	n2	Add one	Adds one to top of stack.
1-	n1	→	n2	Subtract one	Subtracts one from top of stack.
2+	n1	→	n2	Add two	Adds two to top of stack.
2-	n1	→	n2	Subtract two	Subtracts two from top of stack.

2*	n1	→	n2	Times two	Multiplies the top of stack by two.
2/	n1	→	n2	Divide by two	Divides top of stack by two.
2!	n1 n2 addr	→		Store two words	Stores the two words n1 and n2 at addresses addr and addr+4.
2@	addr	→	n1 n2	Load two words	Places the two words starting at addr on the top of the stack
2CONSTANT x	n1 n2	→		Double word constant	Declares a double word constant x. When x is executed, n1 and n2 are placed on the stack.
2DROP	n1 n2	→		Double drop	Discards the two top items from the stack.
2DUP	n1 n2	→	n1 n2 n1 n2	Duplicate two	Duplicates the top two items on the stack.
2OVER	n1 n2 n3 n4	→	n1 n2 n3 n4 n1 n2	Double over	Copies the second pair of items on the stack to the top of stack.
2ROT	n1 n2 n3 n4 n5 n6	→	n3 n4 n5 n6 n1 n2	Double rotate	Rotates the third pair on the stack to the top, moving down the first and second pairs.
2SWAP	n1 n2 n3 n4	→	n3 n4 n1 n2	Double swap	Swaps the first and second pairs on the stack.
2VARIABLE x		→		Double variable	Creates a two cell (8 byte) variable named x. When x is executed, the address of the 8 byte area is placed on the stack.
ABORT		→		Abort	Clears the stack and performs a QUIT.
ABORT" str		→		Abort with message	Prints the string literal that follows in line, then aborts, clearing all execution state to return to the interpreter.
ABS	n1	→	n2	n2=|n1|	Replaces top of stack with its absolute value.

ACOS	f1	→	f2	f2=arccos f1	Replaces floating point top of stack with its arc cosine.
AGAIN		→		Indefinite loop	Marks the end of an indefinite loop opened by the matching BEGIN.
ALLOT	n	→		Allocate heap	Allocates n bytes of heap space. The space allocated is rounded to the next higher multiple of 4.
AND	n1 n2	→	n3	Bitwise and	Stores the bitwise and of n1 and n2 on the stack.
ARRAY x	s1 s2 ¼ sn n esize	→		Declare array	Declares an array x of elements of esize bytes each with n subscripts, each ranging from 0 to sn-1.
ASIN	f1	→	f2	f2=arcsin f1	Replaces floating point top of stack with its arc sine.
ATAN	f1	→	f2	f2=arctan f1	Replaces floating point top of stack with its arc tangent.
ATAN2	f1 f2	→	f3	f3=arctan f1/ f2	Replaces the two floating point numbers on the top of the stack with the arc tangent of their quotient, properly handling zero denominators.
BEGIN		→		Begin loop	Begins an indefinite loop. The end of the loop is marked by the matching AGAIN, REPEAT, or UNTIL.
>BODY	cfa	→	pfa	Body address	Given the compile address of a word, return its body (parameter) address.
BRANCH		→		Branch	Jump to the address that follows in line.

?BRANCH	flag	→		Conditional branch	If the top of stack is zero, jump to the address which follows in line. Otherwise skip the address and continue execution.
C!	n addr	→		Store byte	The 8 bit value n is stored in the byte at address addr.
C@	addr	→	n	Load byte	The byte at address addr is placed on the top of the stack.
C,	n	→		Compile byte	The 8 bit value n is stored in the next free byte of the heap and the heap pointer is incremented by one.
C=		→		Align heap	The heap allocation pointer is adjusted to the next four byte boundary. This must be done following a sequence of C, operations.
CLEAR		→		Clear stack	All items on the stack are discarded.
COMPARE	s1 s2	→	n	Compare strings	The two strings whose addresses are given by s1 and s2 are compared. If s1 is less than s2, -1 is returned; if s1 is greater than s2, 1 is returned. If s1 and s2 are equal, 0 is returned.
CONST x	n	→		Declare constant	Declares a constant named x. When x is executed, the value n will be left on the stack.
COS	f1	→	f2	Cosine	The floating point value on the top of the stack is replaced by its cosine.
CR		→		Carriage return	The standard output stream is advanced to the first character of the next line.

CREATE		→		Create object	Create an object, given the name which appears next in the input stream, with a default action of pushing the parameter field address of the object when executed. No storage is allocated; normally the parameter field will be allocated and initialised by the defining word code that follows the CREATE.
DEPTH		→	n	Stack depth	Returns the number of items on the stack before DEPTH was executed.
DO	limit n	→		Definite loop	Executes the loop from the following word to the matching LOOP or +LOOP until n increments past the boundary between limit-1 and limit. Note that the loop is always executed at least once (see ?DO for an alternative to this).
?DO	limit n	→		Conditional loop	If n equals limit, skip immediately to the matching LOOP or +LOOP. Otherwise, enter the loop, which is thenceforth treated as a normal DO loop.
DOES>		→		Run-time action	Sets the run-time action of a word created by the last CREATE to the code that follows. When the word is executed, its body address is pushed on the stack, then the code that follows the DOES>; will be executed.

DROP	n	→		Discard top of stack	Discards the value at the top of the stack.
DUP	n	→	n n	Duplicate	Duplicates the value at the top of the stack.
?DUP	n	→	0 / n n	Conditional duplicate	If top of stack is nonzero, duplicate it. Otherwise leave zero on top of stack.
ELSE		→		Else	Used in an IF-ELSE-THEN sequence, delimits the code to be executed if the if-condition was false.
EXECUTE	addr	→		Execute word	Executes the word with compile address addr.
EXIT		→		Exit definition	Exit from the current definition immediately. Note that EXIT cannot be used within a DO-LOOP; use LEAVE instead.
EXP	f1	→	f2	f2=ef1	The floating point value on the top of the stack is replaced by its natural antilogarithm.
F+	f1 f2	→	f3	f3=f1+f2	The two floating point values on the top of the stack are added and their sum is placed on the top of the stack.
F-	f1 f2	→	f3	f3=f1-f2	The floating point value f2 is subtracted from the floating point value f1 and the result is placed on the top of the stack.
F*	f1 f2	→	f3	f3=f1´f2	The two floating point values on the top of the stack are multiplied and their product is placed on the top of the stack.

F/	f1 f2	→	f3	f3=f1¸f2	The floating point value f1 is divided by the floating point value f2 and the quotient is placed on the top of the stack.
F.	f	→		Print floating point	The floating point value on the top of the stack is printed.
F<	f1 f2	→	flag	Floating less than	The top of stack is set to -1 if f1 is less than f2 and 0 otherwise.
F<=	f1 f2	→	flag	Floating less than or equal	The top of stack is set to -1 if f1 is less than or equal to f2 and 0 otherwise.
F<>	f1 f2	→	flag	Floating not equal	The top of stack is set to -1 if f1 is not equal to f2 and 0 otherwise.
F=	f1 f2	→	flag	Floating equal	The top of stack is set to -1 if f1 is equal to f2 and 0 otherwise.
F>	f1 f2	→	flag	Floating greater than	The top of stack is set to -1 if f1 is greater than f2 and 0 otherwise.
F>=	f1 f2	→	flag	Floating greater than or equal	The top of stack is set to -1 if f1 is greater than or equal to f2 and 0 otherwise.
FABS	f1	→	f2	f2=|f1|	Replaces floating point top of stack with its absolute value.

FIND	s	→	word flag	Look up word	The word with name given by the string s is looked up in the dictionary. If a definition if not found, word will be left as the address of the string and flag will be set to zero. If the word is present in the dictionary, its compilation address is placed on the stack, followed by a flag that is 1 if the word is marked for immediate execution and -1 otherwise.
FIX	f	→	n	Floating to integer	The floating point number on the top of the stack is replaced by the integer obtained by truncating its fractional part.
(FLIT)		→	f	Push floating point literal	Pushes the floating point literal that follows in line onto the top of the stack.
FLOAT	n	→	f	Integer to floating	The integer value on the top of the stack is replaced by the equivalent floating point value.
FMAX	f1 f2	→	f3	Floating point maximum	The greater of the two floating point values on the top of the stack is placed on the top of the stack.
FMIN	f1 f2	→	f3	Floating point minimum	The lesser of the two floating point values on the top of the stack is placed on the top of the stack.
FNEGATE	f1	→	f2	f2 = -f1	The negative of the floating point value on the top of the stack replaces the floating point value there.

FORGET w		→		Forget word	The most recent definition of word w is deleted, along with all words declared more recently than the named word.
HERE		→	addr	Heap address	The current heap allocation address is placed on the top of the stack.
I		→	n	Inner loop index	The index of the innermost DO-LOOP is placed on the stack.
IF	flag	→		Conditional statement	If flag is nonzero, the following statements are executed. Otherwise, execution resumes after the matching ELSE clause, if any, or after the matching THEN.
IMMEDIATE		→		Mark immediate	The most recently defined word is marked for immediate execution it will be executed even if entered in compile state.
J		→	n	Outer loop index	The loop index of the next to innermost DO-LOOP is placed on the stack.
LEAVE		→		Exit DO-LOOP	The innermost DO-LOOP is immediately exited. Execution resumes after the LOOP statement marking the end of the loop.
(LIT)		→	n	Push literal	Pushes the integer literal that follows in line onto the top of the stack.
LOG	f1	→	f2	f2=ln f1	The floating point value on the top of the stack is replaced by its natural logarithm.

LOOP		→		Increment loop index	Adds one to the index of the active loop. If the limit is reached, the loop is exited. Otherwise, another iteration is begun.
+LOOP	n	→		Add to loop index	Adds n to the index of the active loop. If the limit is reached, the loop is exited. Otherwise, another iteration is begun.
MAX	n1 n2	→	n3	Maximum	The greater of n1 and n2 is left on the top of the stack.
MEMSTAT		→		Print memory status	The current and maximum memory usage so far are printed on standard output. The sizes allocated for the stack, return stack, and heap are edited, as well as the percentage in use.
MIN	n1 n2	→	n3	Minimum	The lesser of n1 and n2 is left on the top of the stack.
MOD	n1 n2	→	n3	Modulus (remainder)	The remainder when n1 is divided by n2 is left on the top of the stack.
/MOD	n1 n2	→	n3 n4	n3 = n1 mod n2, n4 = n1 ¸ n2	Divides n1 by n2 and leaves quotient on top of stack, remainder as next on stack.
NEGATE	n1	→	n2	n2=-n1	Negates the value on the top of the stack.
(NEST)		→		Invoke word	Pushes the instruction pointer onto the return stack and sets the instruction pointer to the next word in line.

NOT	n1	→	n2	Logical not	Inverts the bits in the value on the top of the stack. This performs logical negation for truth values of -1 (True) and 0 (False).
OR	n1 n2	→	n3	Bitwise or	Stores the bitwise or of n1 and n2 on the stack.
OVER	n1 n2	→	n1 n2 n1	Duplicate second item	The second item on the stack is copied to the top.
PICK	n2 n1 n0 index	→	n0 nindex	Pick item from stack	The indexth stack item is copied to the top of the stack. The top of stack has index 0, the second item index 1, and so on.
POW	f1 f2	→	f3	f3=f1f2	The second floating point value on the stack is taken to the power of the top floating point stack value and the result is left on the top of the stack.
QUIT		→		Quit execution	The return stack is cleared and control is returned to the interpreter. The stack is not disturbed.
>R	n	→		To return stack	Removes the top item from the stack and pushes it onto the return stack.
R>		→	n	From return stack	The top value is removed from the return stack and pushed onto the stack.
R@		→	n	Fetch return stack	The top value on the return stack is pushed onto the stack. The value is not removed from the return stack.

REPEAT		→		Close BEGIN-WHILE-REPEAT loop	Another iteration of the current BEGIN-WHILE-REPEAT loop having been completed, execution continues after the matching BEGIN.
ROLL	n2 n1 n0 index	→	n0 nindex	Rotate indexth item to top	The stack item selected by index, with 0 designating the top of stack, 1 the second item, and so on, is moved to the top of the stack. The intervening stack items are moved down one item.
ROT	n1 n2 n3	→	n2 n3 n1	Rotate 3 items	The third item on the stack is placed on the top of the stack and the second and first items are moved down.
-ROT	n1 n2 n3	→	n3 n1 n2	Reverse rotate	Moves the top of stack to the third item, moving the third and second items up.
S!	s1 s2	→		Store string	The string at address s1 is copied into the string at s2.
S+	s1 s2	→		String concatenate	The string at address s1 is concatenated to the string at address s2.
SHIFT	n1 n2	→	n3	Shift n1 by n2 bits	The value n1 is logically shifted the number of bits specified by n2, left if n2 is positive and right if n2 is negative. Zero bits are shifted into vacated bits.
SIN	f1	→	f2	Sine	The floating point value on the top of the stack is replaced by its sine.
SQRT	f1	→	f2	f2 = sqrt f1	The floating point value on the top of the stack is replaced by its square root.

STATE		→	addr	System state variable	The address of the system state variable is pushed on the stack. The state is zero if interpreting, nonzero if compiling.
STRCAT	s1 s2	→		String concatenate	The string at address s1 is concatenated to the string at address s2.
STRCHAR	s1 s2	→		String character search	The string at address s1 is searched for the first occurrence of the first character of string s2. If that character appears nowhere in s1, 0 is returned. Otherwise, the address of the first occurrence in s1 is left on the top of the stack.
STRCMP	s1 s2	→	n	String compare	The string at address s1 is compared to the string at address s2. If s1 is less than s2, -1 is returned. If s1 and s2 are equal, 0 is returned. If s1 is greater than s2, 1 is returned.
STRCPY	s1 s2	→		Store string	The string at address s1 is copied into the string at s2.
STRING x	size	→		Declare string	Declares a string named x of a maximum of size-1 characters.
STRINT	s1	→	s2 n	String to integer	Scans an integer from s1. The integer scanned is placed on the top of the stack and the address of the character that terminated the scan is stored as the next item on the stack.
STRLEN	s	→	n	String length	The length of string s is placed on the top of the stack.

(STRLIT)		→	s	String literal	Pushes the address of the string literal that follows in line onto the stack.
STRREAL	s1	→	s2 f	String to real	Scans a floating point number from s1. The floating point number scanned is placed on the top of the stack and the address of the character that terminated the scan is stored as the next item on the stack.
SWAP	n1 n2	→	n2 n1	Swap top two items	The top two stack items are interchanged.
TAN	f1	→	f2	Tangent	The floating point value on the top of the stack is replaced by its tangent.
THEN		→		End if	Used in an IF-ELSE-THEN sequence, marks the end of the conditional statement.
TRACE	n	→		Trace mode	If n is nonzero, trace mode is enabled. If n is zero, trace mode is turned off.
TYPE	s	→		Print string	The string at address s is printed on standard output.
UNTIL	flag	→		End BEGIN-UNTIL loop	If flag is zero, the loop continues execution at the word following the matching BEGIN. If flag is nonzero, the loop is exited and the word following the UNTIL is executed.
VAR x		→		Declare variable	A variable named x is declared and its value is set to zero. When x is executed, its address will be placed on the stack. Four bytes are reserved on the heap for the variable's value.

WALKBACK	n	→		Walkback mode	If n is nonzero, a walkback trace through active words will be performed whenever an error occurs during execution. If n is zero, the walkback is suppressed.
WHILE	flag	→		Decide BEGIN-WHILE-REPEAT loop	If flag is nonzero, execution continues after the WHILE. If flag is zero, the loop is exited and execution resumed after the REPEAT that marks the end of the loop.
WORDS		→		List words defined	Defined words are listed, from the most recently defined to the first defined. If the system supports keystroke trapping, pressing any key will pause the display of defined words pressing carriage return will abort the listing-any other key resumes it. On other systems, only the 20 most recently defined words are listed.
XOR	n1 n2	→	n3	Bitwise exclusive or	Stores the bitwise exclusive or of n1 and n2 on the stack.
(XDO)	limit n	→		Execute loop	At runtime, enters a loop that will step until n increments and becomes equal to limit.
(X?DO)	limit n	→		Execute conditional loop	At runtime, tests if n equals limit. If so, skips until the matching LOOP or +LOOP. Otherwise, enters the loop.

(XLOOP)		→		Increment loop index	At runtime, adds one to the index of the active loop and exits if equal to the limit. Otherwise returns to the matching DO or ?DO.
(+XLOOP)	incr	→		Add to loop index	At runtime, increments the loop index by the top of stack. If the loop is not done, begins the next iteration.

Components for MicroFM/MicroVIS					
Time and Date Functions					
SYST		→	Time	Get Systemtime	Returns the actual System Time
SETSYST	TIME	→		Set Systemtime	Sets the Systemtime
DTIME	TIME	→	WD,H,M,S	Decode Time	Splits the 32-Bit Timevalue into 4 values: Weekday, Hour, Minute, Second
CTIME	WD H M S	→	TIME	Code Time	Concantenates the values for Weekday, Hour, Minute, Second to one 32-Bit Time-value
SYSD		→	Date	Get Systemdate	Returns the actual System Date
SETSYSD	DATE	→		Set Systendate	Sets the Systemdate
DDATE	DATE	→	J M D	Decode Date	Splits the 32-Bit Datevalue into 3 values: Jear Month and Day
CDATE	J M D	→	Date	Code Date	Concantenates the values for year Month and Day to one 32-Bit Date-value
TICKS		→	Ticks	Get Systemticks	Get the Number of Milliseconds the device is running and Timerfunctions (Resolution is 10ms)

TINIT	Nr Function Timeout	→		Init Timer	Init Timer Nr (0 to 15) with timeout and let it execute the function after timeout. Timeout is set in 1/100 second intervalls
TSTART	Nr	→		Start Timer	Starts the Timer Nr (0 to 15)
TSTOP	Nr	→	Flag	Stop Timer	Stops the Timer, Flag is 0 if timer was running, -1 otherwise
TCONT	NR	→	Rest	Continue Timer	Timer is restarted without reset, this function returns the rest amount of time, even if the timer was not stopped before
EIB-Functions					
ONEIBC	Address	→		On eib change	Defines a function to be executed by an incoming eib telegram which changes the value of an element
ONEIBU	Address	→		On eib update	Defines a function to be executed by an incoming eib telegram which does not chang the value of an element
STARTEIB		→			
Activate EIB functions					
STOPEIB		→			Stop EIB functions
EIBGET	Objnr	→	Value	Get EIB Value	Get the value from object and push it on the stack.
EIBSET	Value, Objnr	→		Set EIB value	Set the object to value
EIBTX	Objnr	→		Send value	Initiates a send of the objects value
EIBPOLL	OBJNR	→		Poll EIB value	Requests the value from the bus

EIS2INT	Value	→	Value	Convert 2-Byte- float value to integer	Multiplies the 2-byte float with 100 and makes it integer
INT2EIS	Value	→	Value	Convert Integer to 2-Byte float	Divides the integer by 100 and makes it 2-byte-float
FLOAT2DBL	4-Byte-Float	→	8-Byte-Double	Convert Float to Double	
DBL2FLOAT	8-Byte-Double	→	4-Byte-Float	Convert Double to Float	
Memory Management					
FSAVE		→		Save Heap	Saves all functions and data defined in the heap into non-volatile memory , so that the heap is present on the next startup
FERASE		→		Clear NVM	Clears the nonvolatile memory
XLOCK		→		Lock program	Locks all console input exept ferase
With FERASE all memory is cleared and the Locking is removed					
System-Extensions					
SIGNAL	Function	→		Execute Function asynchronous	Send a signal to the system, that the function should be executed asynchronous
SELF		→	Addr	Return Runtime-address	The address of the running Function is pushed on the stack. Used inside a create does> statement, you get the functions adress in the create statement
RESET		→		Reset System	Systemreset and restart
PRINT	Addr n	→	Addr	Print integer	Print integer n in readable form

FPRINT	Addr f	→	Addr	Print double	Print double f in readable form
MicroFM-Extensions					
GETSW		→	Value	Get switch	Polls the value of the switch
SETLED	0/1 nr	→		Set LED	Sets led nr on or off
MicroVIS-Extensions					
PIXEL	x y color	→		Print Pixel	Print a Pixel on Position x,y with color
RECT	x y w h color	→		Print Rectangle	Print a Rectangle on Position x,y with width w hight h and color
DISPSTR	straddr elementdef mask	→		Print String	
CIRCLE	x y d color	→		Print Circle	
LINE	x1 y1 x2 y2 w color	→		Print Line	
SIGNALOBJ	objnr	→		send change event	Send a Signal to the user interface indicating that data has changed on this object
OBJTYPE	objnr	→		get eib-type of object	Determine the kind of object-data from object
CONSUME		→		consume operation event	Inhibit execution of the current operation event by the user interface
BEEP	flag	→		set buzzer state	if flag=0 set buzzer of else on
SETBRIGHT	brightness	→		set brightness	set the current brightness to a value between 0 and 255
STOREBRIGHT		→		store brightness	Store the actual brightness value into eeprom

SETCONTRAST	contrast	→		set contrast	set the current contrast to a value between 0 and 255
STORECONTRAST		→		store brightness	Store the actual contrast value into eeprom
SETTIMEOUT		→		set timeout	Set the actual timeout value
WAKEUP		→		wakeup	wakeup from sleep mode
JUMPPAGE	pagenr	→		jump to page	jump to page nr pagenr if pagenr>;0 else jump back from current page
Special Functions					
ONPB (MicroFM only)					Function to be defined that is executed by pressing the button
ONINIT					Function to be defined that is executed at startup

2.5.6 Examples

First of all you would like to see something:

```
1 0 setled
```

Notice: the red LED (with the number 0) is lit.

```
0 0 setled
```

Now it's off again Now you don't want to enter anything, but to use the pushbutton:

```
var ledstat
: onpb ledstat @ 1 xor
dup ledstat ! 1 setled
;
```

When you press the key, the green LED goes on and then off. The function onpb will be completed automatically by pressing the button. What's going on, in details:

`: onpb`	→	start of function onpb
`ledstat @`	→	get the content of the variable ledstat
`1 xor`	→	exclusive or with 1, the value switches from 0 to 1
`dup`	→	the obtained value is duplicated on the stack.
`ledstat !`	→	the value will be written to the address of ledstat
`1 setled`	→	as well, the duplicated value will be written on led 1 (green)
`;`	→	end of definition

Now let’s forget the whole thing!

```
forget ledstat
```

Now let’s switch on/ off the LED governed by the rotary switch

```
: onpb getsw
dup 1 and 0= if 0 else 1
then 0 setled
dup 2 and 0= if 0 else 1
then 1 setled
4 and 0= if 0 else 1
then 2 setled ;
```

getsw gets the switch-setting, binary represented by the LEDs, when you press the key forget onpb deletes all again. Now we try to work with a timer:

```
: ledon 1 0 setled ;
: ledoff 0 0 setled ;
0 ’ ledoff 100 tinit
: onpb ledon 0 tstart ;
```

Operating the pushbutton, the LED goes on for a second. We recognise the first and second line as functions switching on and off the red LED. In the third line a timer will be initialised (number 0 out of 0 to15) which performs "ledoff" after 100/100 seconds. (The term " ’ ledoff" gets the runtime-address). In the fourth line the function onpb switches on the LED and starts the timer (number 0). If e.g., the timeout should be governed by the rotary-switch, it’s possible to declare the timer in runtime.

```
: onpb ledon 0 [’] ledoff getsw 100 * tinit 0 tstart ;
```

The rotary-switch determines now the amount of seconds. Notice the change [’] ledoff with which the runtime- address can be determined in functions to functions. Now enough about the operating control:

```
forget ledon
```

You bought the device because of the EIB- interface. Now construct (build up) a 1 bit group-object of your choice under EIB- settings (at best a light, which you have in your sight). Name it **testobj** without setting the flags (You don’t want to receive this object, and it should not answer any recalls from the bus). Transfer your new group- table to the device and carry out a reset. (enter:‘reset‘). With the new connection working, the term ‘connected’ shows once more. By now connect your MicroFM with the EIB/KNX- Bus.

```
 : on0 1 eib.testobj
eibset eib.testobj eibtx
;   on0
```

Your have switched on your light.

```
: off0 0 eib.testobj
eibset eib.testobj eibtx
;
off0
```

Now it’s off again. **EIBSET** sets the internal group- object and **EIBTX** sends the information to the bus.

To send is one thing, to receive another. Again add a 1 bit group- object (this time a pushbutton) under EIB-settings. Name it pushbutton 1 and set the receive-flag (we watch our flags by our MicroFM) Now again transmit (transfer) and reset.

```
var ledstat
: toggle ledstat @ 1 xor
dup ledstat ! 0 setled
;
```

You recognise an acquaintance : the red LED will be switched.

```
: eibin eib.Taster1
=  if toggle then ;
' eibin dup oneibc
oneibu
```

You defined a function, which tests if the stack-value at the top corresponds with the object-number of your pushbutton- signals. If this is the case, the LED will be switched (turned, shifted), and you have passed on the function-address of this function to the call parameter **oneibc** and **oneibu**.

So the function will be registered for incoming EIB-telegrams (both at changed values and at value-update). When a telegram is coming in, the object-number is written on the stack and the function **eibin** is carried out. Now all has only to be activated.

```
starteib
```

Now you can switch on/off your LED with your pushbutton. **stopeib** stops the reception again. If after a reset all should function again and start immediately, you have as well to define the start-function oninit.

```
: oninit ['] eibin dup oneibc oneibu starteib ;
```

Until now all is stored in RAM, but have now to be transferred to the persistent memory

```
fsave
```

Now take a look at the memory capacity:

```
memstat
```

The following output appears:

```
Stack:
Curr:  0 Items: 100 0 %
Return stack:
Curr:  0 Items: 100 0 %
Heap:
Curr:  49 Items: 2176 2 %
Flash:
Curr:  192 Items: 8192 2 %
```

2% of the memory is already used. After a reboot all appears as programmed before and your LED-switch is functioning without your assistance.

With **ferase** you can delete the persistent memory again. The device starts automatically anew. If you would like to protect the device against unauthorised access, type in

```
xlock
```

then you have furthermore console-output but no command input except **ferase**. If you really make a mistake by programming having no further access because the device says good bye directly after the start, you can start the device by down-pressing the pushbutton.

.

Now delete the bad program with **ferase** and begin anew.

2.5.7 Preprocessor

Each text, sent from the project - page, runs through a pre - processor, which recognises the following commands, not passed on to the MicroFM.
`#define xx yyy` Any xx occurring later on will be replaced with yyy.
`#include filename` The file filename will be read in. The file is searched in relation to the folder microfm/includes.
Any line, beginning with `#` otherwise will be ignored.

3 The MicroVis Module

3.1 Introduction

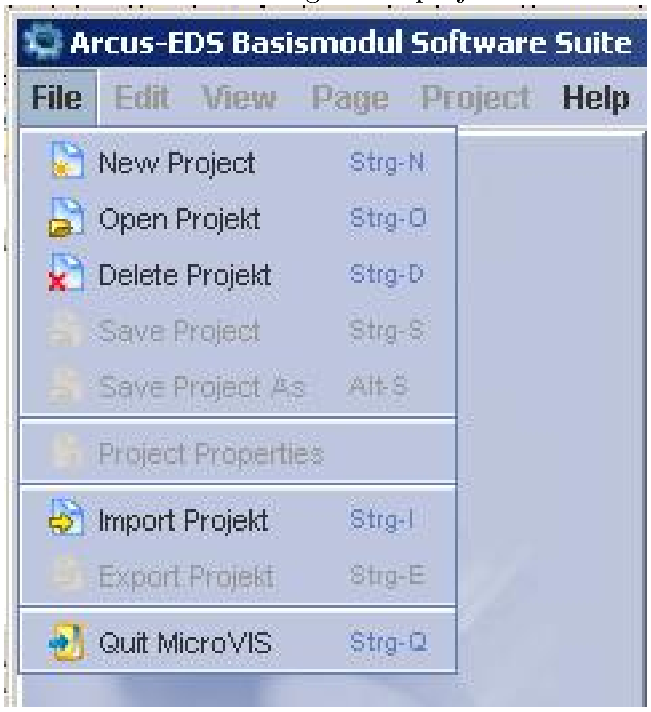
This document contains information such as data, illustrations, measurements and others which are subject to change without prior notice. Additional information is available at http://www.arcus-eds.de.
Technical changes preserved!
All product- descriptions used in this documentation are registered trademarks of the respective companies. No parts of this document must be copied or transmitted without the explicit permission of Arcus-eds GmbH, irrespective of the purpose or the way in which it is carried out.
All rights reserved!
©by Arcus-EDS GmbH

3.2 Systeminformation

This device is a product for the Instabus- EIB / KNX- System. Detailed knowledge in depth of the Instabus- EIB/KNX- System is essential. The functions of the device are software- depending. Detailed information, which software can be loaded and which function- capacity is then available as well as information about the software itself, have a look at the software- details of the manufacturer.
It is operated with the software tool 'ARC Suite' and is ready for downloading under http://www.arcus-eds.de.
This device is working with a real- time operating system FreeRTOS (www.freertos.org).

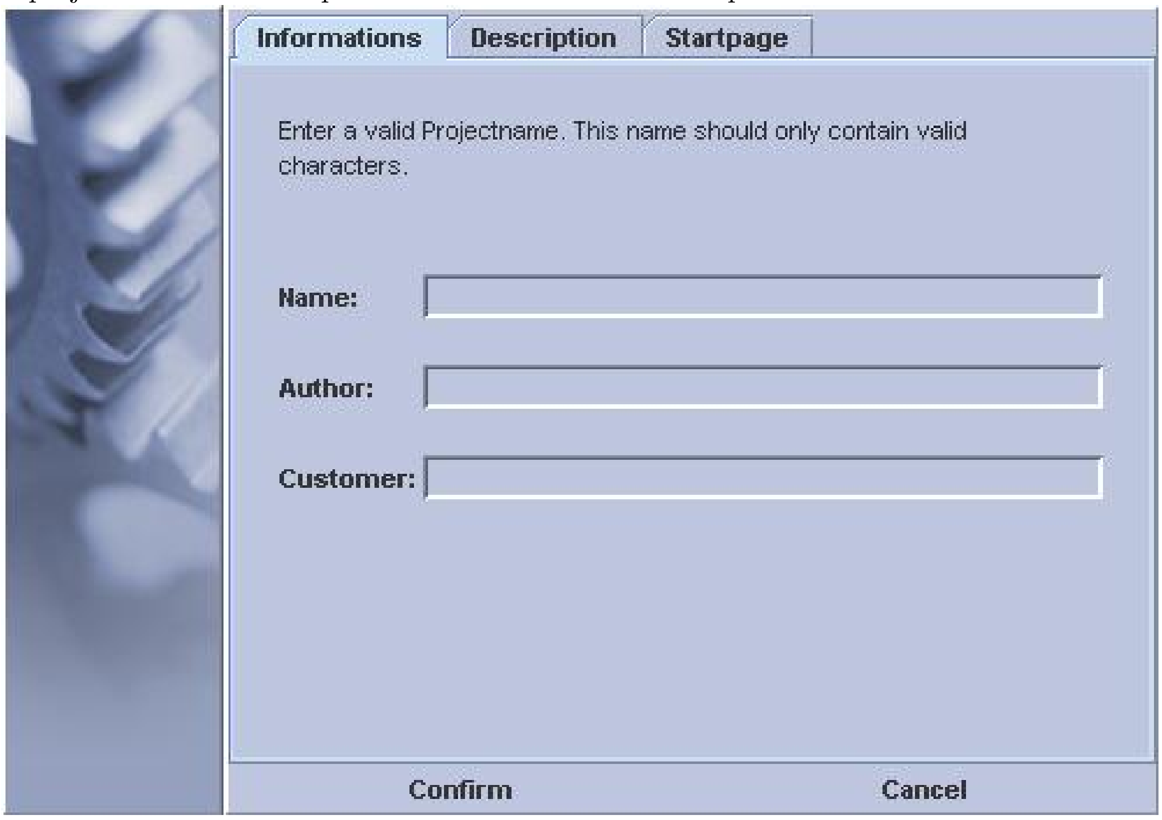
3.3 Project Administration

In the menu item **File** you find the tools referring to the project administration.



New project:

A new project demands a unique name and contains further optional instructions.



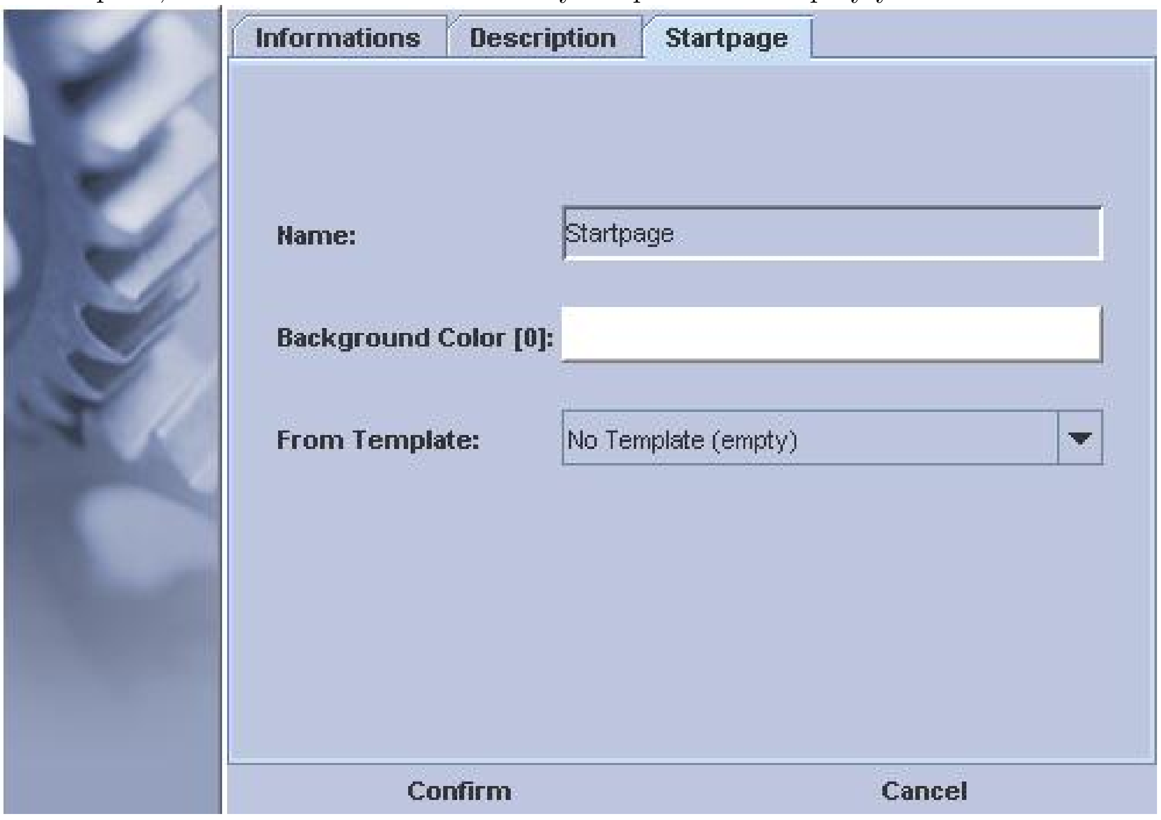
Enter a valid Projectname. This name should only contain valid characters.

Name:	
Author:	
Customer:	

Under its project name the project can be selected or deleted later.

The entry of author's and client's name or a description is optional.

The instruction for the start page enables you to enter the selected background colour as well as the use of a blueprint, which can be selected from any template drawn up by you.



Name: Startpage

Background Color [0]:

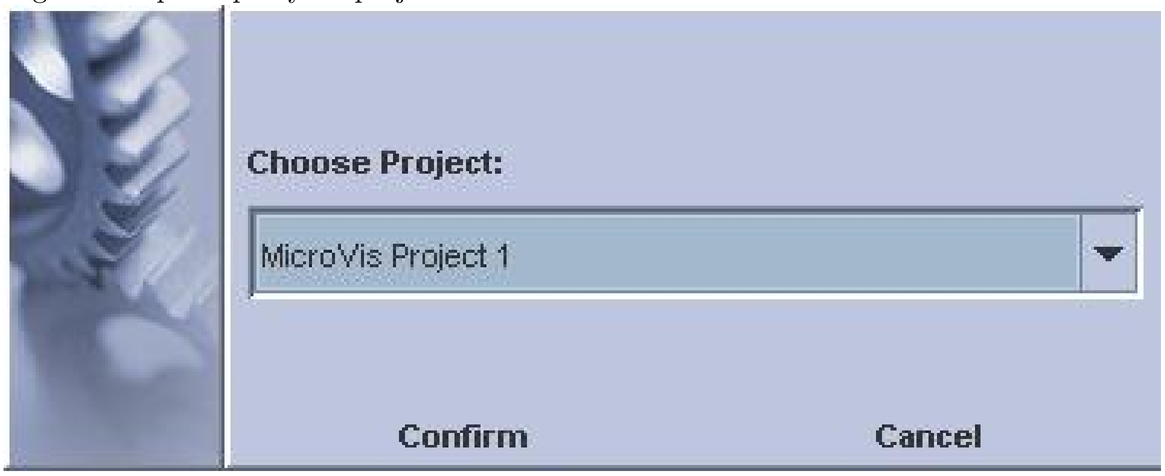
From Template: No Template (empty)

If there are no templates at disposition, the indication: "empty" displays.

The settings can be changed under **Project Properties** at any time.

Open Project:

A dialog shows up to open your projects



Choose Project: MicroVis Project 1

The projects are displayed under the project name you have chosen

Delete Project:

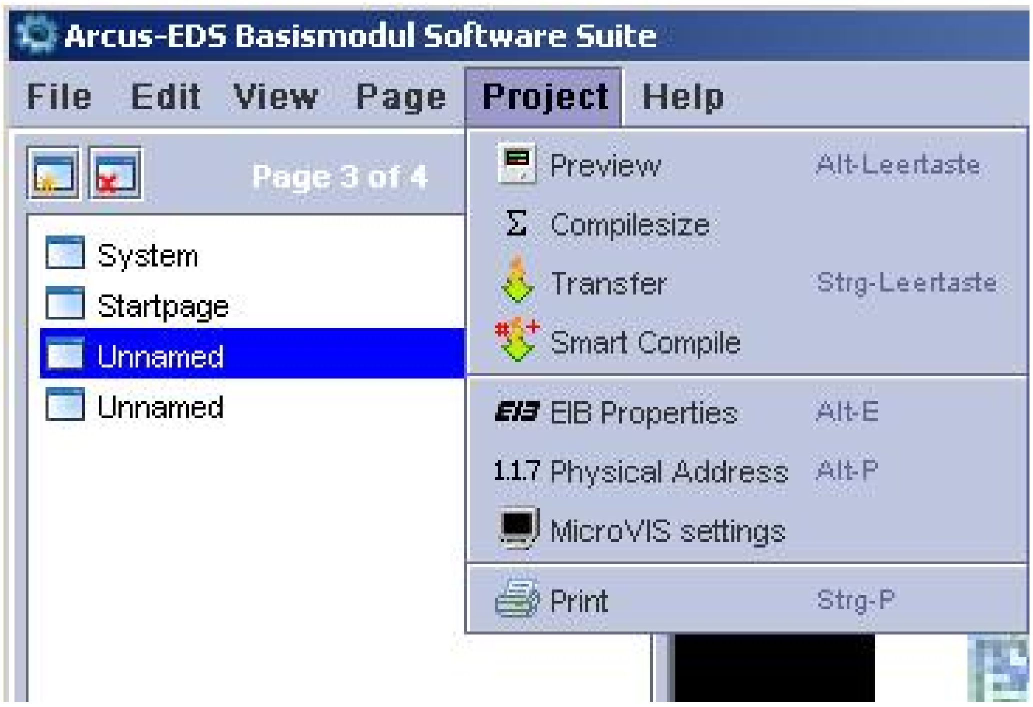
Displays a dialog to delete your projects

The projects are displayed under the project name you have chosen

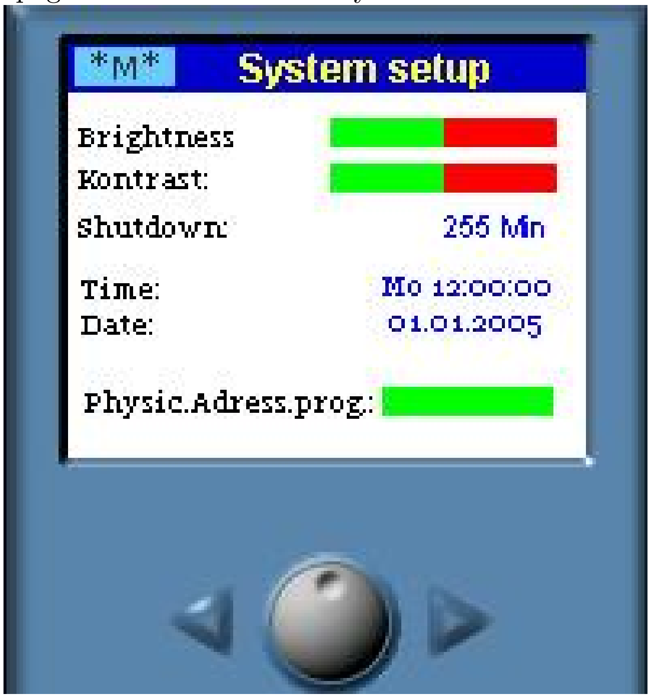
Saving the Project:
Saves the active project with all the changes you have made.
Save Project As:
Saves the current project with all the alteration you have made under a different name.
Project Properties:
Here you can display settings you have made carrying out your project and possibly change them.
Exporting the Project:
The current project can be packed in an export file under a chosen name. You can export it e.g. to another PC or save the current version for the purpose of documentation.
Importing the Project:
A previously exported project can be imported under a chosen name.

3.4 Current Project

Tools for your current project:



With the **Project Preview** you will get a realistic impression about the optical appearance of your project. The display scale corresponds approximately with the scale of the MicroVIS-display, so you can proof the sequence of pages about their useability.



M System setup

Brightness
Kontrast:
Shutdown: 255 Min
Time: Mo 12:00:00
Date: 01.01.2005
Physic.Adress.prog:

Using the arrow keys you can skip from one page to another.

Project Size:

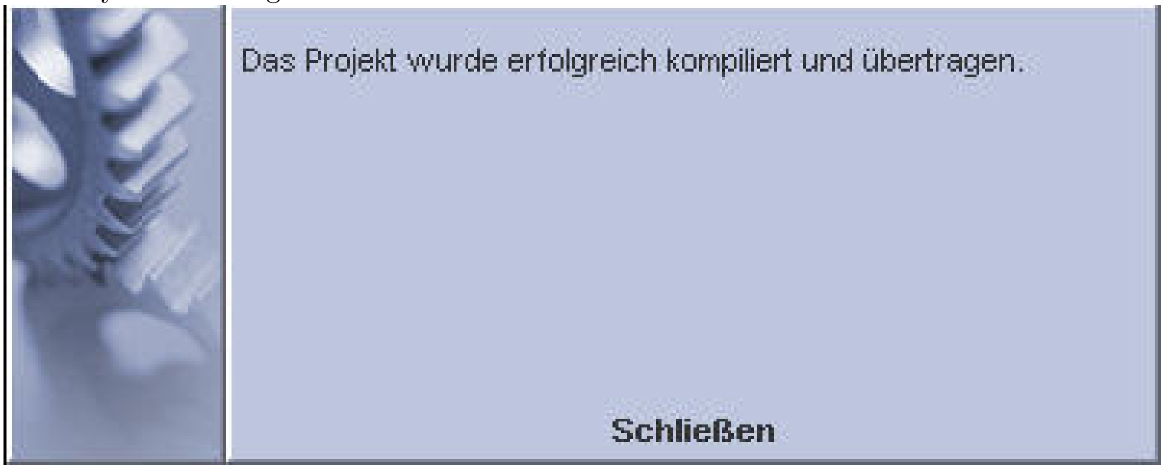
The current memory usage of your MicroVis Project is calculated and shown. The values are displayed in k- byte and in **Transmission:**

The current project is transfered to MicroVIS, which is connected with a USB cable.

An error occurring during the transmission can be mostly solved by a second transmission.

Sometimes you have to disconnect the device from the USB cable and reconnect it while down-pressing the button; the transmission should then be accurate in any case.

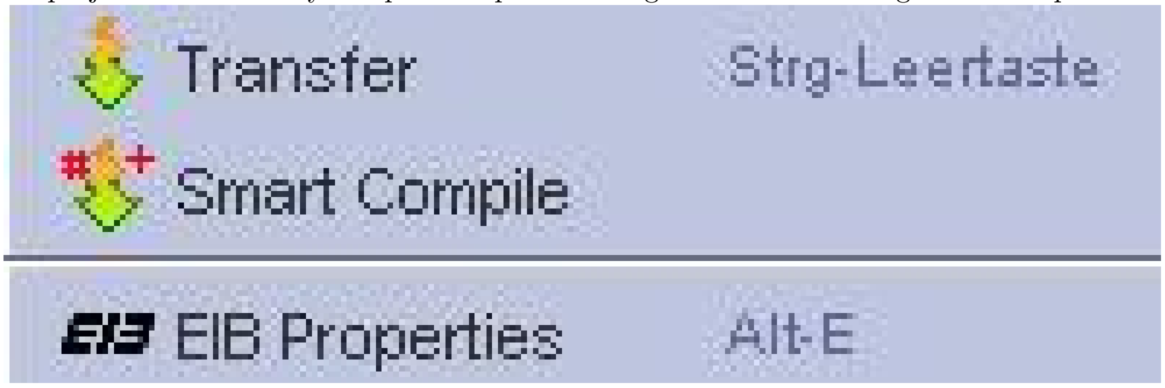
Please have in your mind, that the transmission executed with a connected EIB/KNX bus is mostly, but not always functioning!



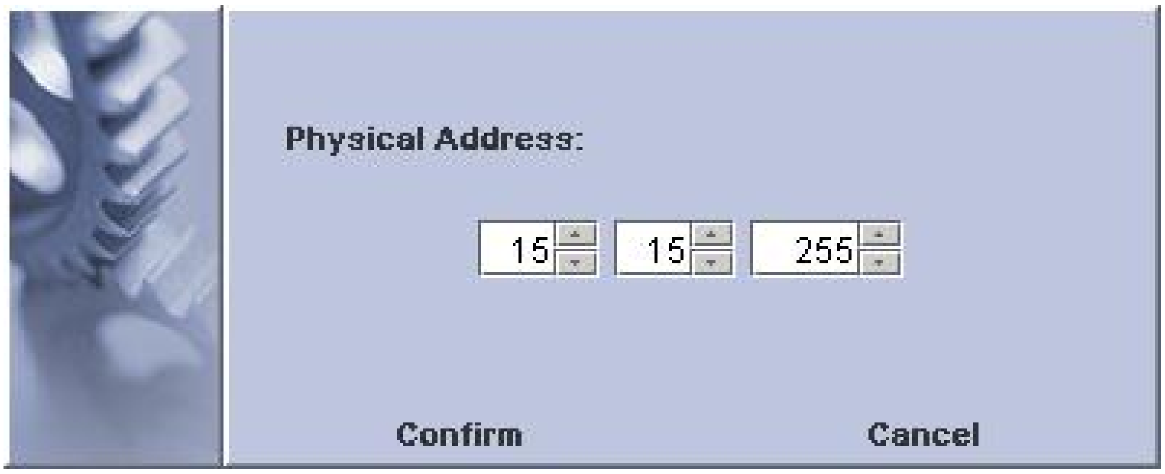
Das Projekt wurde erfolgreich kompiliert und übertragen.

Smart Compile:

In MicroVIS2 Projects the Menue-Item Smart Compiler is available. With this you can change and transfer projects controlled by compiler-scripts. The Pages as well as the logic-code is updated and.



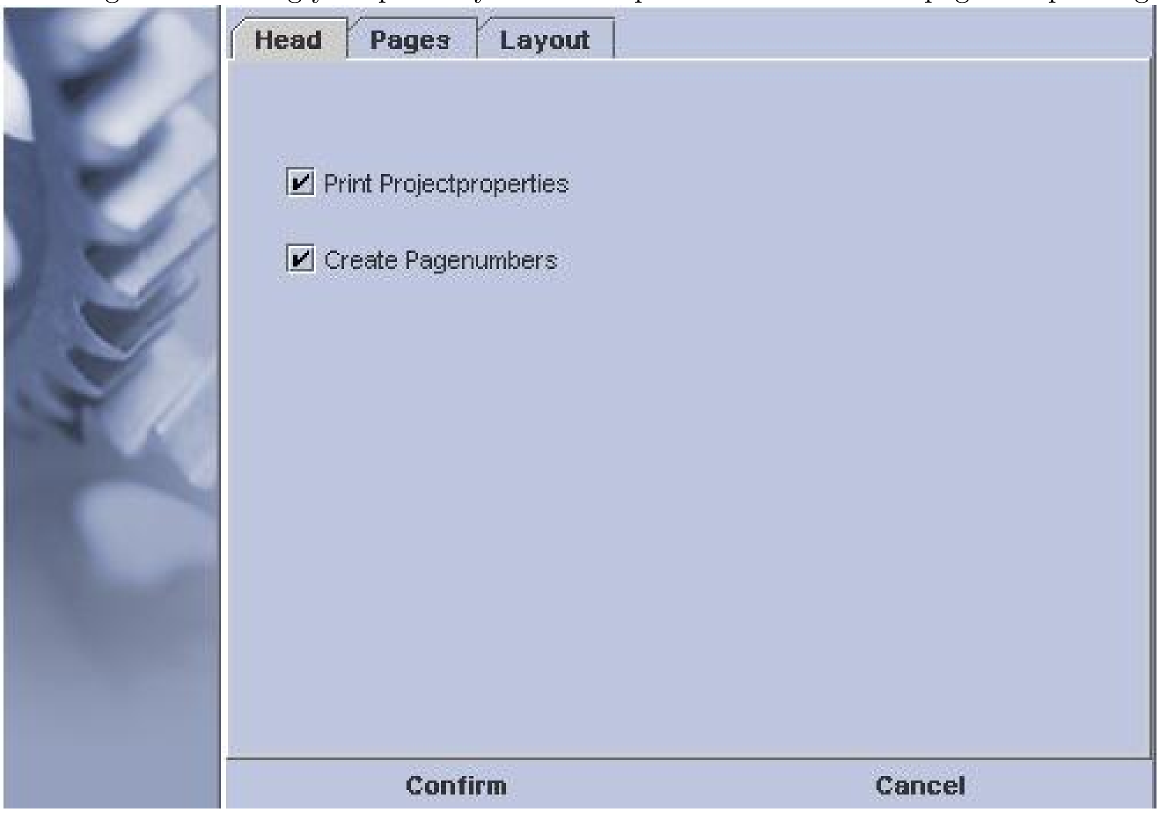
Physical Address:



The Physical Address of the MicroVis display is transmitted together with the project. It has to be a unique address within an EIB/KNX network.

Printing:

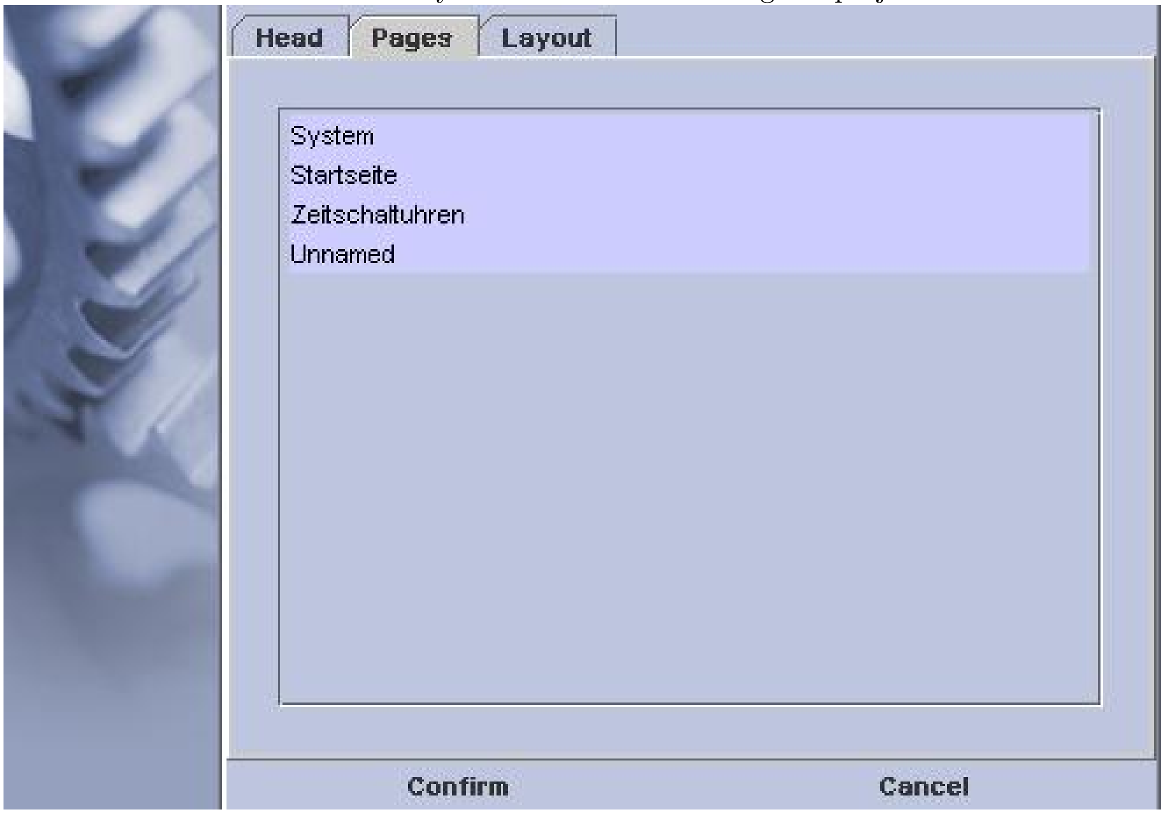
After selecting and installing your printer you can setup the information and pages for printing



[✓] Print Projectproperties

[✓] Create Pagenumbers

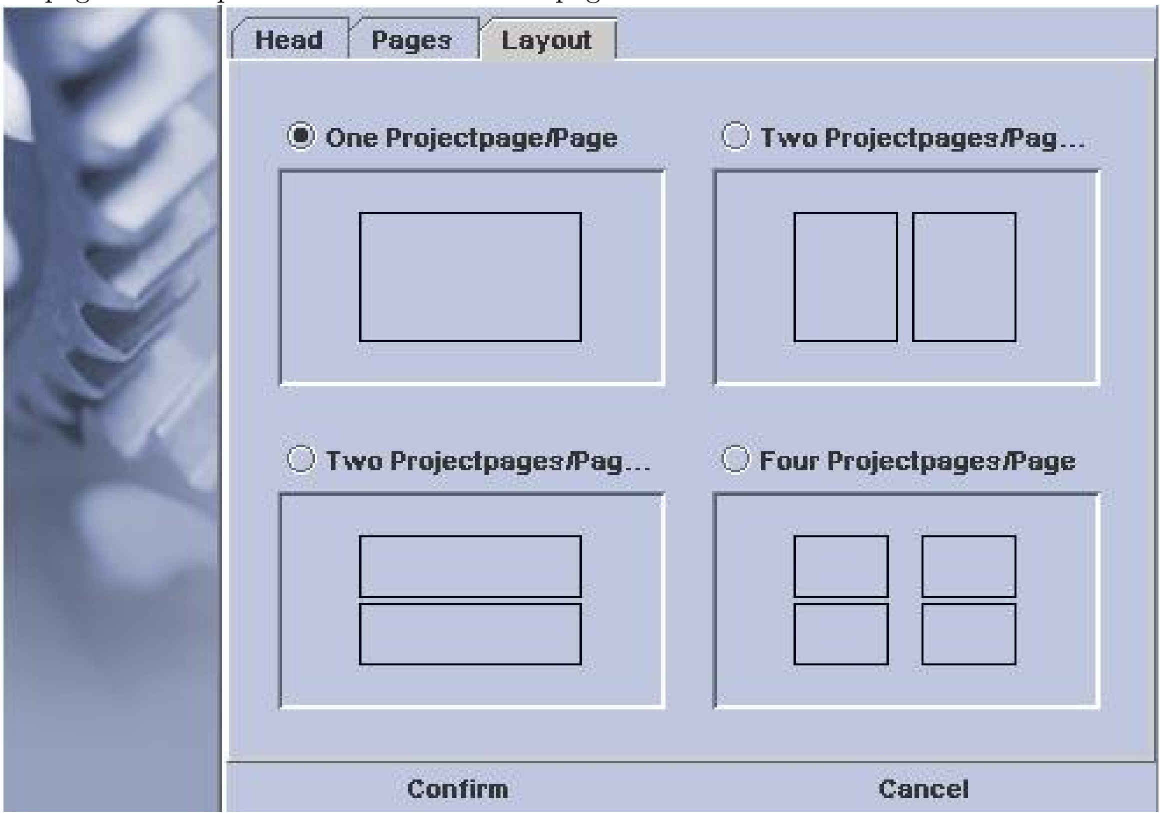
Project information is the information you have declared creating the project.



System
Startseite
Zeitschaltuhren
Unnamed

All pages are marked by default and therefore printed.
Pages, which should not be printed, can be skipped.

Up to 4 pages can be printed on one document- page.

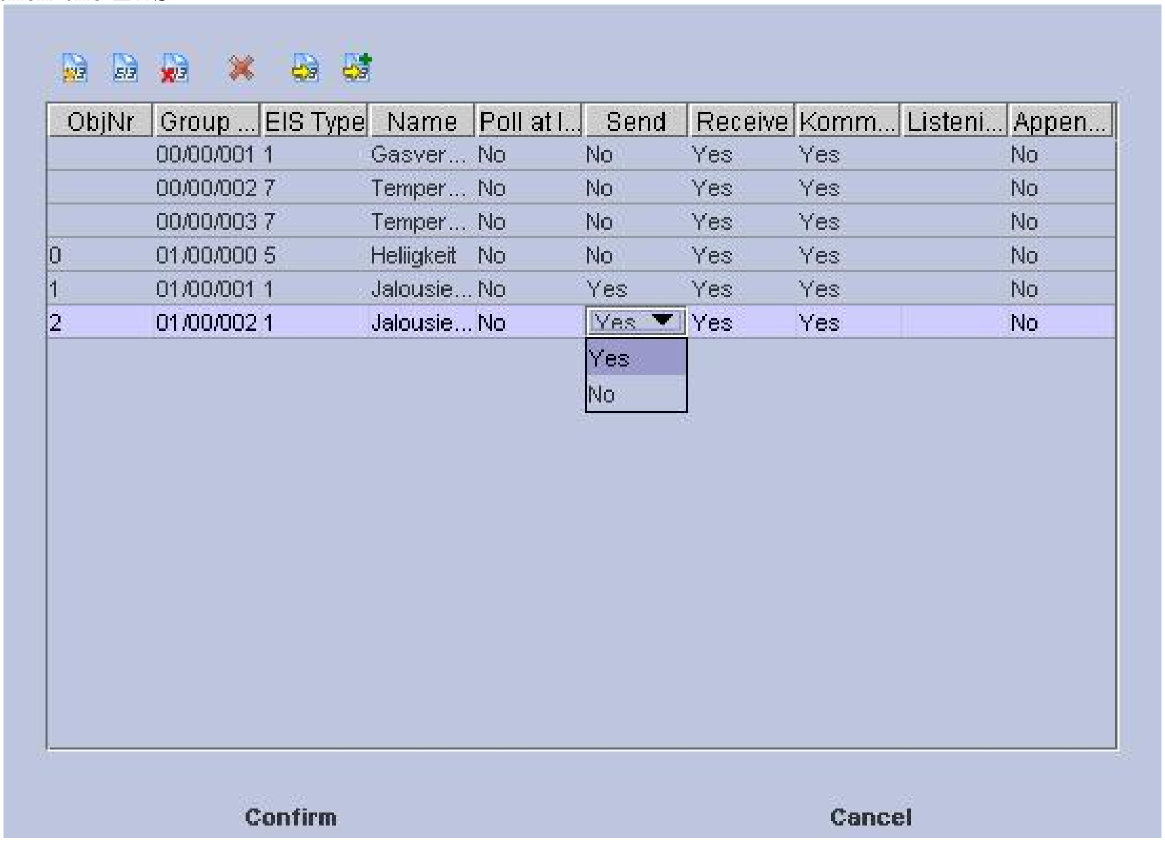


[✓ One Projectpage/Page]
Two Projectpages/Pag...
Two Projectpages/Pag...
Four Projectpages/Page

3.5 EIB settings

At this point you need special knowledge about the European Installation Bus (EIB). You can ask your system integrator or your EIB service provider.
Any number of objects can be defined but only Up to 44 (128 on MicroVIS2) EIB group-addresses can be used (including listening GA‘s) . The import of group addresses is carried out by direct read out of the ETS3 database. Address- points can be as well entered and processed manually one by one. Attention! If you want to work parallelly with Micro VIS and ETS, start the Micro VIS Module first

and then the ETS.

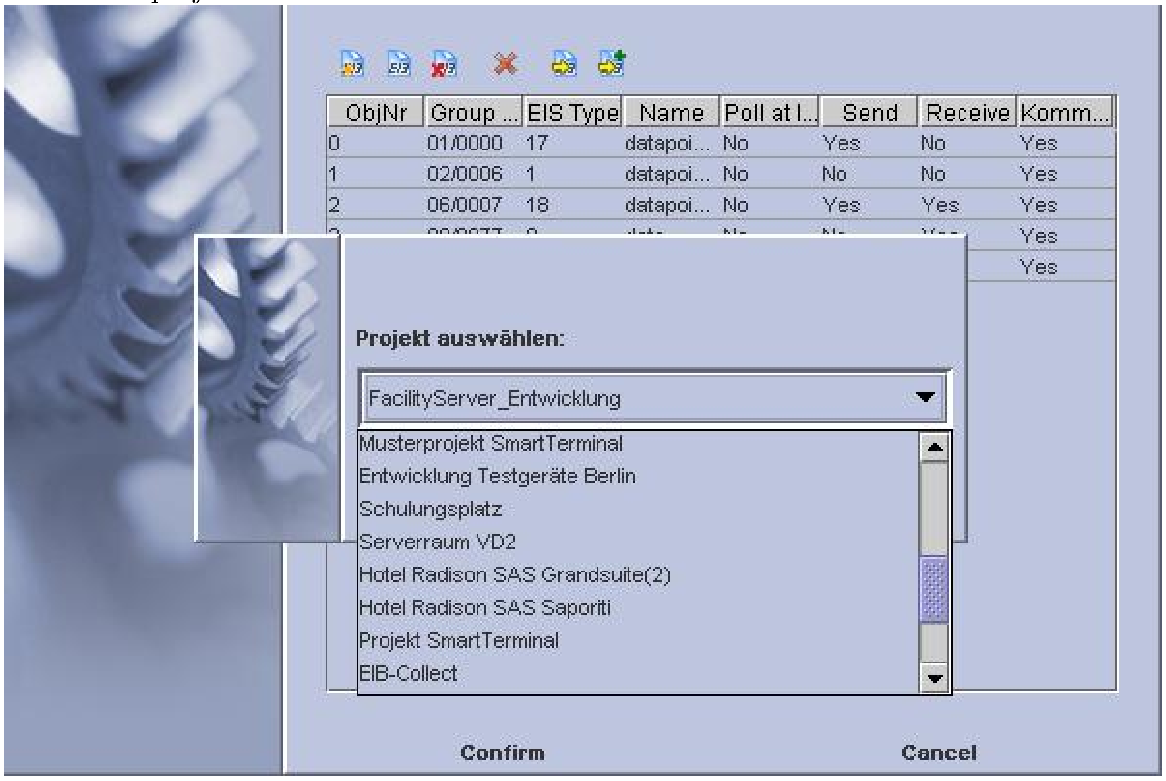


ObjNr	Group ...	EIS Type	Name	Poll at I...	Send	Receive	Komm...	Listeni...	Appen...
	00/00/001	1	Gasver...	No	No	Yes	Yes		No
	00/00/002	7	Temper...	No	No	Yes	Yes		No
	00/00/003	7	Temper...	No	No	Yes	Yes		No
0	01/00/000	5	Heliigkeit	No	No	Yes	Yes		No
1	01/00/001	1	Jalousie...	No	Yes	Yes	Yes		No
2	01/00/002	1	Jalousie...	No	Yes	Yes	Yes		No

Import of EIB- objects out of ETS:

A new import overwrites the whole datapoint list. Carrying out a first import select your appropriate ETS database. This entry is stored under settings and can be changed any time.

Select the ETS- project:



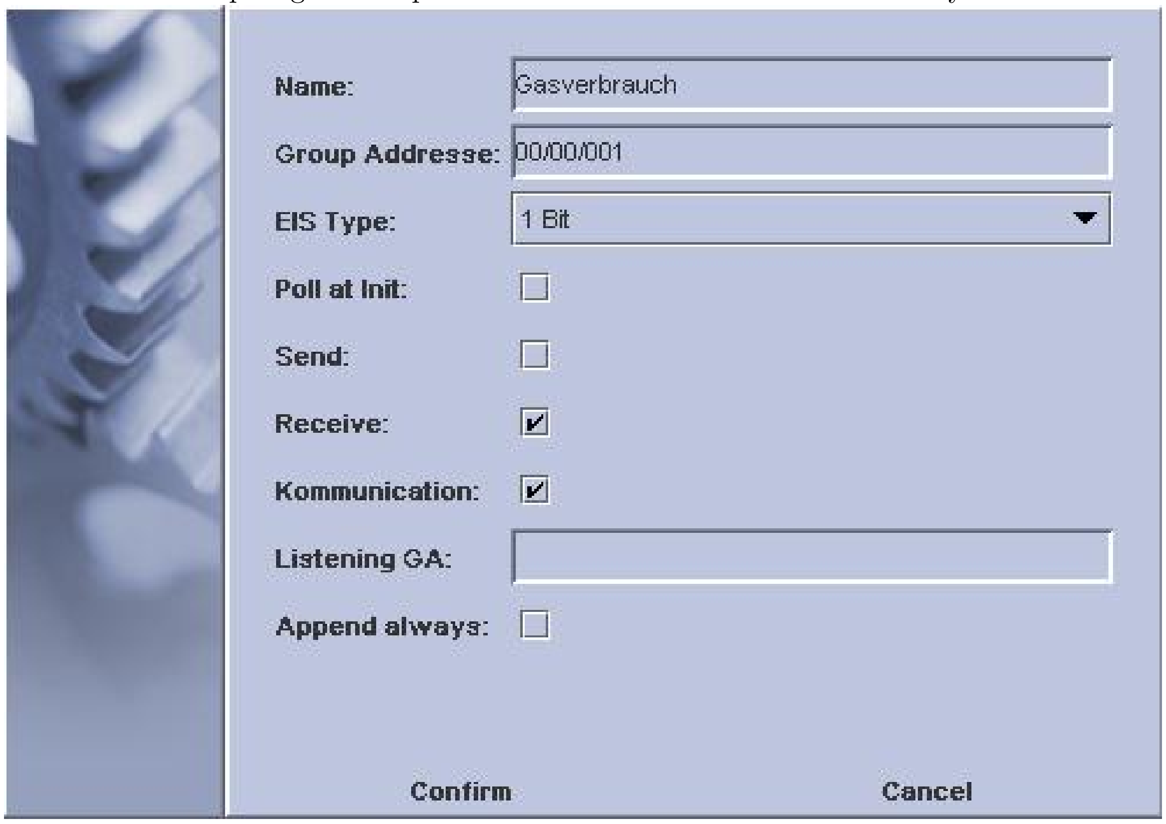
ObjNr	Group ...	EIS Type	Name	Poll at I...	Send	Receive	Komm...
0	01/0000	17	datapoi...	No	Yes	No	Yes
1	02/0006	1	datapoi...	No	No	No	Yes
2	06/0007	18	datapoi...	No	Yes	Yes	Yes
							Yes
							Yes

Projekt auswählen:

FacilityServer_Entwicklung

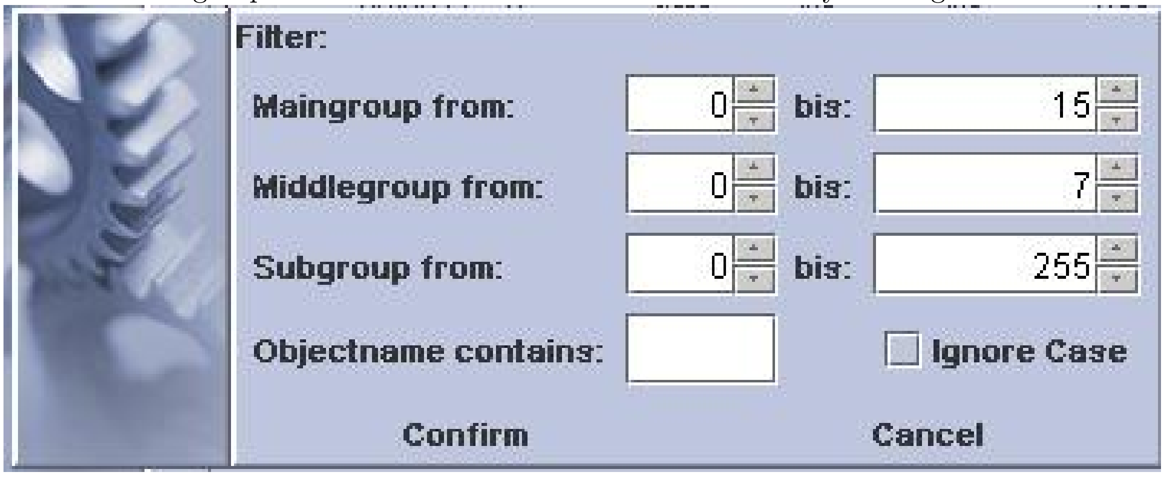
- Musterprojekt SmartTerminal
- Entwicklung Testgeräte Berlin
- Schulungsplatz
- Serverraum VD2
- Hotel Radison SAS Grandsuite(2)
- Hotel Radison SAS Saporiti
- Projekt SmartTerminal
- EIB-Collect

You can confirm or skip single data- points or take over all of them immediately.



Name:	Gasverbrauch
Group Addresse:	00/00/001
EIS Type:	1 Bit
Poll at Init:	☐
Send:	☐
Receive:	✓
Kommunication:	✓
Listening GA:	
Append always:	☐

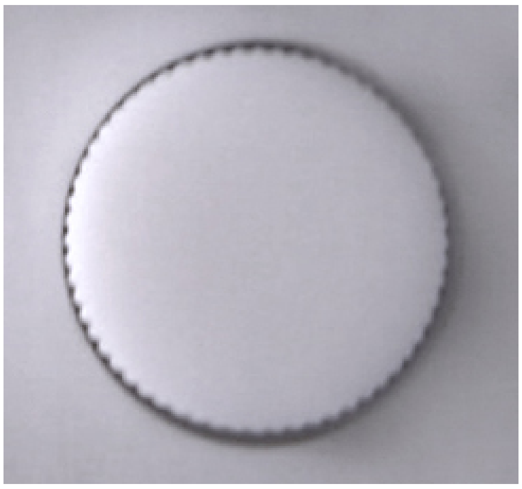
Polling at start:
By starting the system, the data- point is called up. Therefore the communication- and receiving- flag have to be set up.
To Send:
The elements connected with this object can send data to the bus. The communication- flag has to be set up.
To Receive:
The elements connected with this object can receive data from the bus. The communication- flag has to be set up.
Communication:
For sending or receiving, the communication has to be allowed.
Listening Adresses:
You can define a list of listening GA‘s for this object. Format is
1/2/3 1/2/4 .... or 1/678 2/679 ....seperated by spaces.
Append Always:
Since the imported EIB-Objects are only appendet to the internal tables and transfered to the Micro-VIS if they are used inside the project, you have to declare those objects, that are only present to the logic-functions in the MicroVIS, to be always appendet to the object-tables.
By double-clicking on the entry these settings can be displayed in the overview and changed.
Add EIB- objects filtered from ETS:
With the button: "add EIB- objects filtered from ETS" you can carry out the selective data- import through the primary- ,secondary- and sub- groups. can Try it as well by the full- text- search: "contains object- name". The group- addresses will then be added to the already existing list.


Filter:

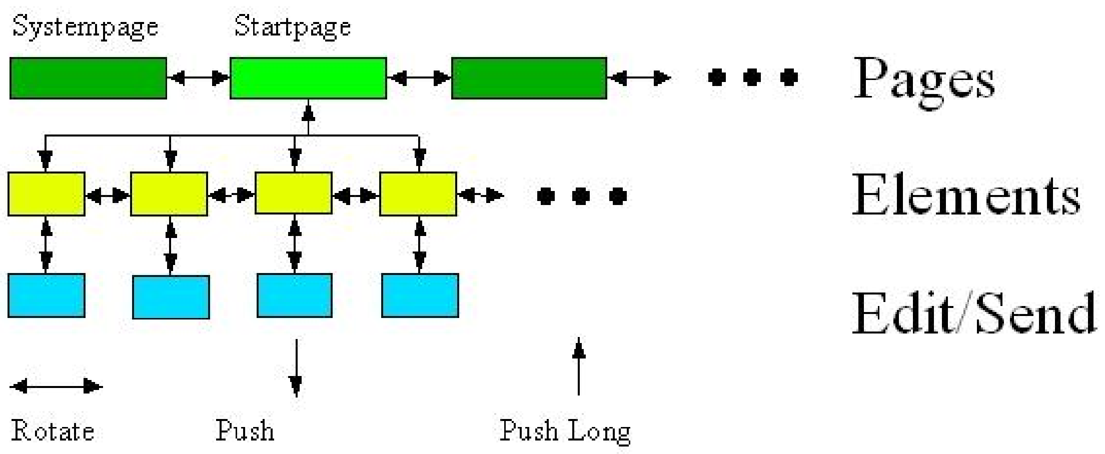
Maingroup from:	0	bis:	15
Middlegroup from:	0	bis:	7
Subgroup from:	0	bis:	255
Objectname contains:			☐ Ignore Case

3.6 Page Architecture and Navigation

3.6.1 Navigation

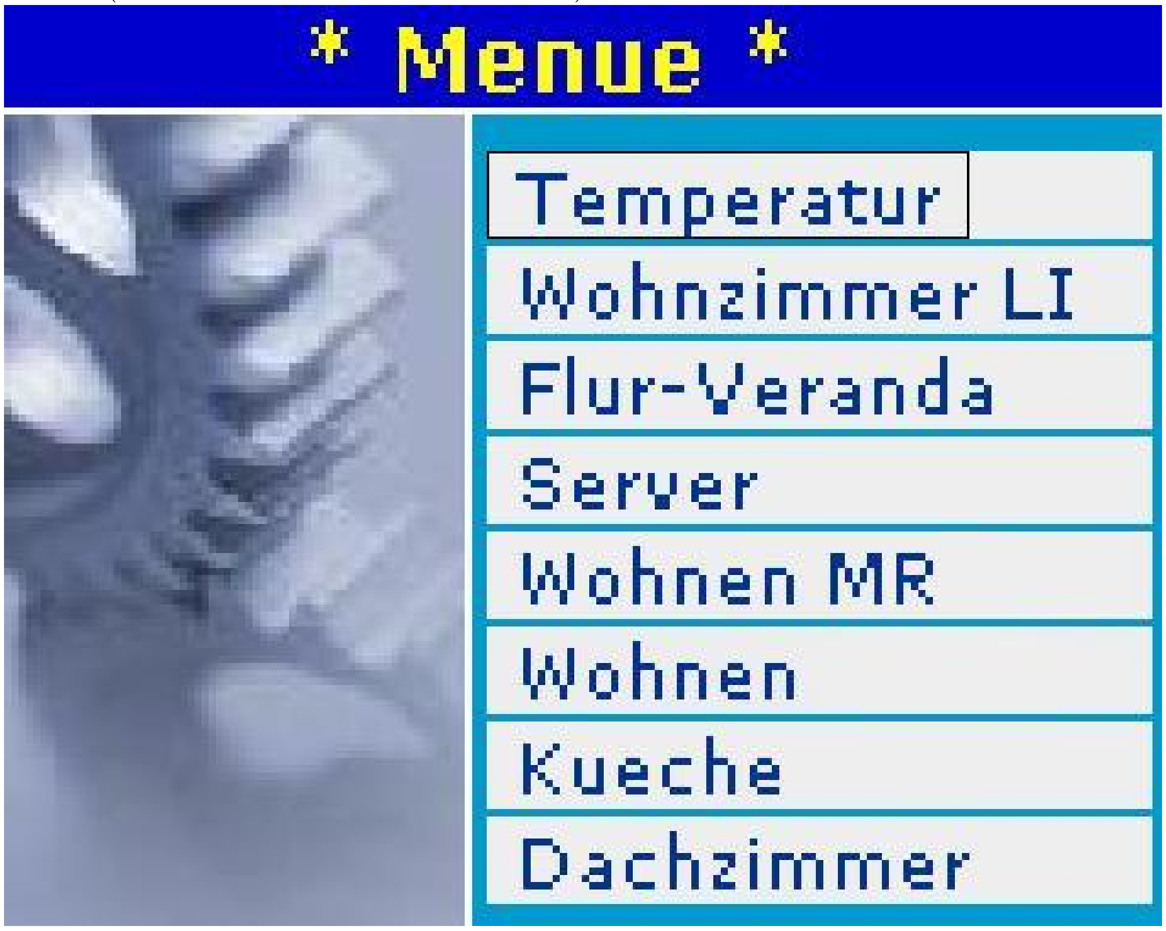


The navigation is done by turning and pressing the operating button.
The elements and pages are organised in a hierarchy:



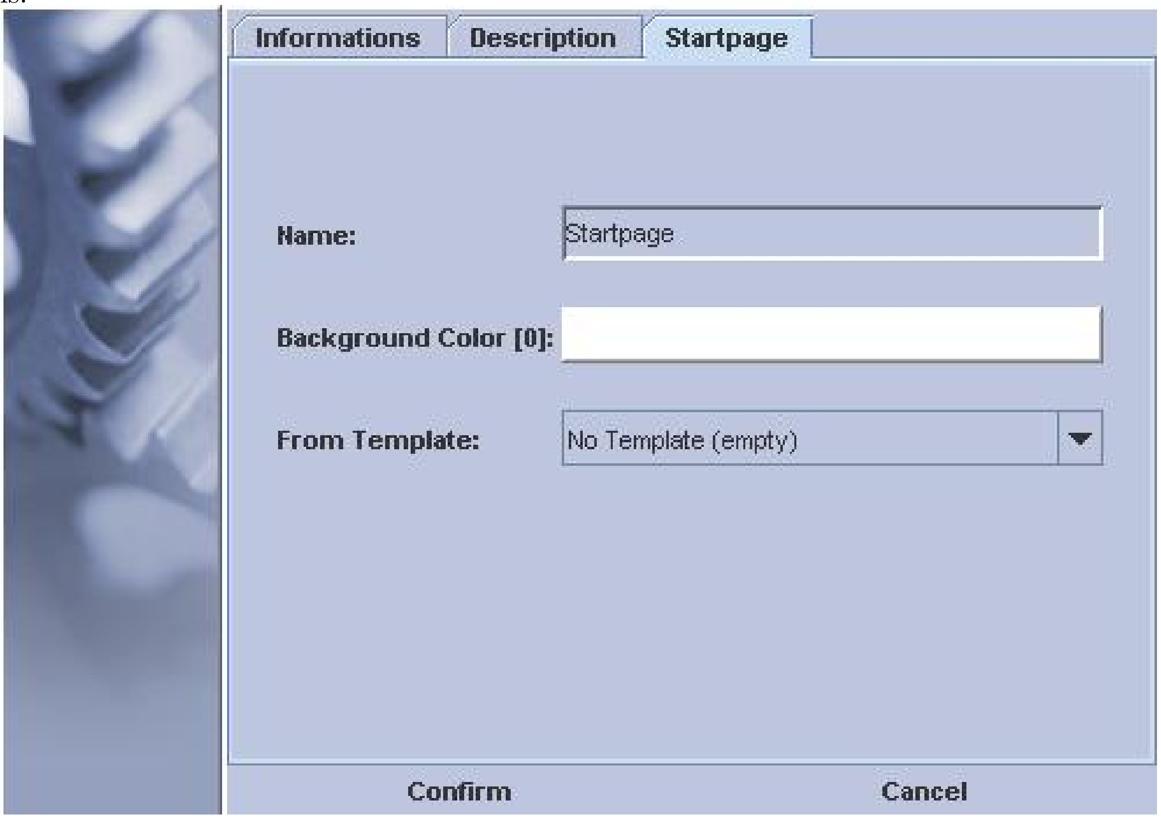
When the system is started the Startpage will be displayed. By turning the button the page displayed can be selected.
By pressing the button the page is opened and the first selectable element is marked. By turning the button the selectable elements are marked in their order selectable within the software. Operating the button once more selects the element. According to the kind of its function a value can be installed or selected as well as a telegram sent to the bus.
By pressing longer than 2 seconds the level is abandoned and the user reaches one level up.
From the Firmware- version 1.5 on a change of pages can also be initiated by operating a special

Textelement (Static Textfield with Focus allowed).



So complex menu navigation can be realised as well.
Since Firmwareversion 2.2 (MicroVIS2) pages can be declared as hidden pages, which is done in the pageproperty-dialog. A hidden Page is not reachable with "normal" operation, only with jumps and

alarms.



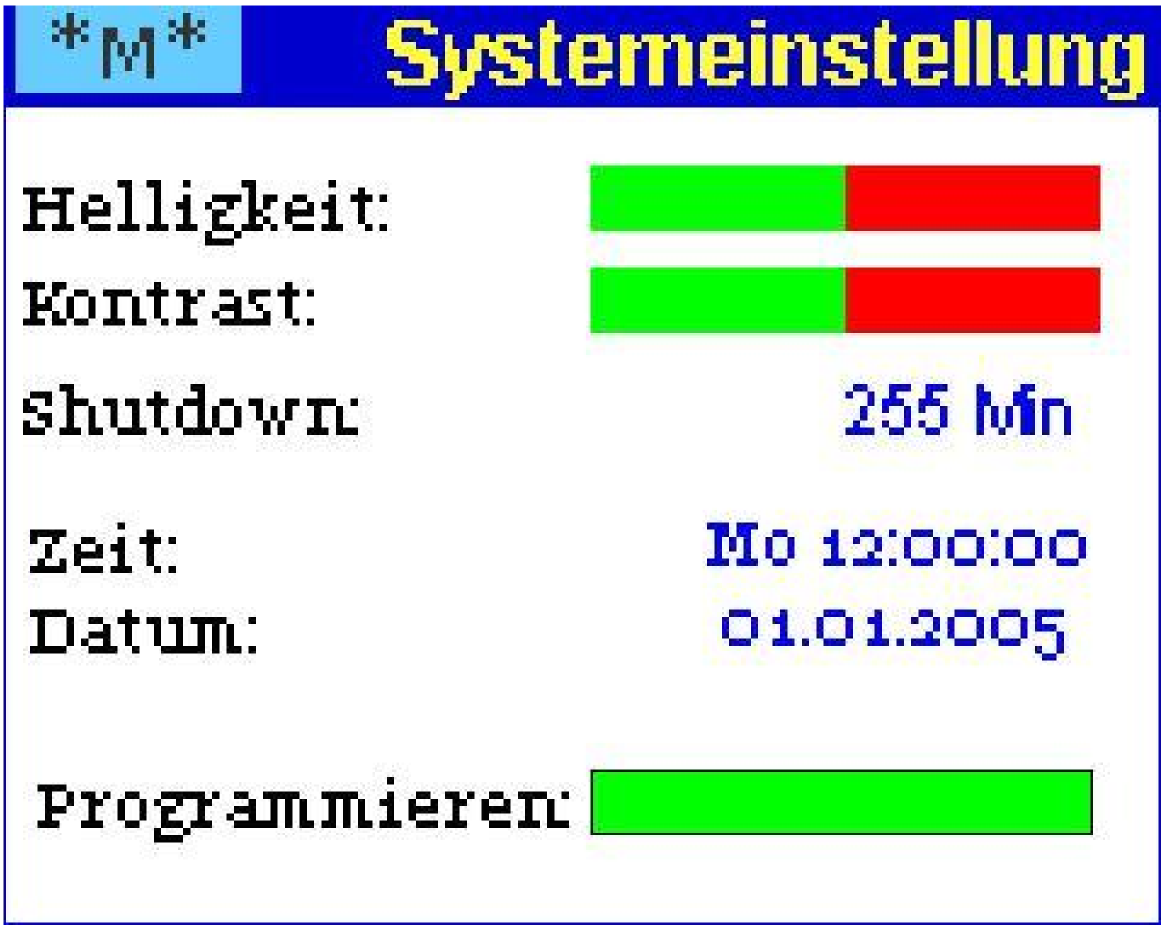
Name: Startpage

Background Color [0]:

From Template: No Template (empty)

Jumping back from a hidden page is done to the page the jump was performed from.

3.6.2 Systempage



M Systemeinstellung

Helligkeit:

Kontrast:

Shutdown: 255 Min

Zeit: Mo 12:00:00

Datum: 01.01.2005

Programmieren:

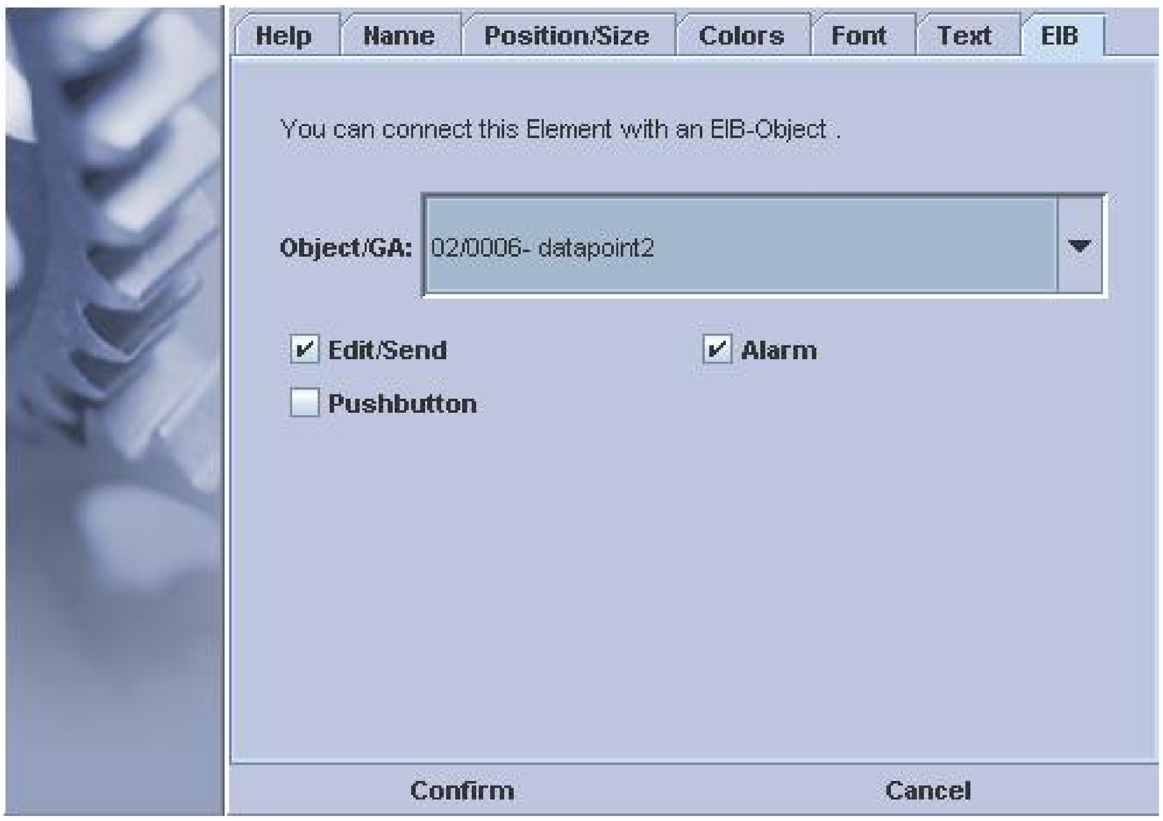
The page for system setting normally contains the system parameters Contrast, brightness, shutdown timeout of the display, setting time and date date and Activation of the ETS programming capability.

- Contrast and brightness with a bargraph from 0 to 100
- The shutdown- time can be programmed from 0 to 255 minutes
- Time and date can be set up manually, if the system time is not taken over the EIB/KNX- Bus
- With the programming key the display can be switched to the programming mode to program the physical address over the EIB/KNX- Bus. The physical address can also be changed in the MicroVis program.

By downloading the project, the address programmed over ETS is overwritten. When the program mode is activated, it is not possible to operate the device until the program mode is finished.
The system setting page can be freely shaped in the program.

Functions also can be deleted, if the user should not have access to the settings (especially the programming mode).

3.6.3 Alarmfunctions



You can connect this Element with an EIB-Object .

Object/GA: 02/0006- datapoint2

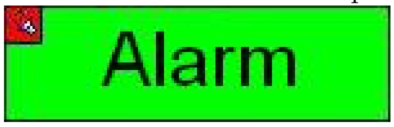
☑ Edit/Send
☑ Alarm
☐ Pushbutton

An element on a MicroVis page can be marked as an alarm function. At reception of new data for this element the responding page is displayed automatically. The backlight is lit automatically (Since Firmware-Version 2.2).

Acoustic Alarm function:

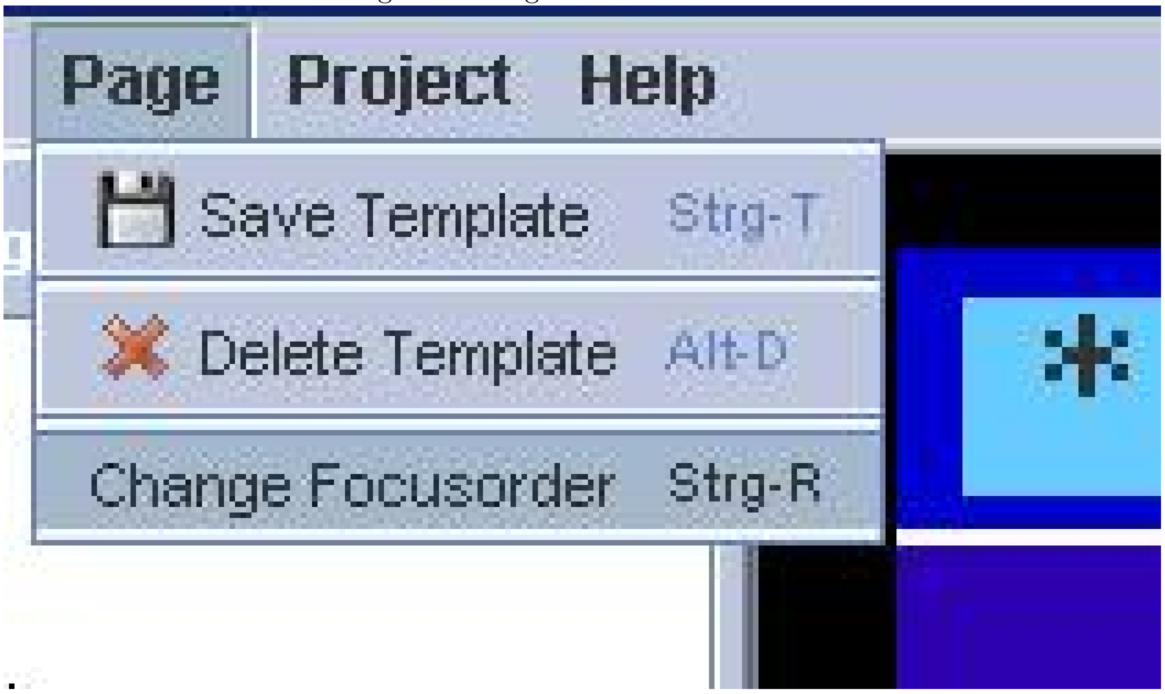
An acoustic alarm- function can be generated additionally for elements, which have a text output (buttons, text lists). A signal tone can be switched on or off with one text (e.g. a button caption), beginning with "#1" and another text with "#0". An alarm is generated, which goes off, when the alarm condition has ended or can continue, until the device will be operated with the button. The front characters ("#1"/"#0") are not displayed.

In the page view a buzzer symbol will be shown within the respective object.

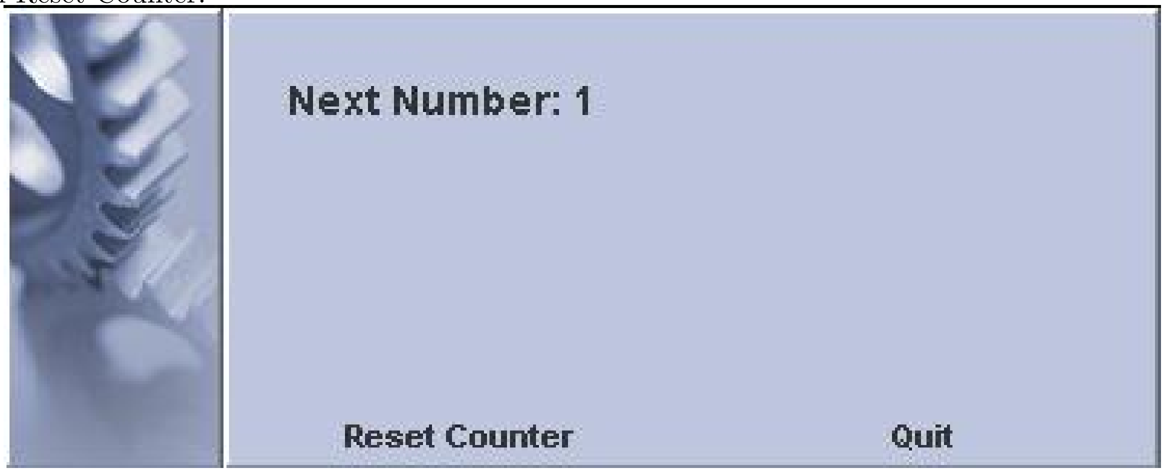


3.6.4 Focus Order

Focus Order :
The order of the focus and therefore the order, in which the elements will be called up at the page, can be defined any time anew. Fixing the Focus Order:
Open the function in the menu Page -> Change Focusorder

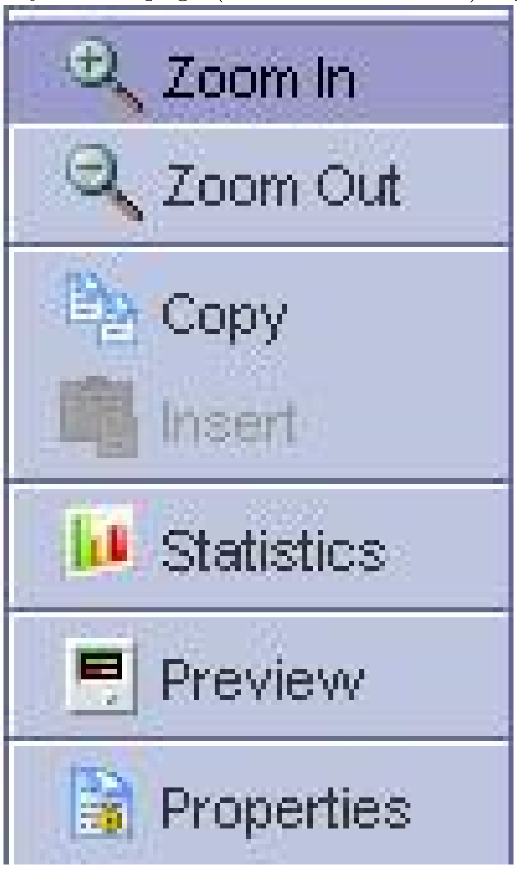


Then Reset Counter.

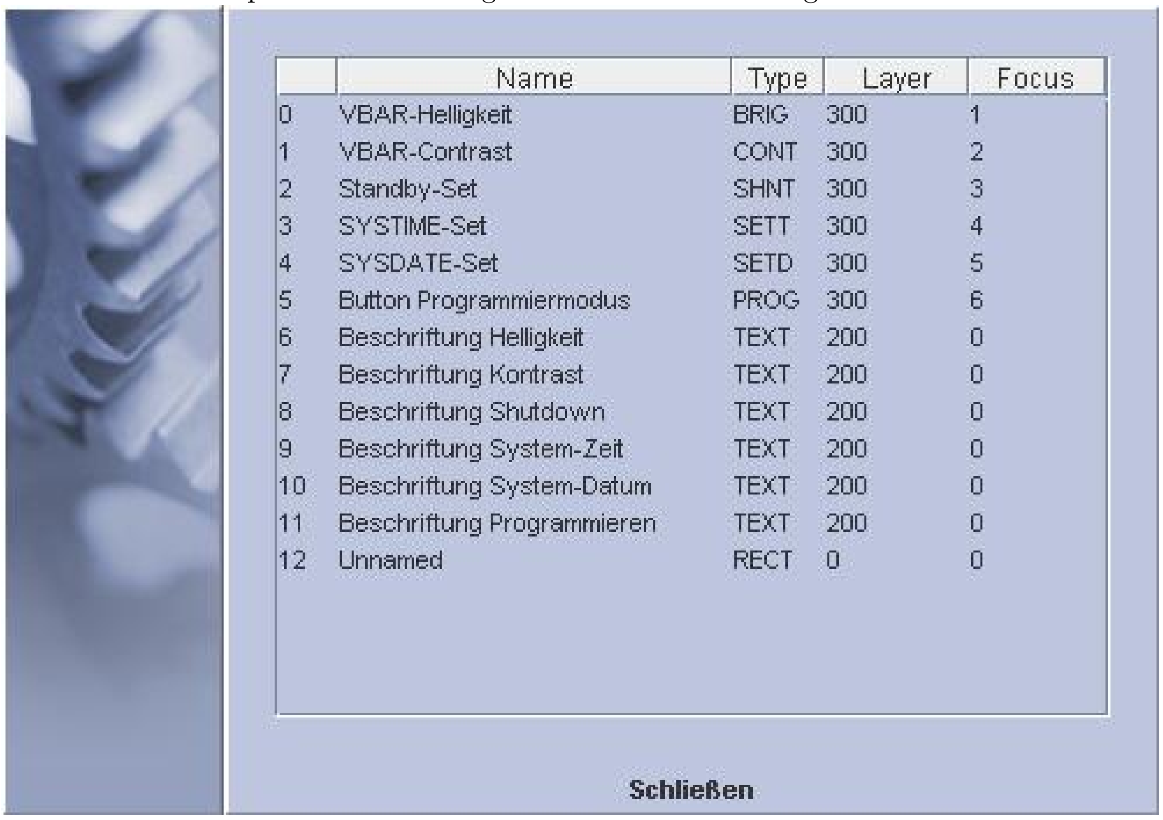


A new order is now established by mouse-clicks at all editable fields one by one.
Order Control:

Clicking with the right mouse-key on the page (not at an element) a pull-down- menu is opened.



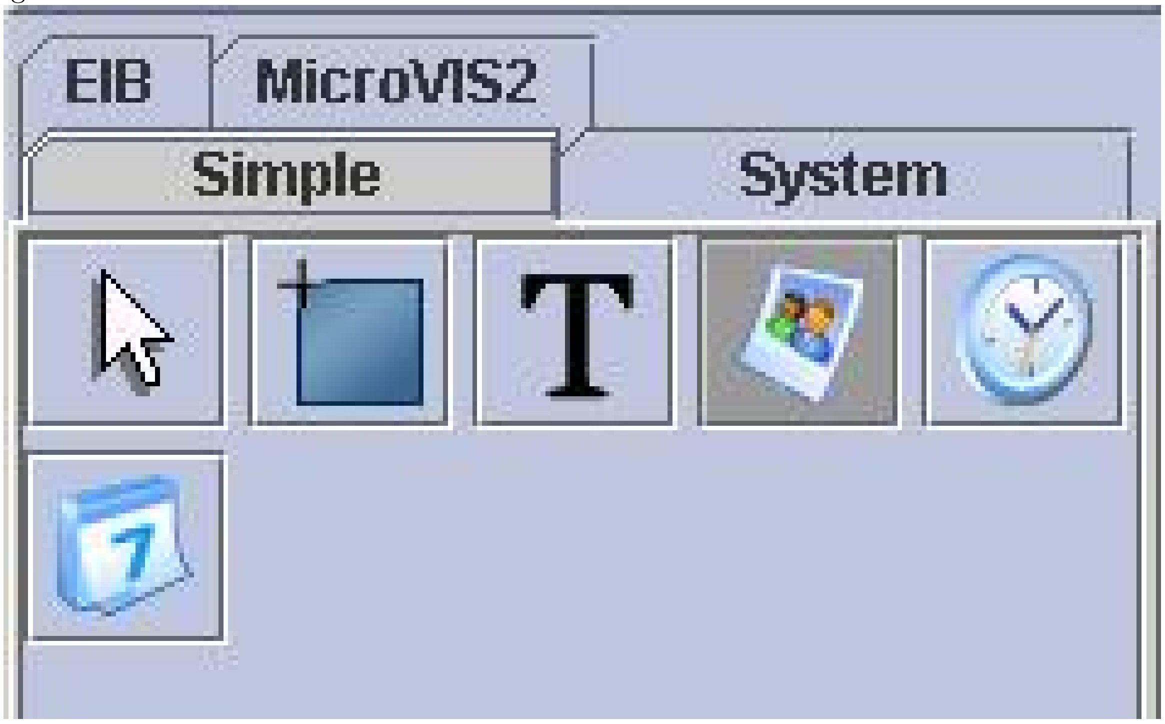
The focus order shows up and can be changed at the statistics dialog.



	Name	Type	Layer	Focus
0	VBAR-Helligkeit	BRIG	300	1
1	VBAR-Contrast	CONT	300	2
2	Standby-Set	SHNT	300	3
3	SYSTIME-Set	SETT	300	4
4	SYSDATE-Set	SETD	300	5
5	Button Programmiermodus	PROG	300	6
6	Beschriftung Helligkeit	TEXT	200	0
7	Beschriftung Kontrast	TEXT	200	0
8	Beschriftung Shutdown	TEXT	200	0
9	Beschriftung System-Zeit	TEXT	200	0
10	Beschriftung System-Datum	TEXT	200	0
11	Beschriftung Programmieren	TEXT	200	0
12	Unnamed	RECT	0	0

3.7 Elements for Page Layout

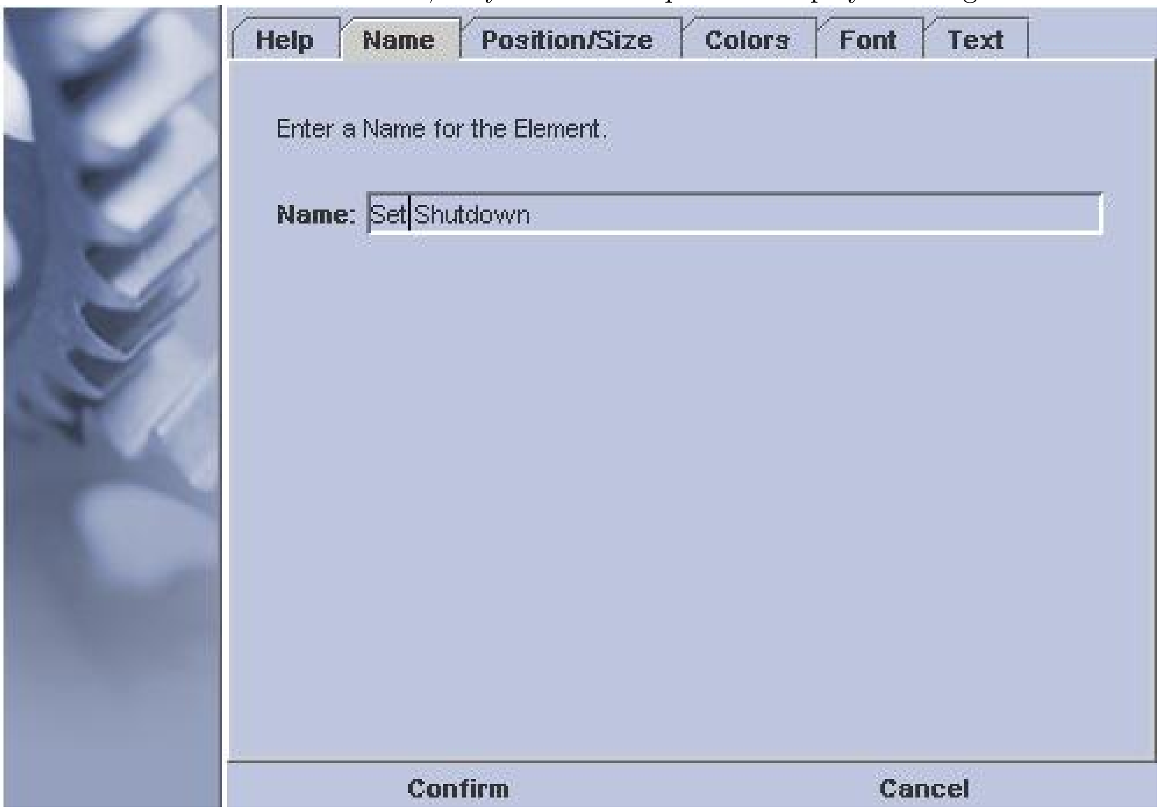
Each page is composed of elements. Simple elements, System elements and EIB/KNX elements are distinguished.



Elements are inserted by selecting the corresponding element in the option of elements and positioned with the mouse at the desirable place. Automatically a dialog pops up about the details of the respective element. Please notice, that a valid EIB Dataobject has always to be assigned to the EIB/KNX elements, otherwise the program refuses to save the element.

Properties of the Elements:

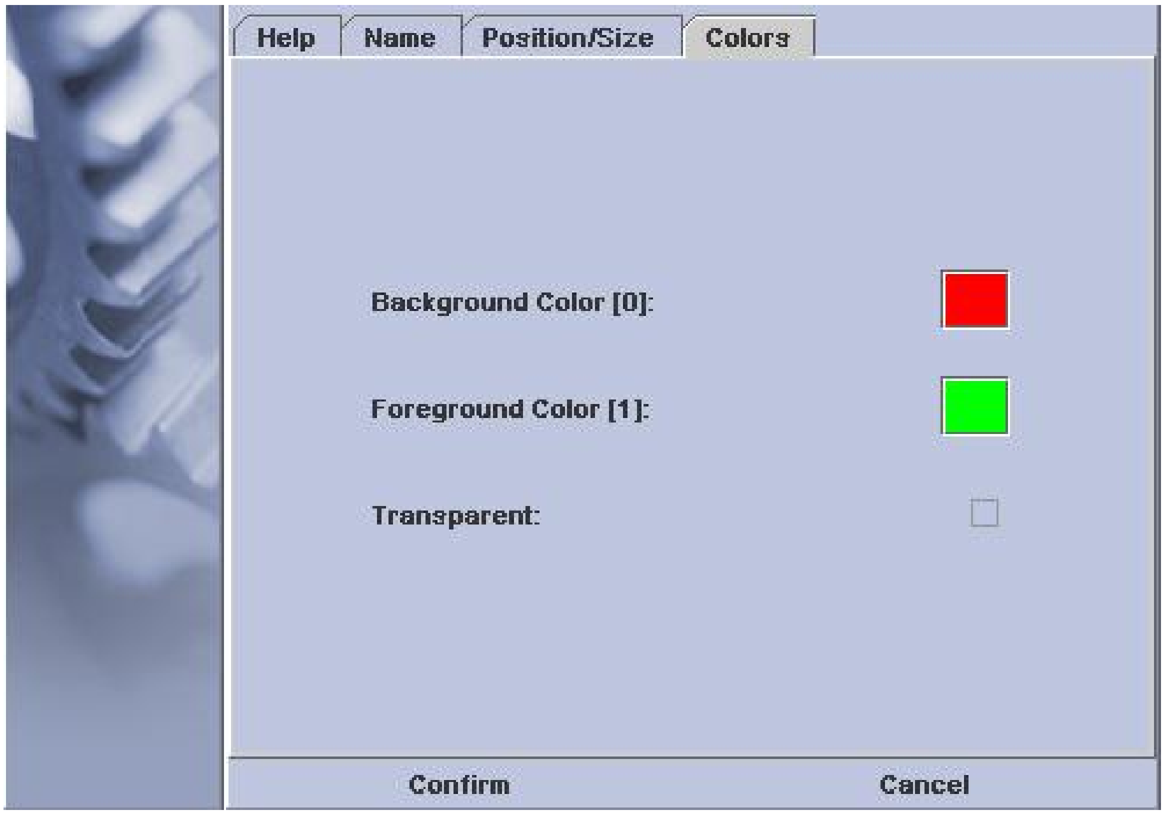
Some elements have special properties, which are listed in detail below the elements. Common used properties are shown as detailed below; they can be set up in the displayed dialogue.



Enter a Name for the Element.

Name: Set Shutdown

The name is used for a differentiation of the different elemets, but is not required for the transmitted project.

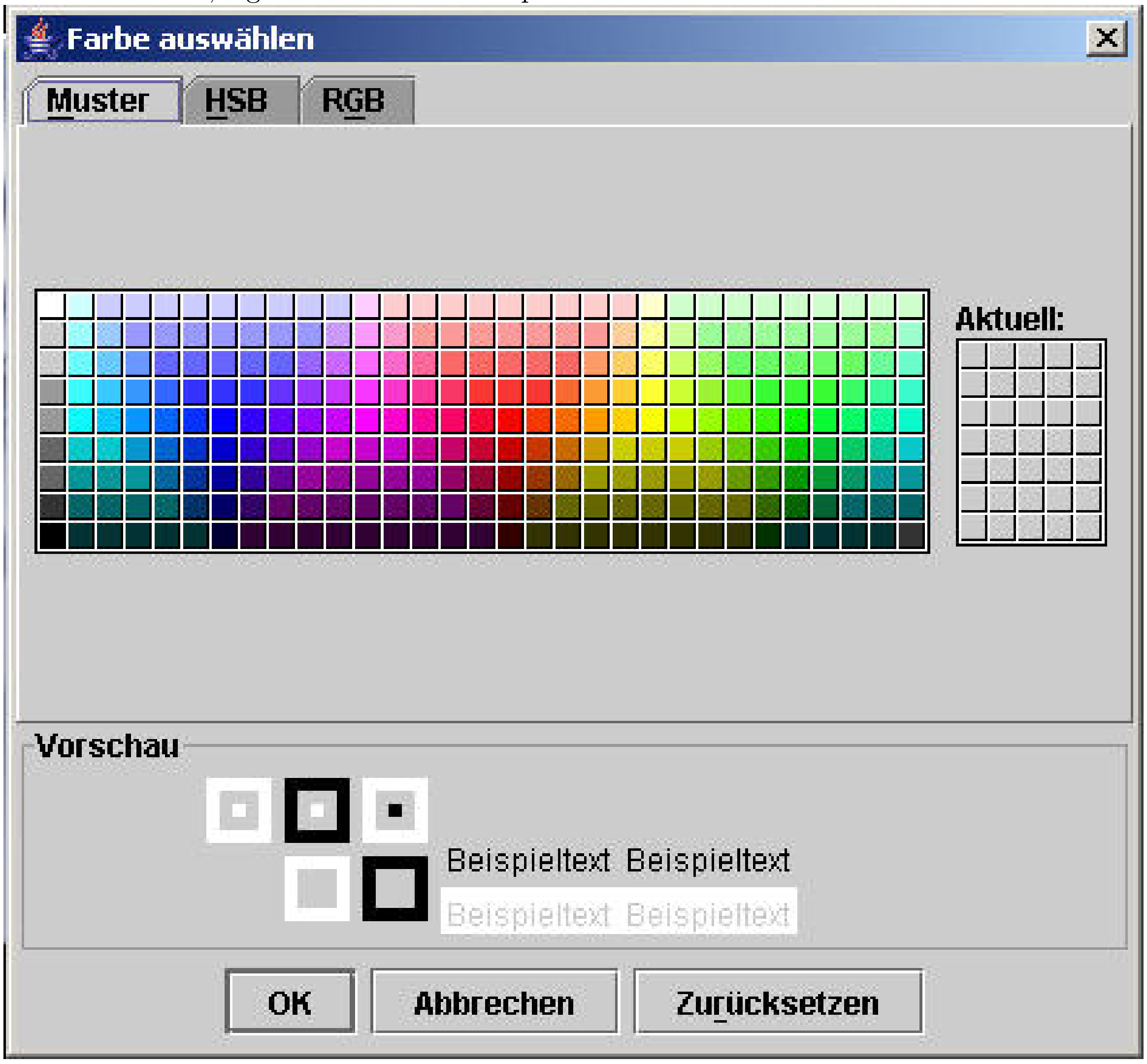


Background Color [0]: 🟥

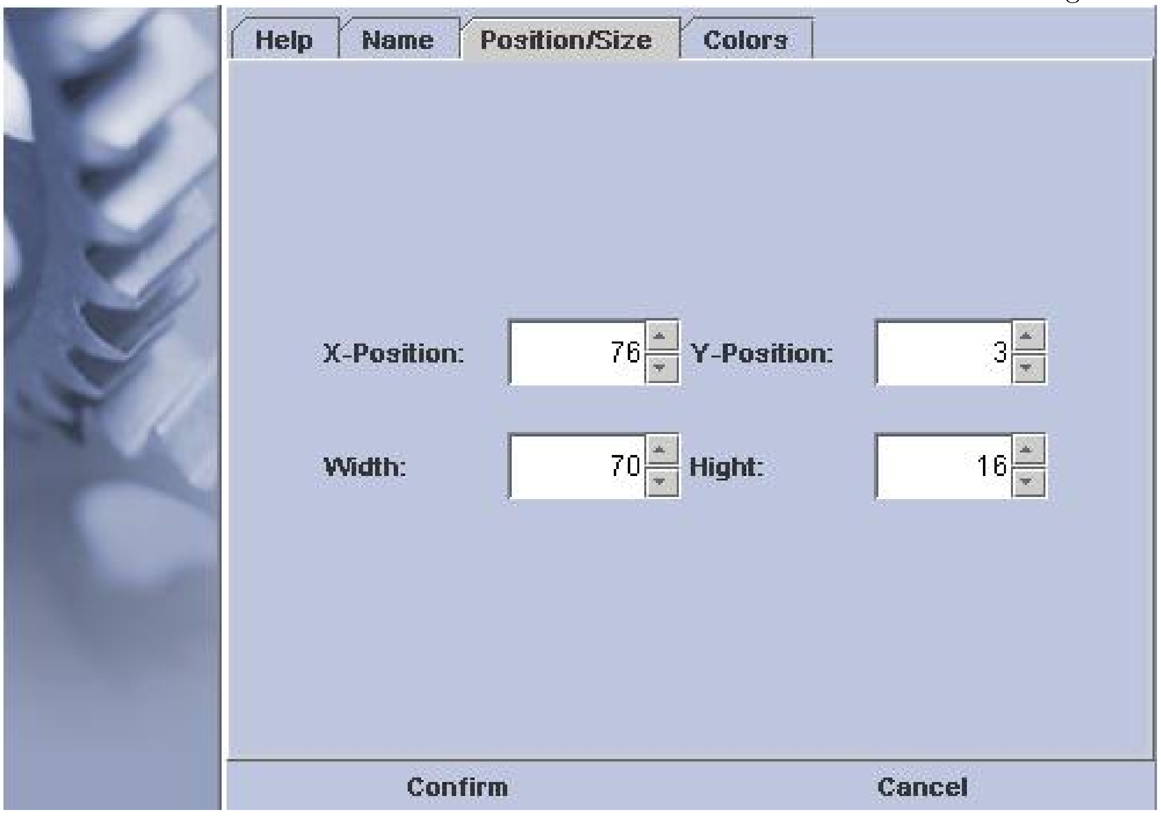
Foreground Color [1]: 🟩

Transparent: ☐

The colours for foreground and background can be determined through the colour dialogue. At the button element the background colour shows the value 0 and the foreground colour the value 1. Colours are determined by a colour option dialogue. Some elements can be drawn transparently, only the foreground will be drawn, e.g. for solid texts over pictures.



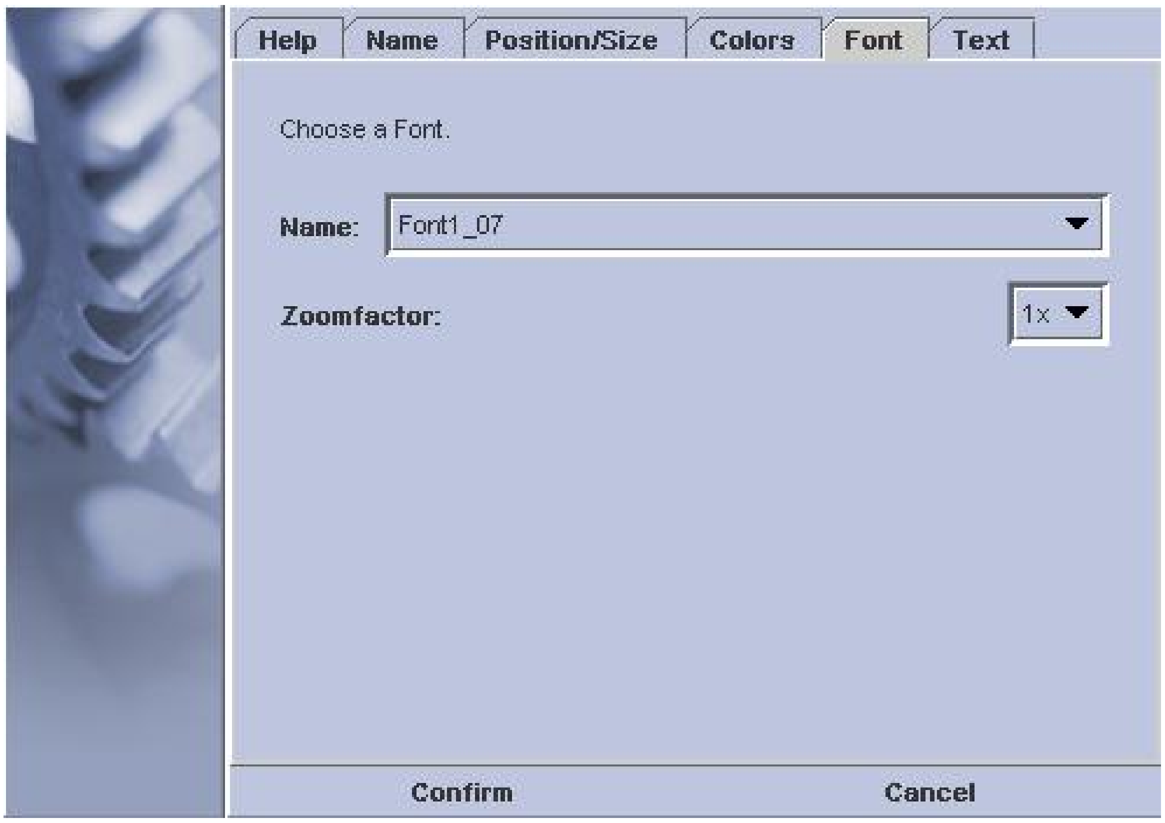
Position and dimension of the element can be determined with the mouse or at the dialogue.



X-Position: 76 Y-Position: 3

Width: 70 Hight: 16

By elements with fonts the used character set can be selected.



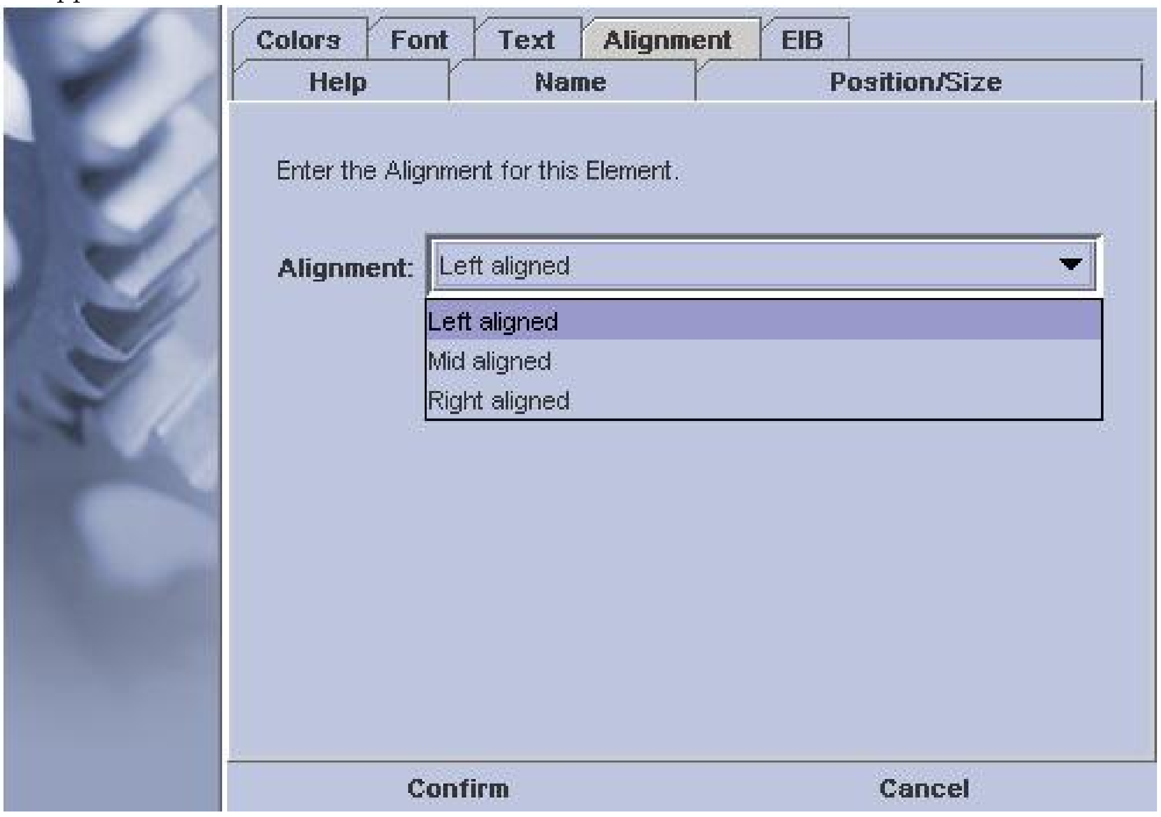
Choose a Font.

Name: Font1_07

Zoomfactor: 1x

The installed character sets could be used, but character sets of one's one could also be produced with the FontEditor.
Big script fonts need a lot of internal memory, therefore big fonts can also be produced with a zoom of smaller character sets. The respective outlines are not as smooth as the original big character sets.
In many elements the organisation of the texts can be adjusted. You can choose: left adjusted, centre or right adjusted. This is especially important for varying texts and decimal number displays for an

optimal appearance.

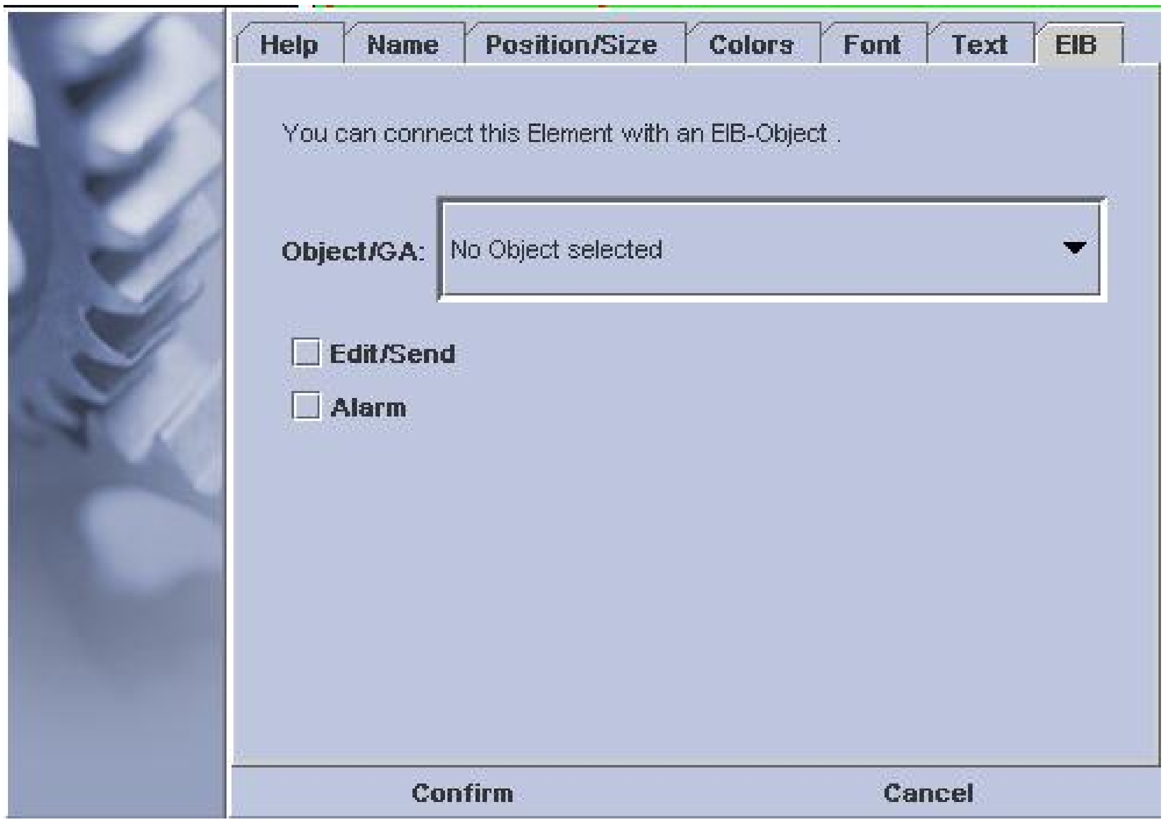


Enter the Alignment for this Element.

Alignment: Left aligned

Left aligned
Mid aligned
Right aligned

Under the dialogue window EIB the data point to use can be selected for this element , if it is an EIB/KNX- element.



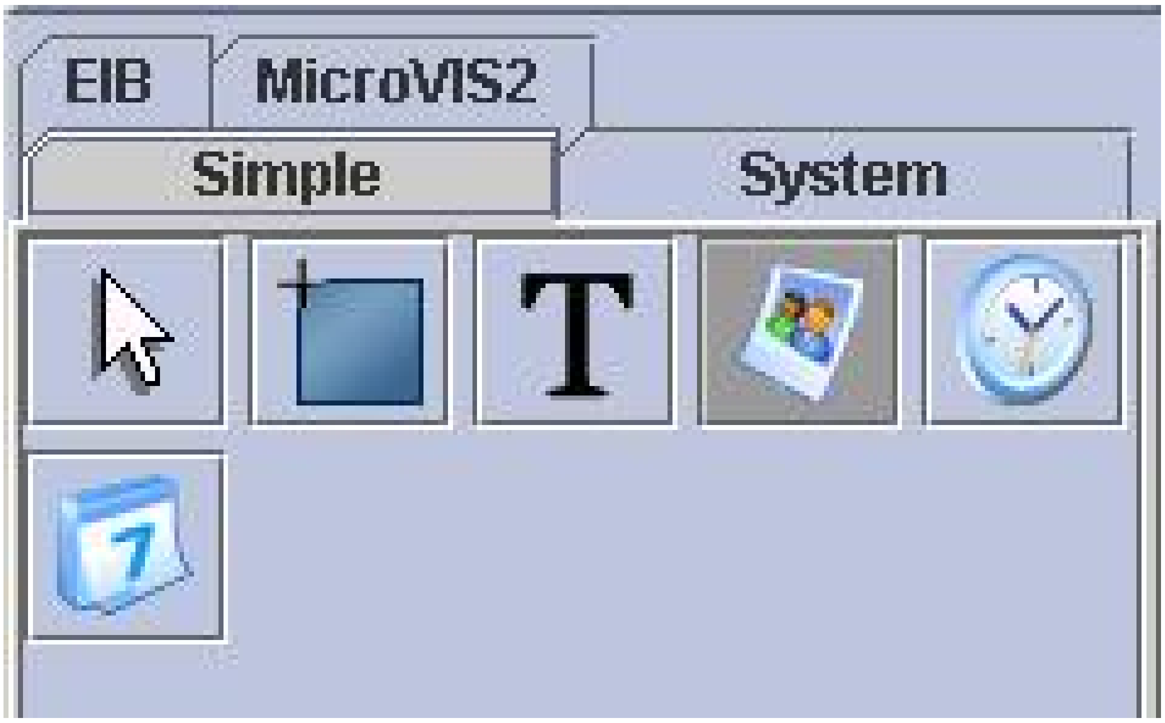
You can connect this Element with an EIB-Object .

Object/GA: No Object selected

☐ Edit/Send

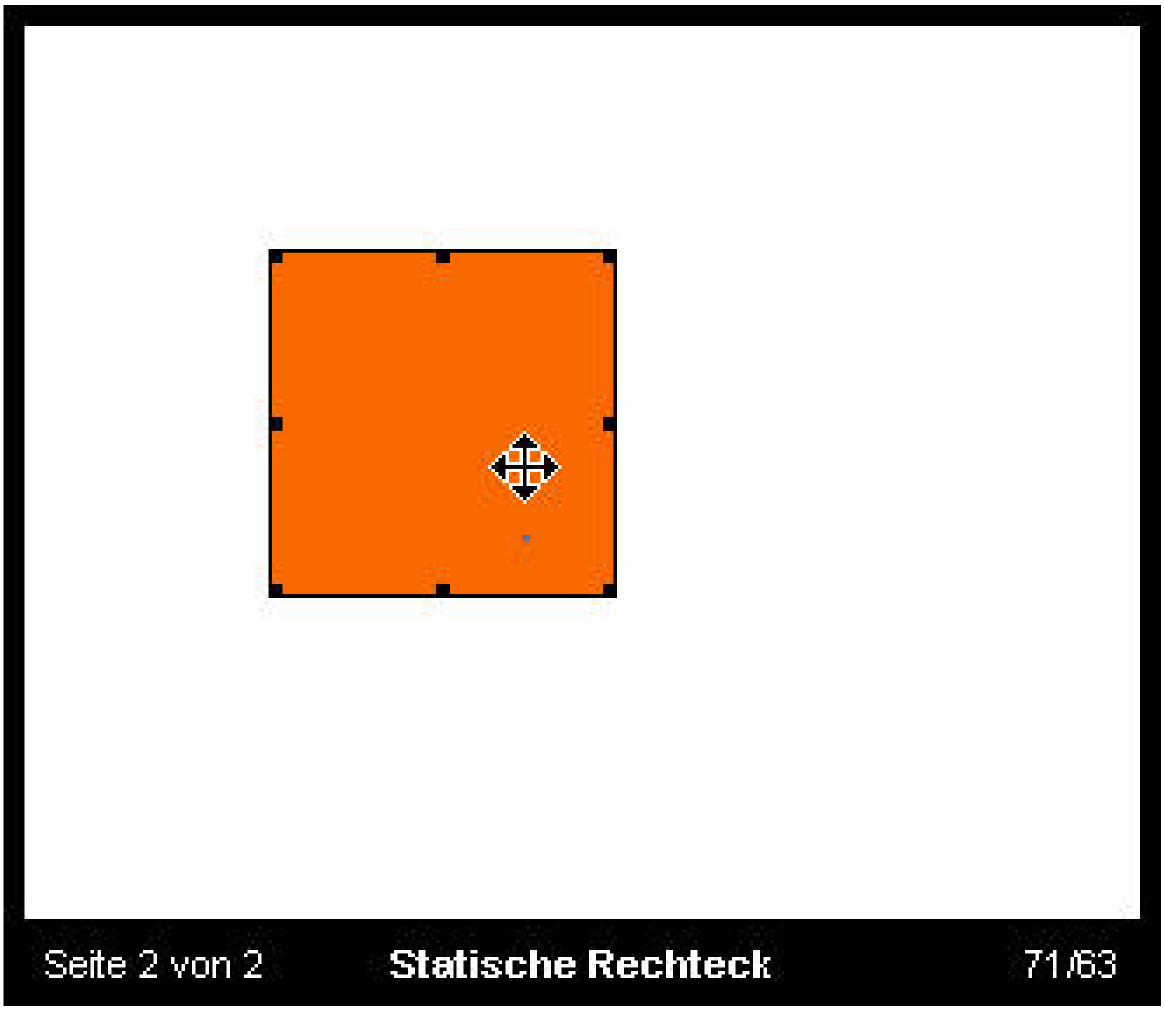
☐ Alarm

3.7.1 Static Elemts



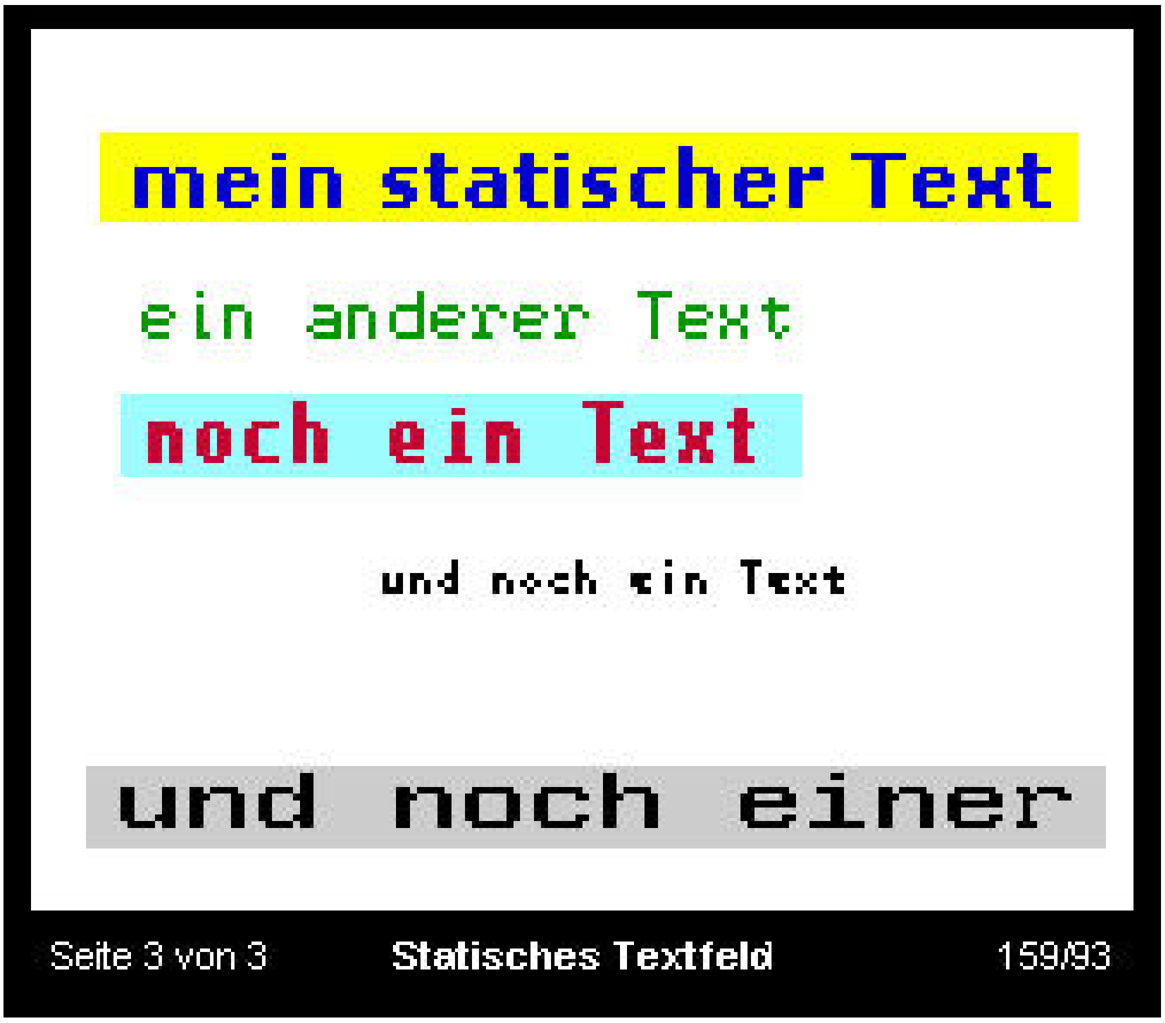
Static elements are typically used for an optical apperance of the page or for marking and explanations.

3.7.1.1 Static Rectangle



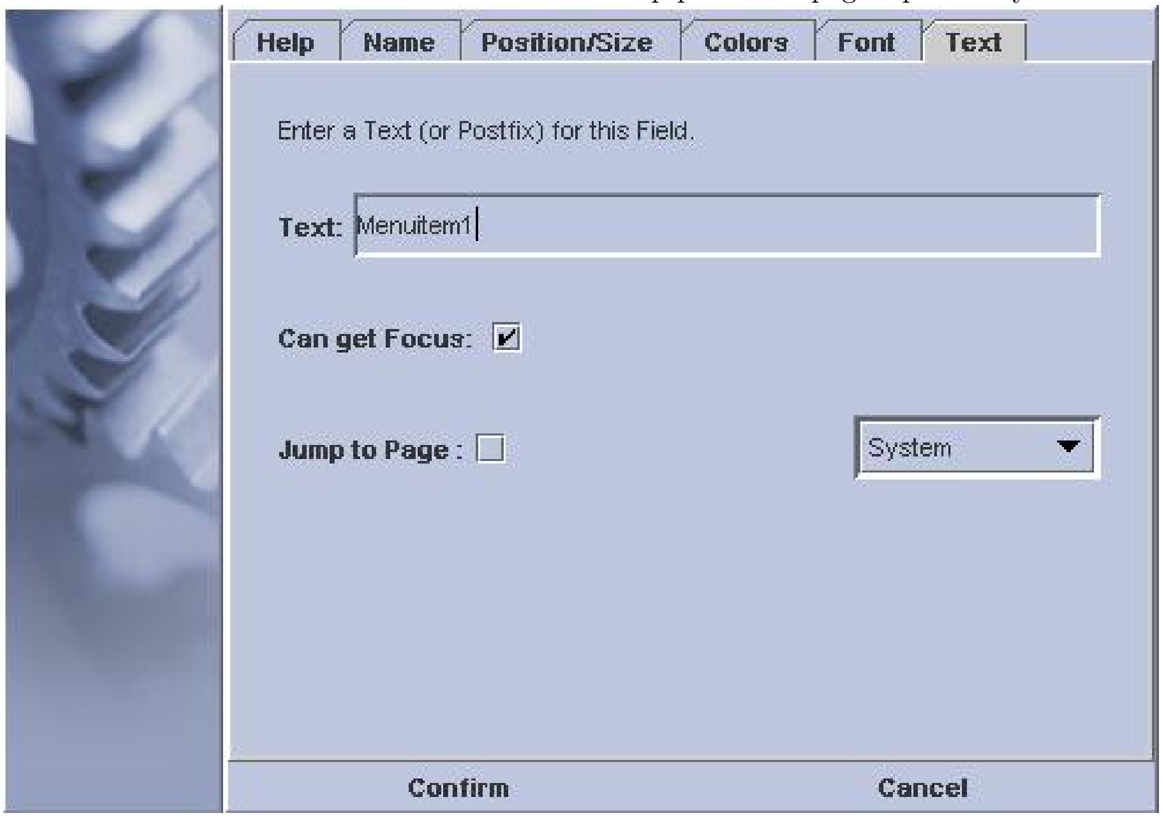
Static rectangles are coloured elements for an optical page configuration with free position and a free selectable size.

3.7.1.2 Static Textfield


mein statischer Text
ein anderer Text
noch ein Text
und noch ein Text
und noch einer
Seite 3 von 3 Statisches Textfeld 159/93

Static texts can be positioned at any place on a page. Position, character set, background colour as well as foreground colour can be chosen freely.
Texts, not selectable by an operating button, can be turned transparent, so that images underneath are not disturbed by the background of the text characters.

Static texts can also be defined as menu- items to call up particular pages specifically.



Enter a Text (or Postfix) for this Field.

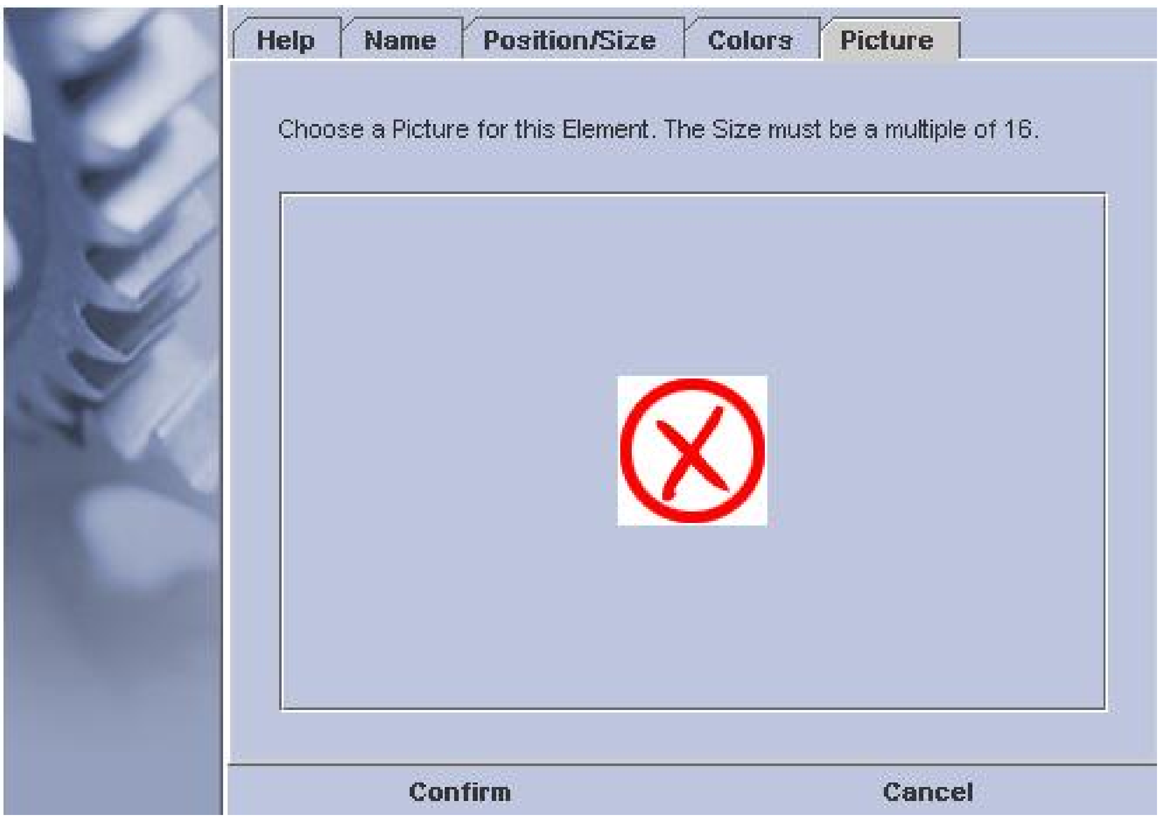
Text: Menuitem1

Can get Focus: ✓

Jump to Page : ☐ System

To be used as a menu item, the Focus has to be permitted and the page jump has to be defined.

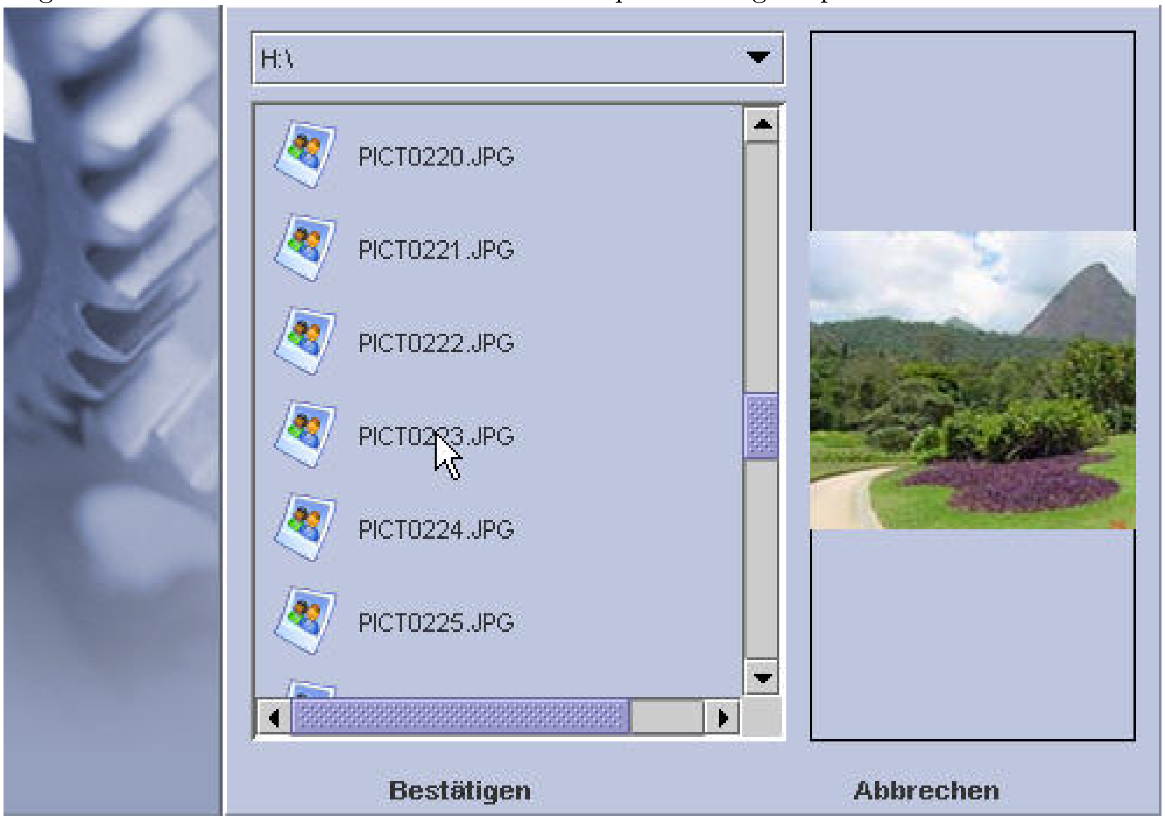
3.7.1.3 Static Image



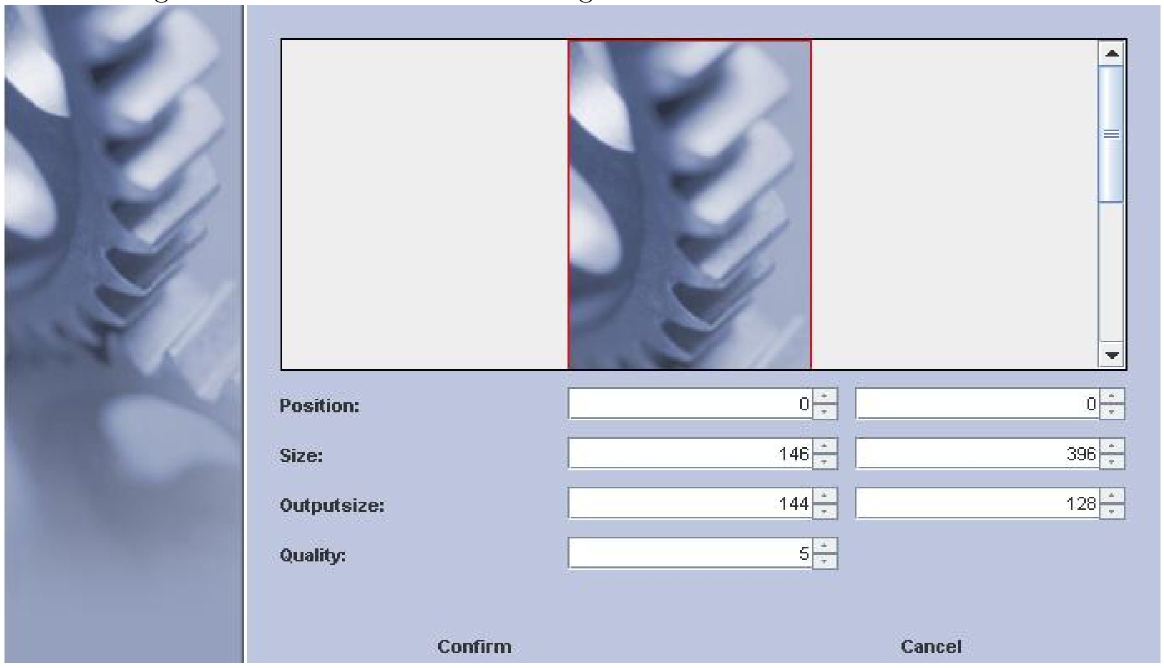
Choose a Picture for this Element. The Size must be a multiple of 16.

Static images can be used for the page configuration. They can cover the whole display area or only parts of it. Any JPEG files can be used and if necessary, scaled by the program.

For image selection click at the red element and an option dialogue opens.



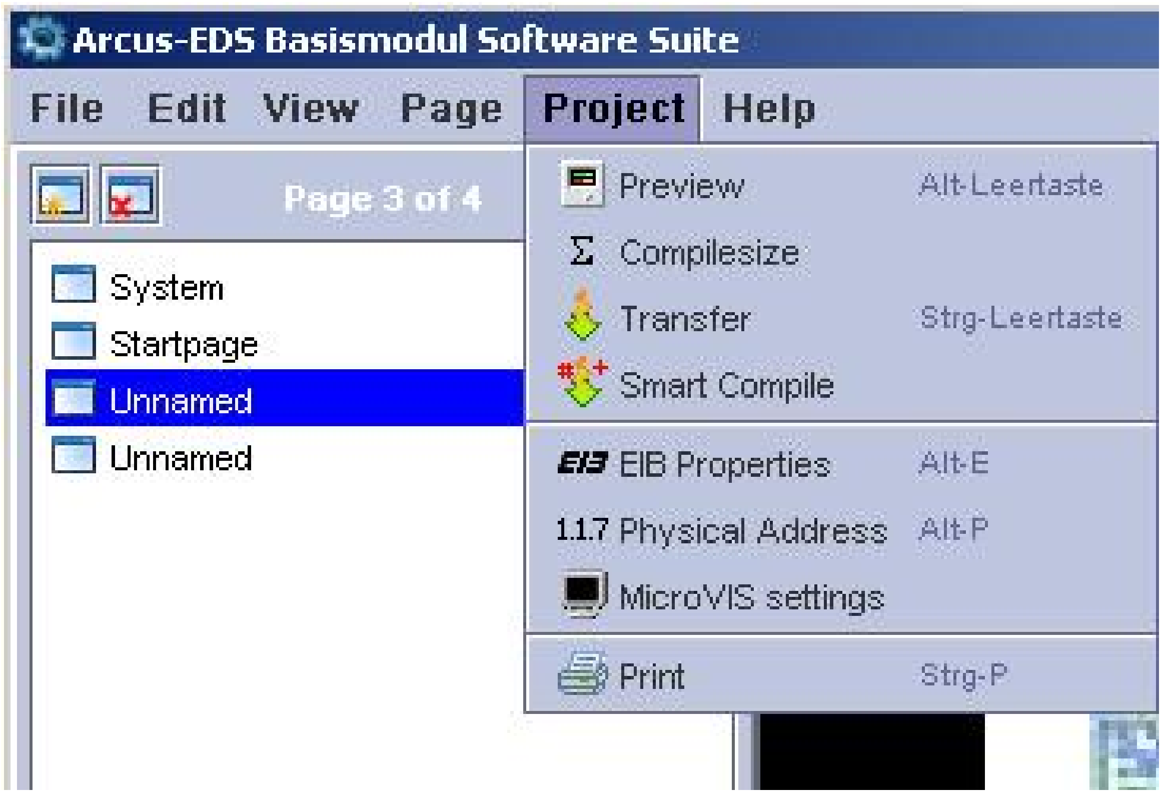
The selected image has now to be scaled for the right scale.



Position: 0 0
Size: 146 396
Outputsize: 144 128
Quality: 5

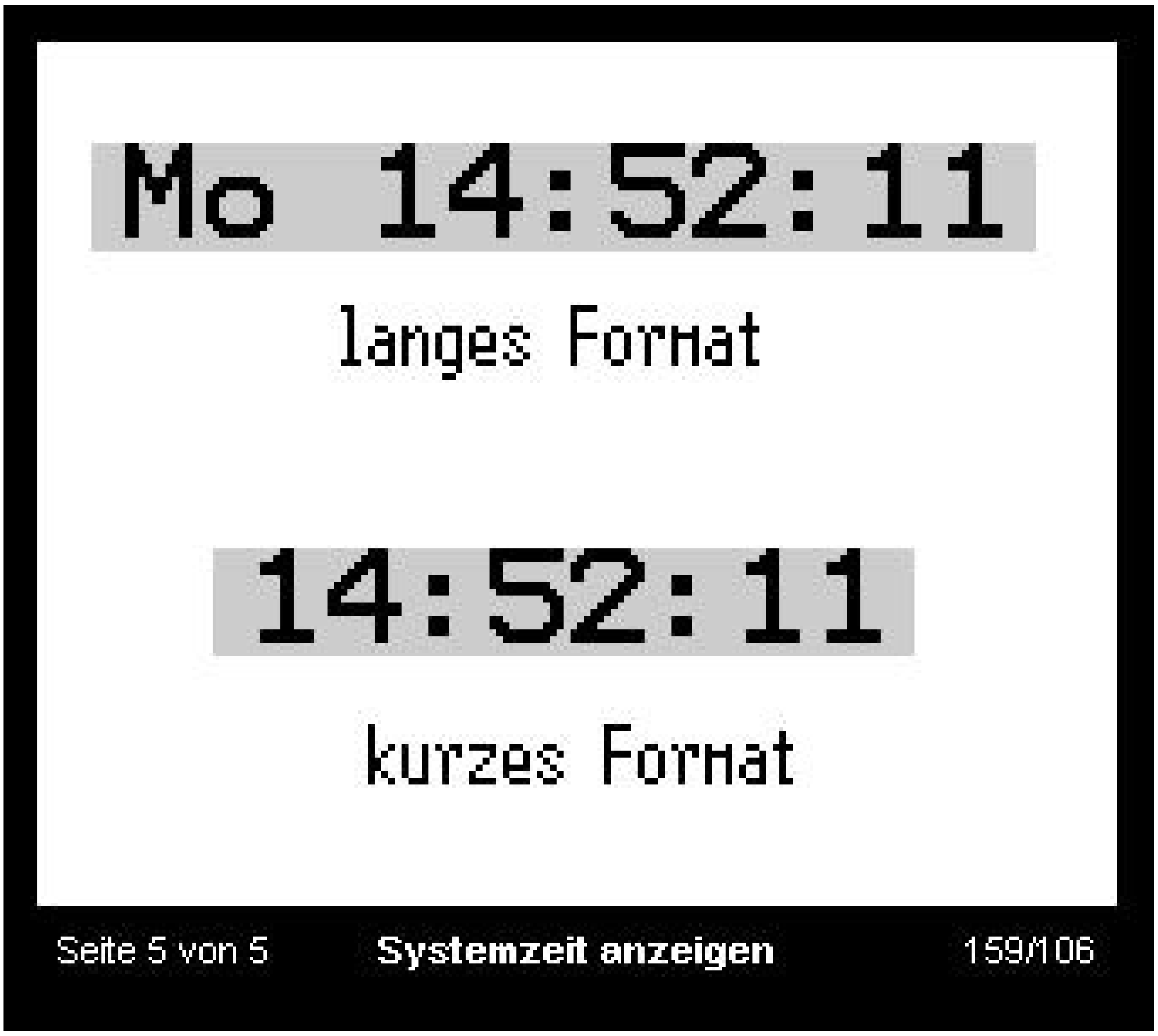
The Quality (and Datasize) of the image can be adjusted between 0 (lowest quality) and 10 (highest quality). A value of 5 is adequate for most cases.

Images (especially with high quality) need a lot of memory space, so your display could easily reach the limit of its memory capacity. Please control your memory requirement under **Project->Compilesize**.

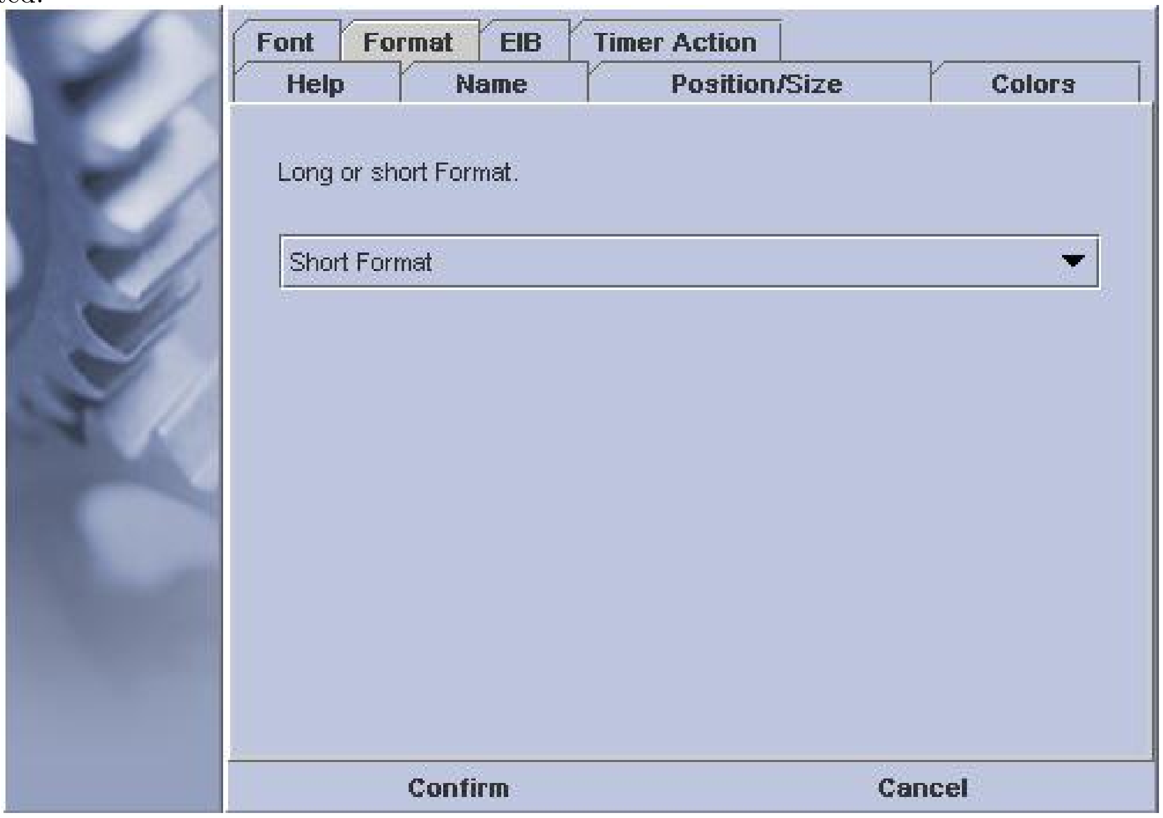


Before you insert your images, edit them with an image editing program and optimise the size of image and memory. Some image programs offer a function like:
'save for web' to further minimise your image- memory.
With a memory scale from 3 to 6 KB for every background complete image (128px high, 160px broad), you can get very good image qualities.

3.7.1.4 Showing Systemtime


Mo 14:52:11
langes Format
14:52:11
kurzes Format
Seite 5 von 5
Systemzeit anzeigen
159/106

The system time is a regularly updated element of the operating system, which cannot be edited or selected.

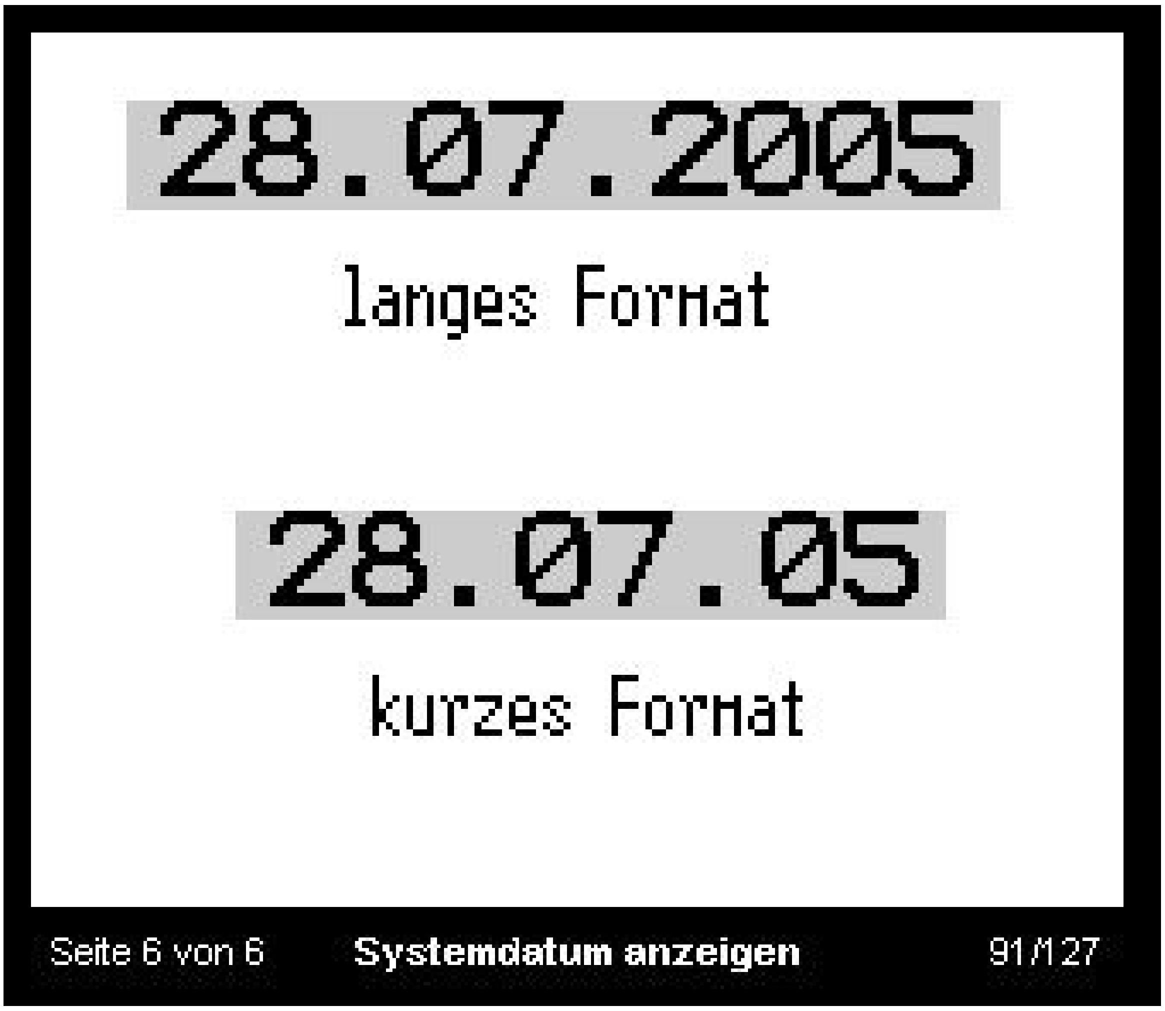


Long or short Format.

Short Format

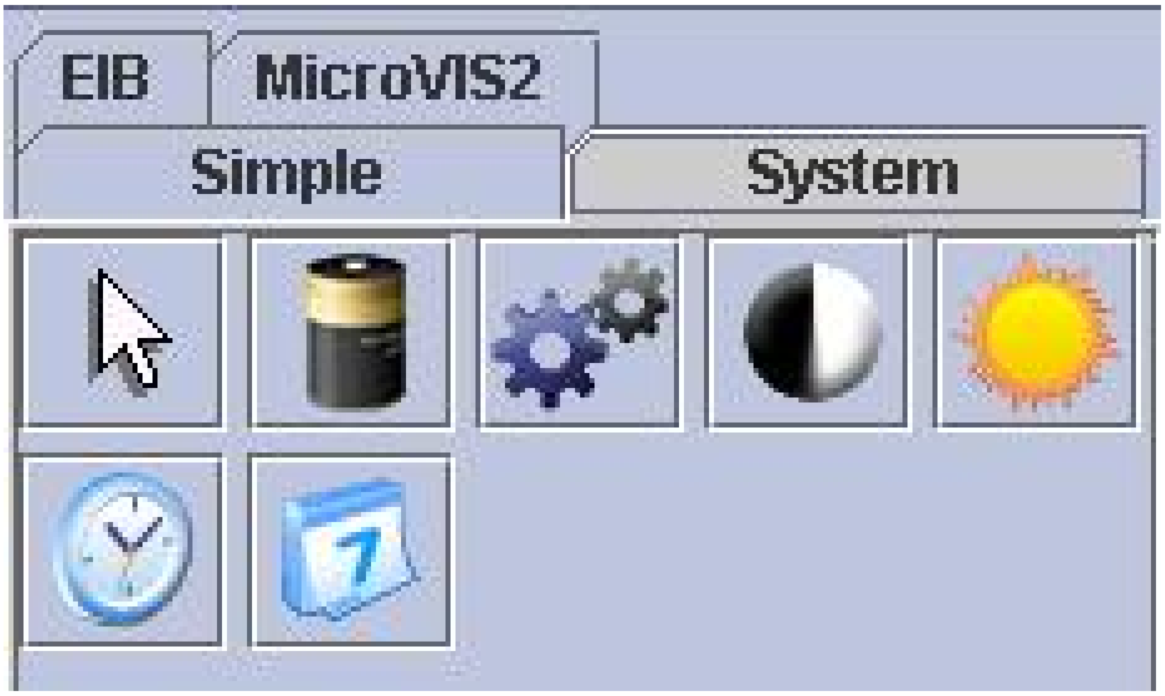
The system time can be displayed by a long (containing a weekday) or short format.
The character set to display as well as the foregroundcolour and backgroundcolour can be freely chosen.

3.7.1.5 Showing Systemdate



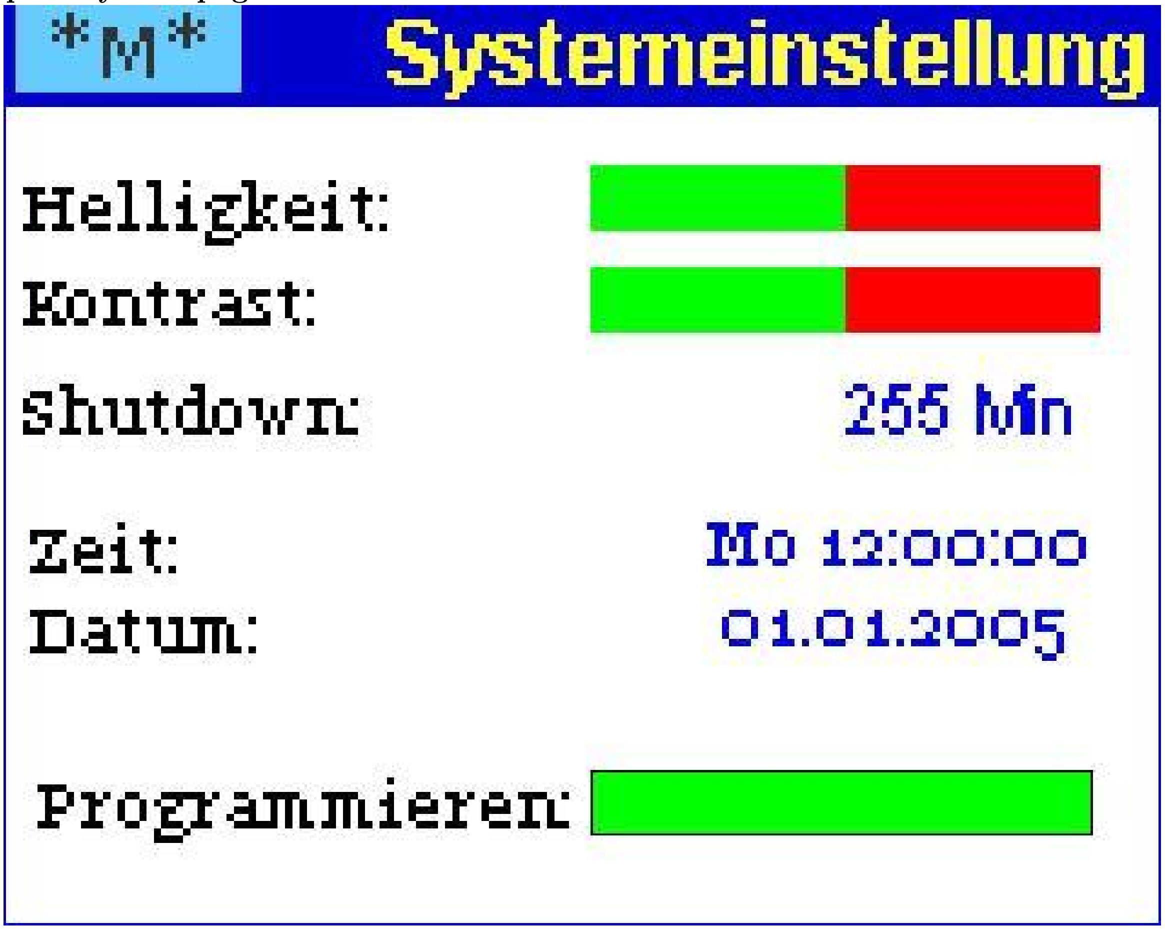
The system date is a regularly updated element of the operating system, which cannot be edited or selected. The system date can be displayed with a long (4 digit year) or short (2 digit year) format. The character set to display as well as the foregroundcolour and backgroundcolour can be freely chosen.

3.7.2 System Elements



System elements are elements for setting system properties.Typically they are summarised on one page, the systempage. None of the elements must be available, often some are useless or the useraccess is not wanted.

A typical system page:



M

Systemeinstellung

Helligkeit:

Kontrast:

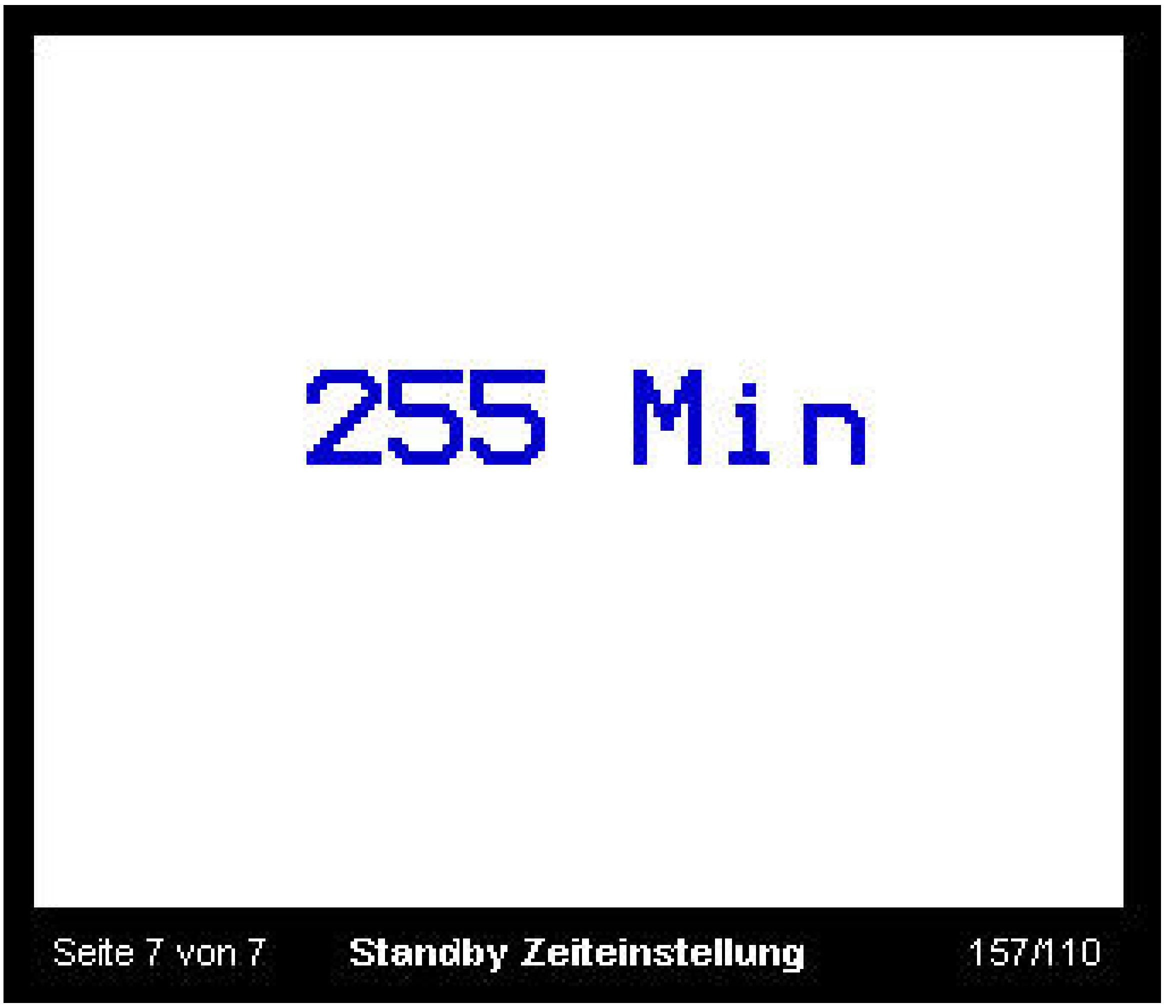
Shutdown: 255 Min

Zeit: Mo 12:00:00

Datum: 01.01.2005

Programmieren:

3.7.2.1 Standby Timeout



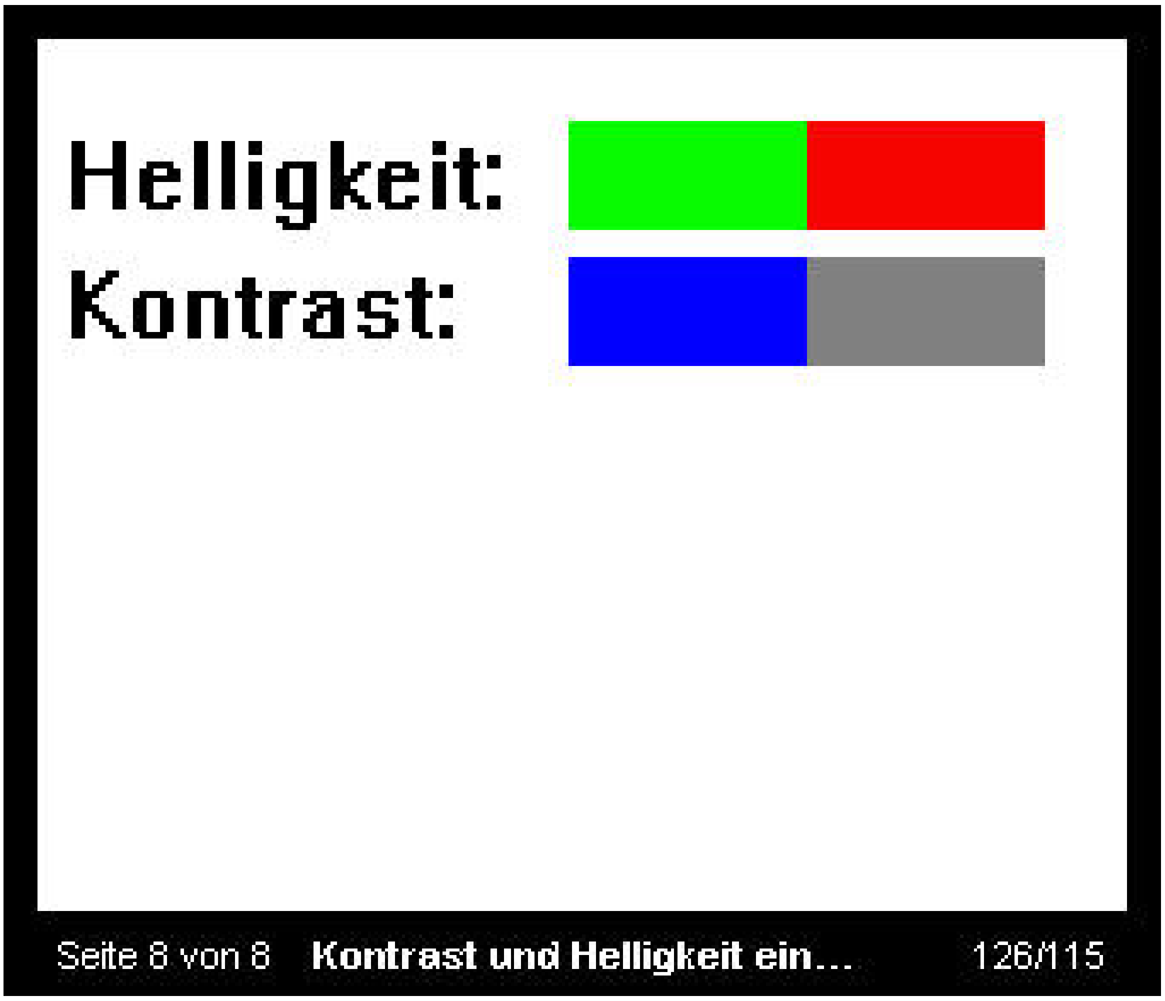
With this system element you can adjust the time, after which the display and the background light go into the standby modef. The value can vary from 1 to 255 minutes, a value of 0 means no standby at all. Character set, colours, adjustment and position can be set.

3.7.2.2 Physical Address

To program the physical address with a dummy application of ETS, the MicroVIS display has to be switched to the programming mode with the button "programming physical address". By operating the programming element, the LCD display at the ETS is in the programming modus. Then it cannot be operated any more, all functions are stopped.
The physical address is also transmitted with the USB download (look at "actual/ current project"),

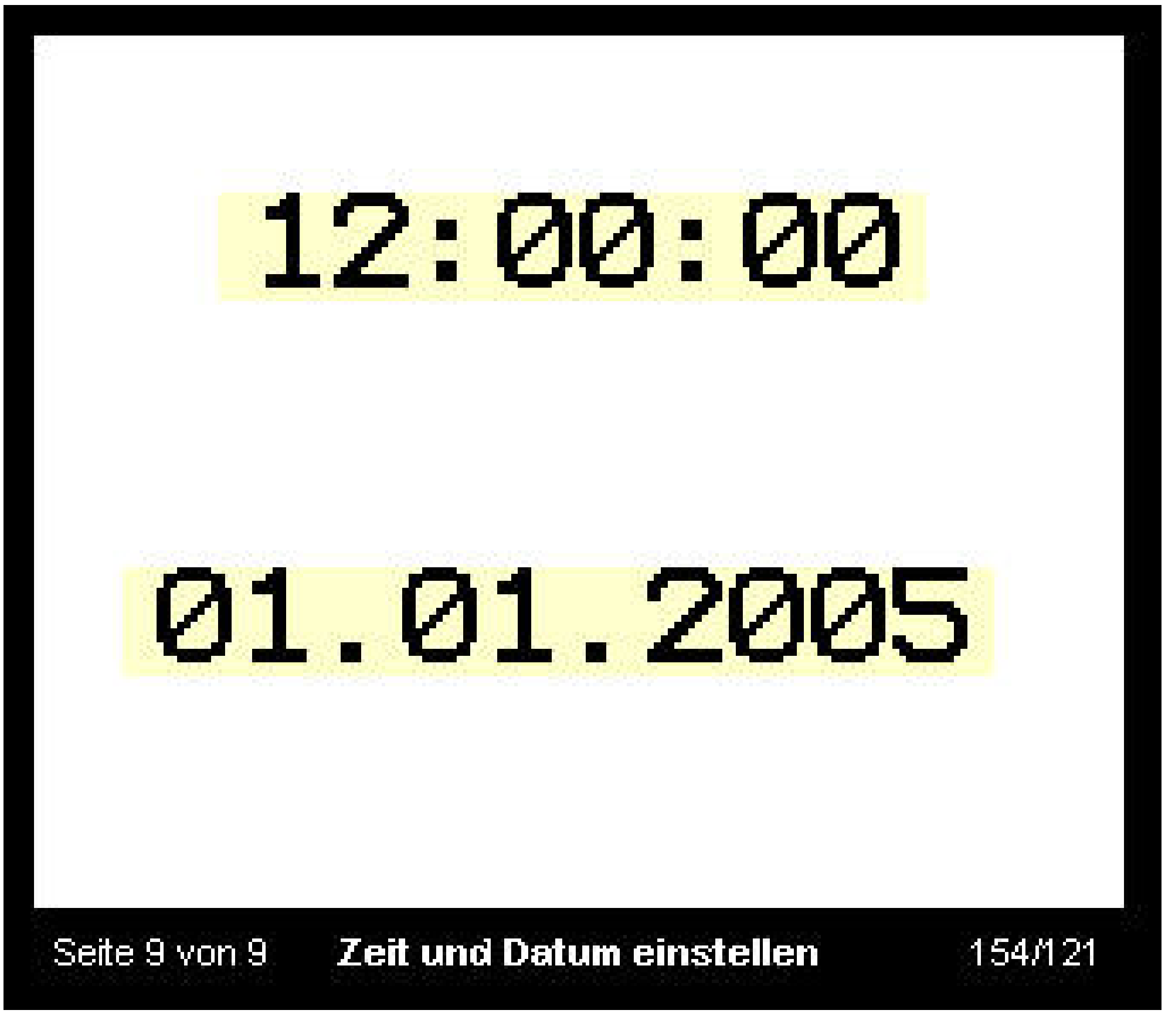
an existing ETS- programming will be overwritten and has to be repeated. At the menu item "physical address" the appropriate address has to be entered. The physical address will be transmitted with every USB download.

3.7.2.3 Contrast and Brightness



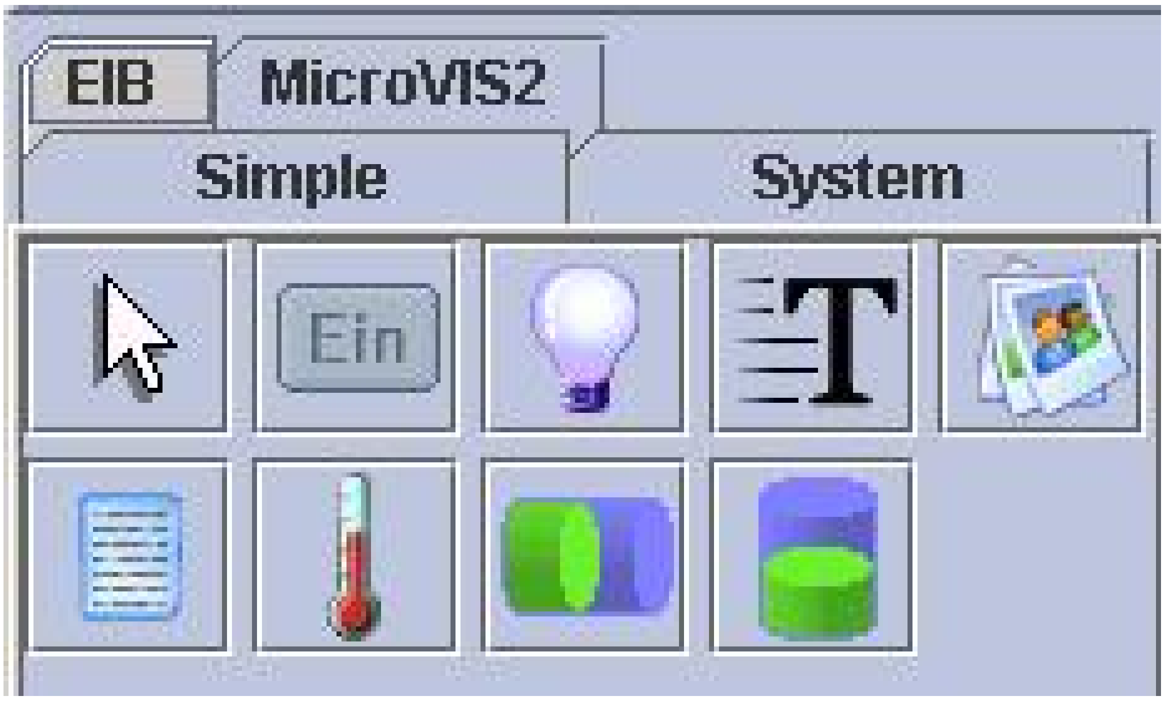
With these elements the user has the possibility to adapt brightness and contrast of the display to his requirement. Foreground and background colours of the element details define the colours of the left and right side of the bargraph.

3.7.2.4 Setting Time Date

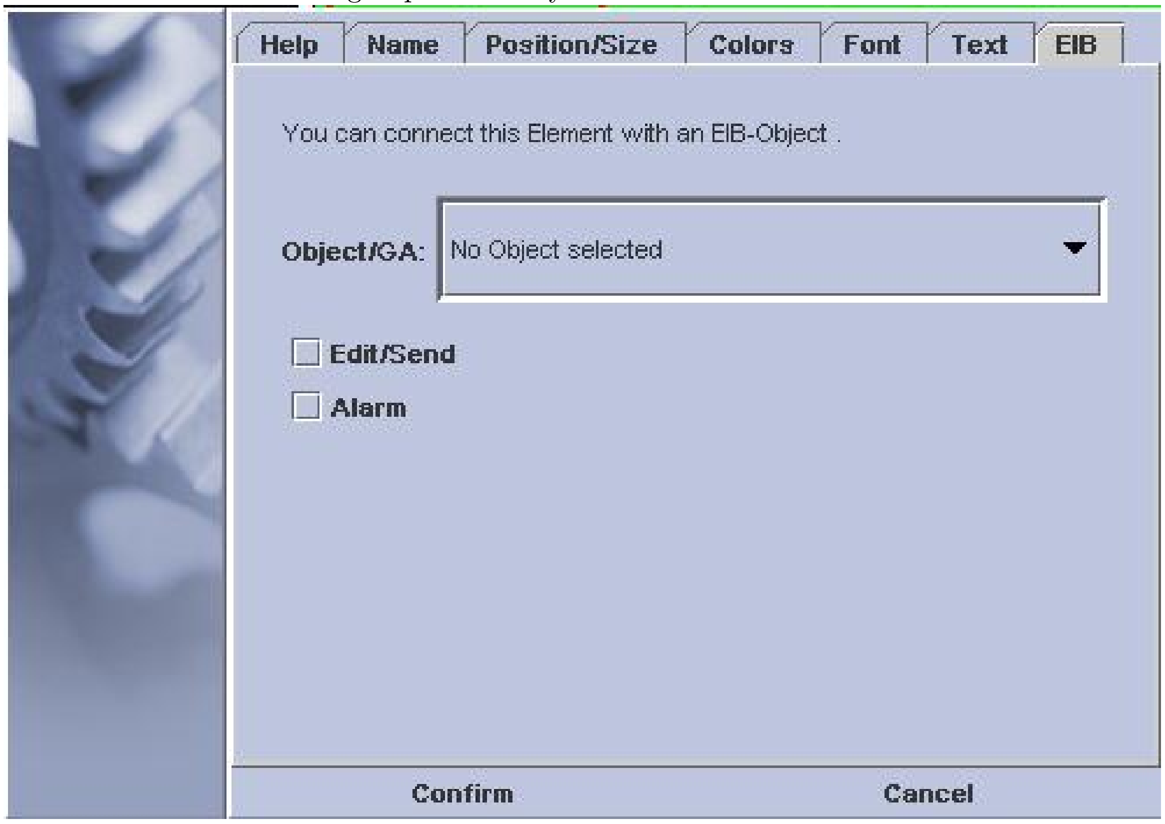


These elements allow settings of date and time manually. If the system time is available over the EIB/KNX Bus, they must be used as well. In the latter case the elements have to be connected with the EIB Groupaddresses. Then they are not selectable and not editable at the device anymore.
Character set, format, alignment, colour and position can be chosen freely.

3.7.3 EIB Elements



EIB/KNX elements are elements for displaying and setting of values in the EIB/KNX bus system. These elements offer the usable group address by choice.



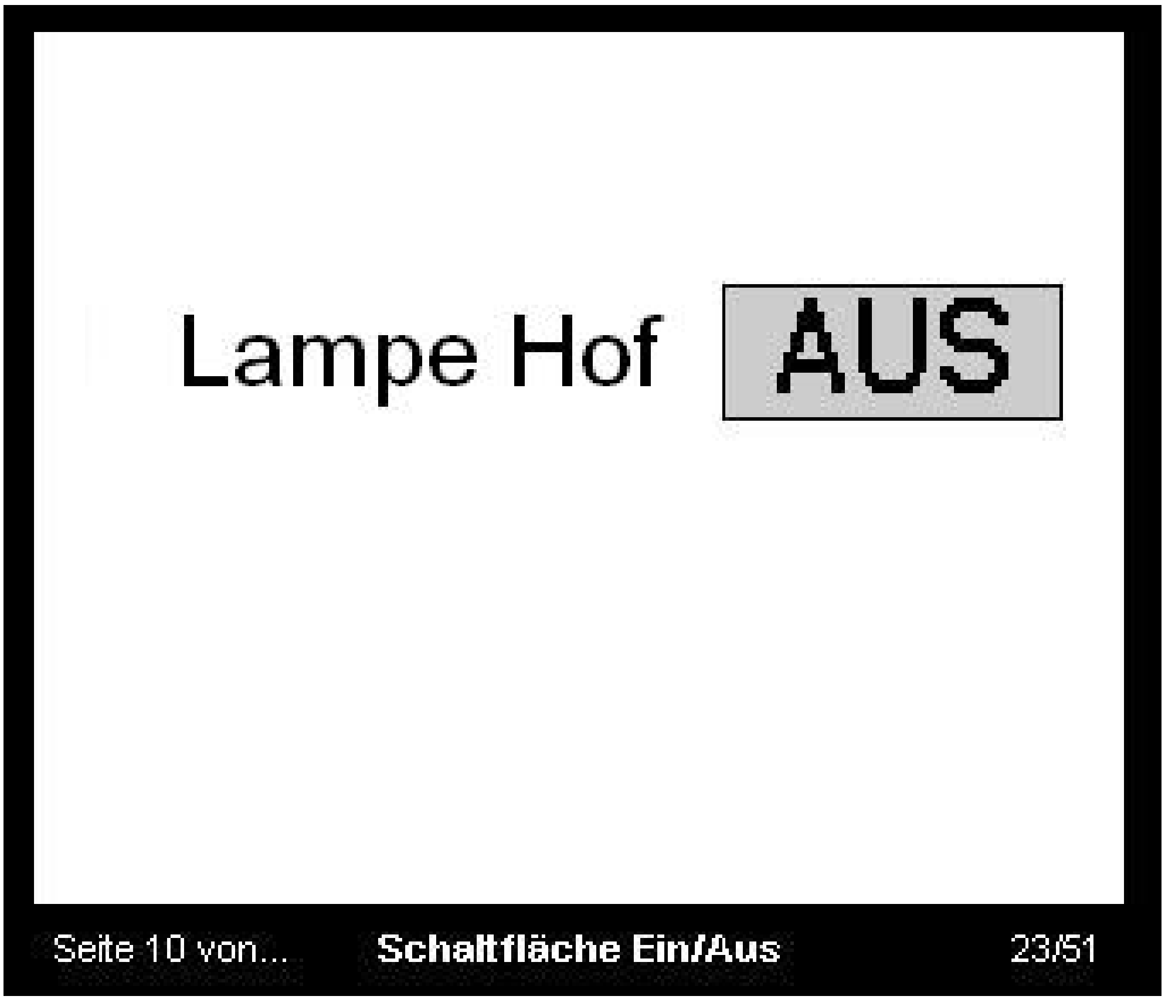
You can connect this Element with an EIB-Object .

Object/GA: No Object selected

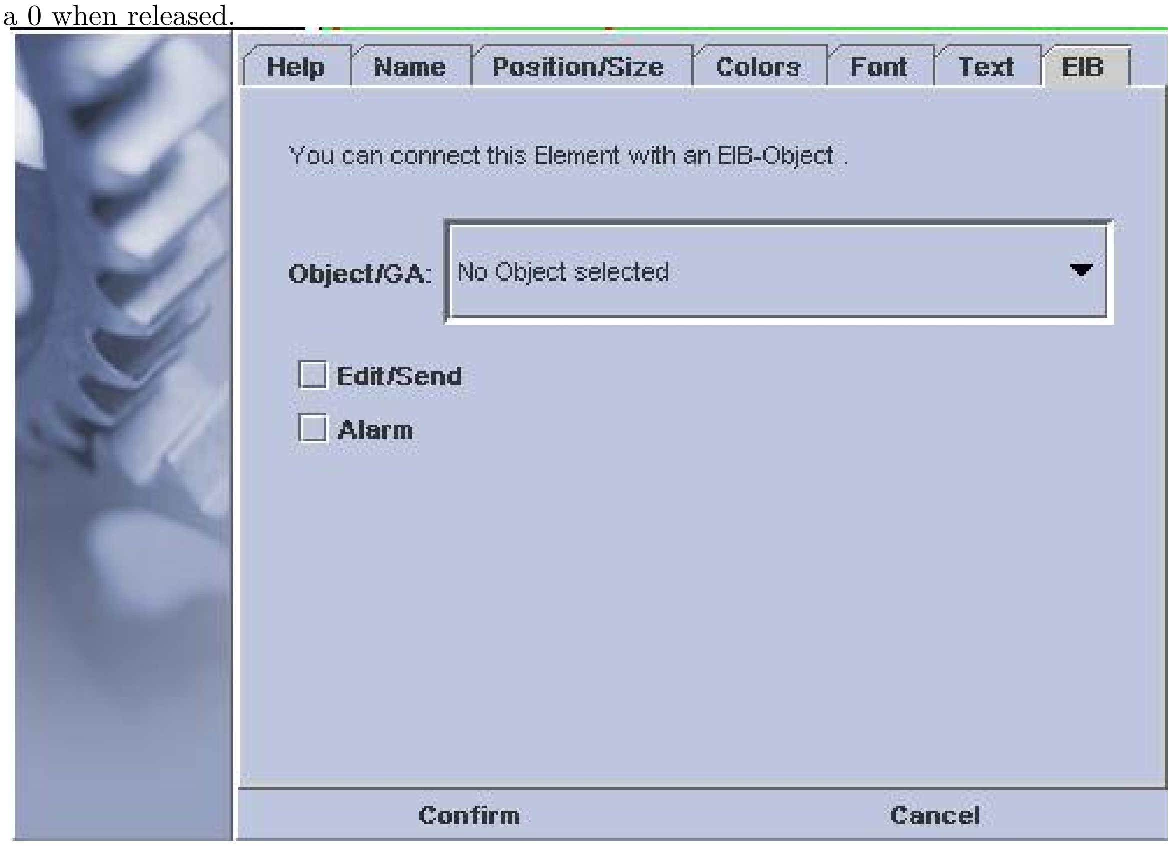
☐ Edit/Send
☐ Alarm

The field **Edit/Send** has to be activated if the element should be eligible on the page so that a value can be sent to the bus. If the field "**Alarmfunction**" is active, the page of this element is called up automatically at a value change. The field "**Pushbutton**" enables buttons that send a 1 by pressing the field and a 0 by releasing the field.

3.7.3.1 Switches and Buttons



Switch and Pushbutton areas can be distinguished in the object setting through the checkbox **Pushbuttonfunction**. Switches toggle their state witch each operation. Pushbuttons send a 1 when pressed and a 0 when released.

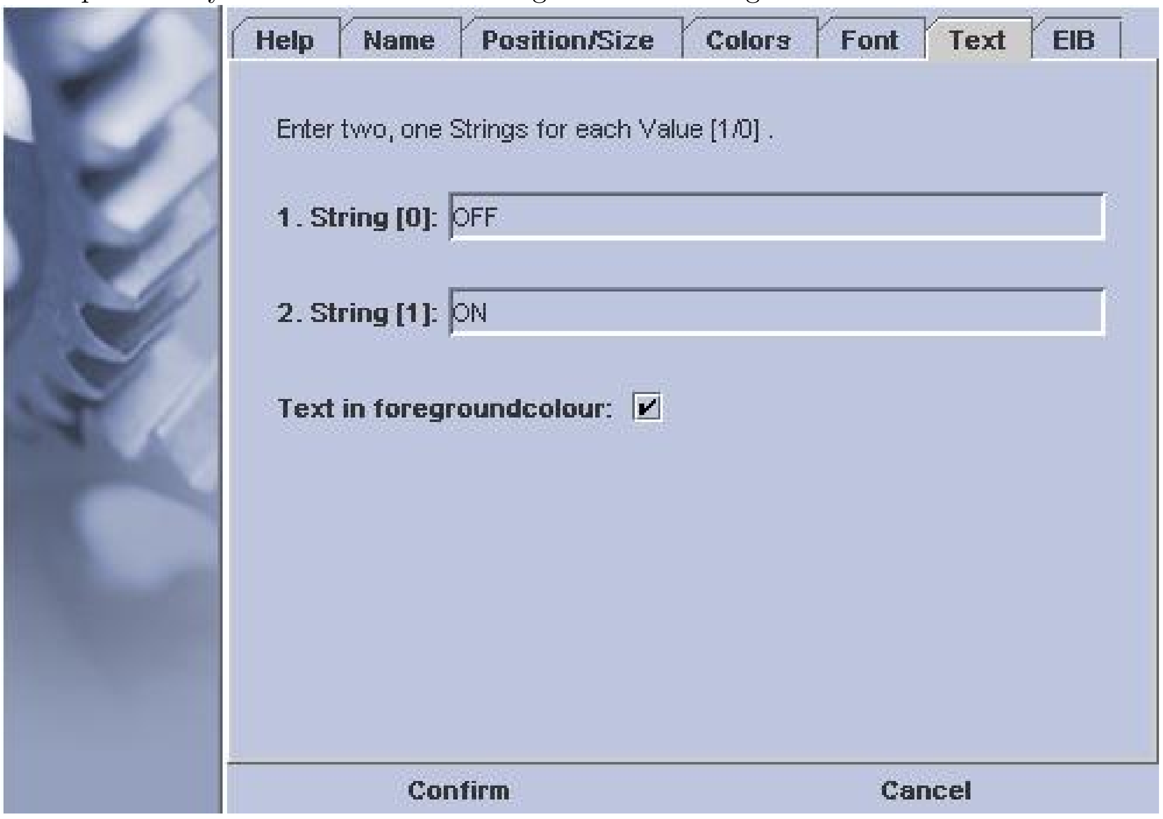


You can connect this Element with an EIB-Object .

Object/GA: No Object selected

The function "long pressing" is deactivated at a pushbutton.

Depending on the current state the colour of the element changes between foreground and background colour. The setting "Text in a complementary colour" can be selected otherwise the Text is always black. Complementary colour refers to the foreground and background colours.



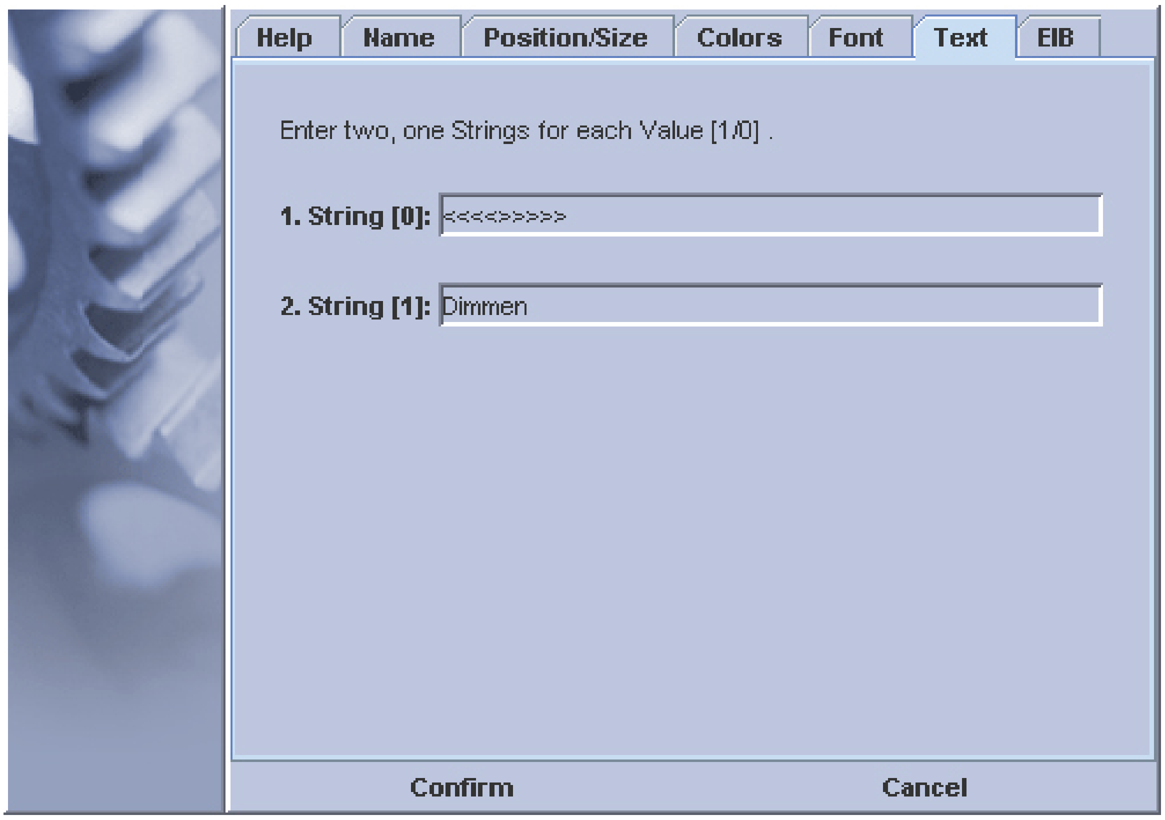
Enter two, one Strings for each Value [1/0] .

1. String [0]: OFF

2. String [1]: ON

Text in foregroundcolour: ✓

3.7.3.2 Dimmer



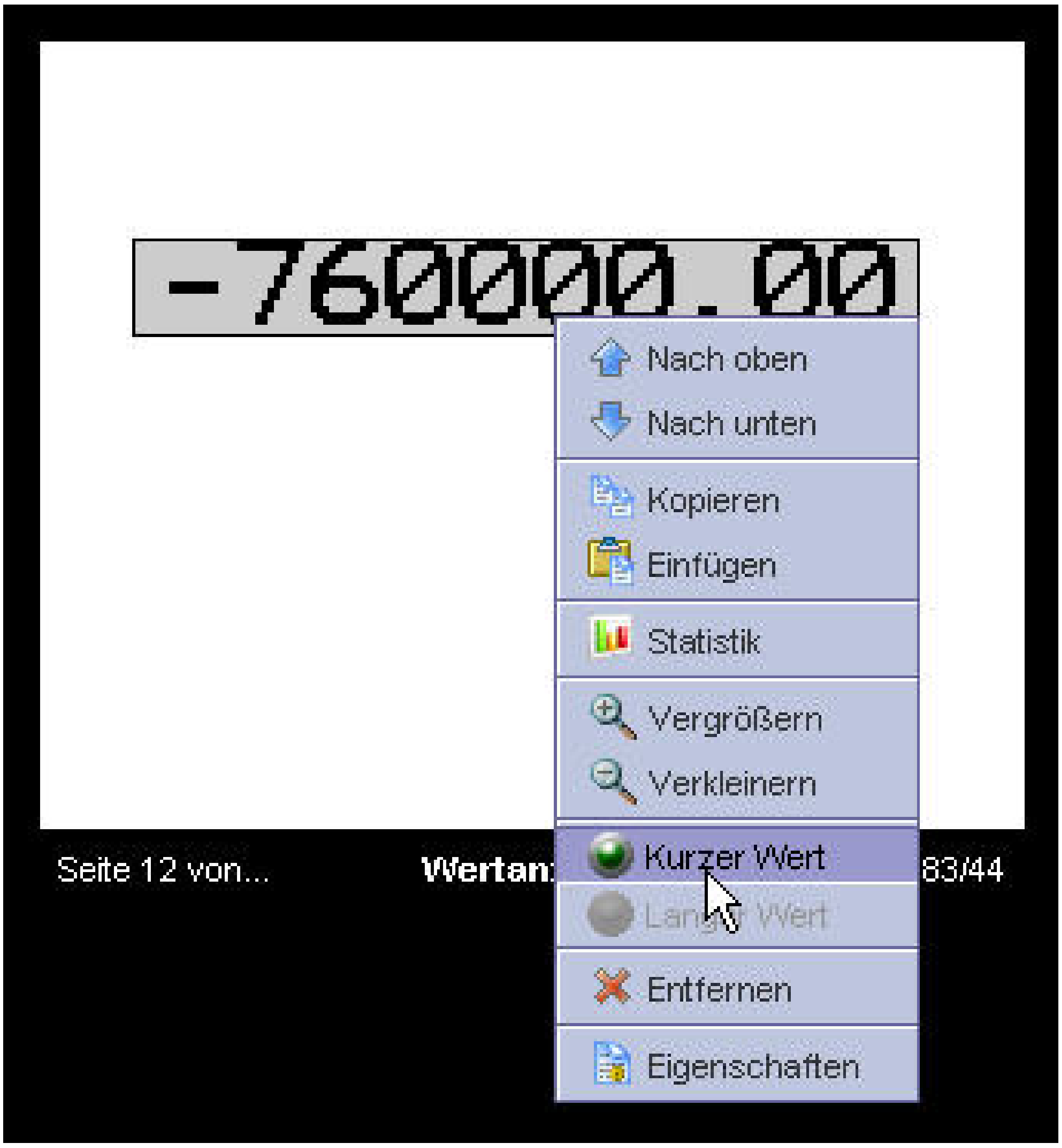
Enter two, one Strings for each Value [1/0] .

1. String [0]: <<<<>>>>>

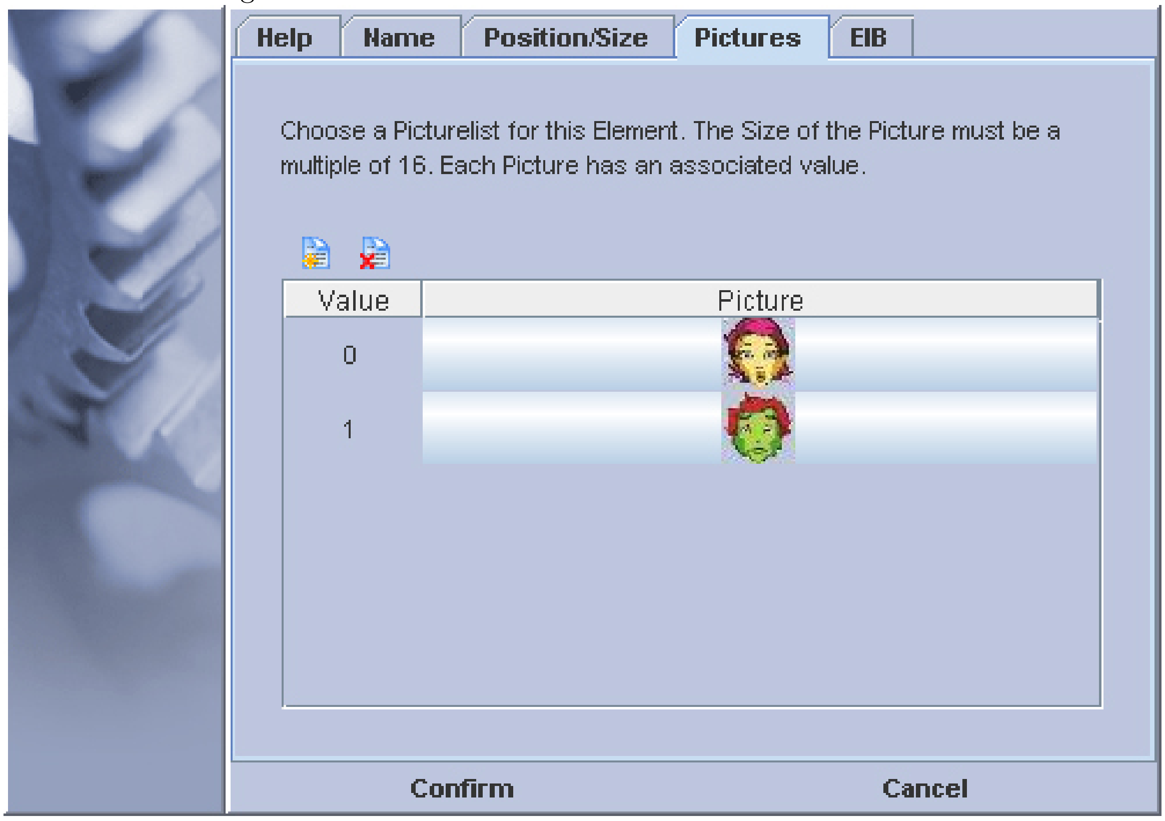
2. String [1]: Dimmen

Dimmer- elements can only be connected with a 4 bit dimm-object. By selecting an object, text and colours can change in accordance with the settings. Turning the rotary- switch iniciates sending the dimmer- value directly.
Turning left: dimming down.
Turning right: dimming up.

3.7.3.3 Values



Value elements show values of a data-point in human readable stile (e.g."25.40° C"). Details of the value elements are character set, alignement, position, colours and units. The unit is attached to the end of the value and aligned with it.

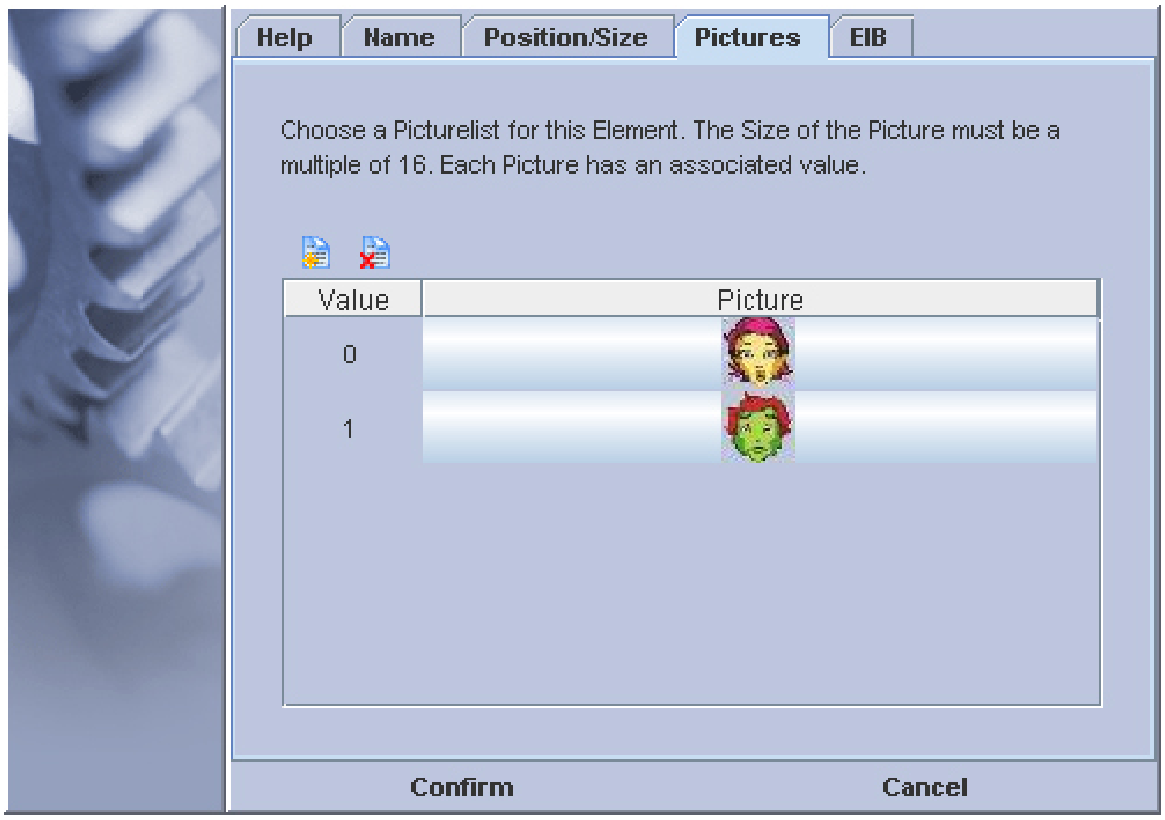


Choose a Picturelist for this Element. The Size of the Picture must be a multiple of 16. Each Picture has an associated value.

Value	Picture
0	
1	

Value displays can be orientated left- adjusted, centred or right- adjusted to display changing length of display optimally. The effect of a different length of text on the optical impression can be tested through changing the length in the pop- up menu between "long value" and "short value".

3.7.3.4 List of Prices and Values

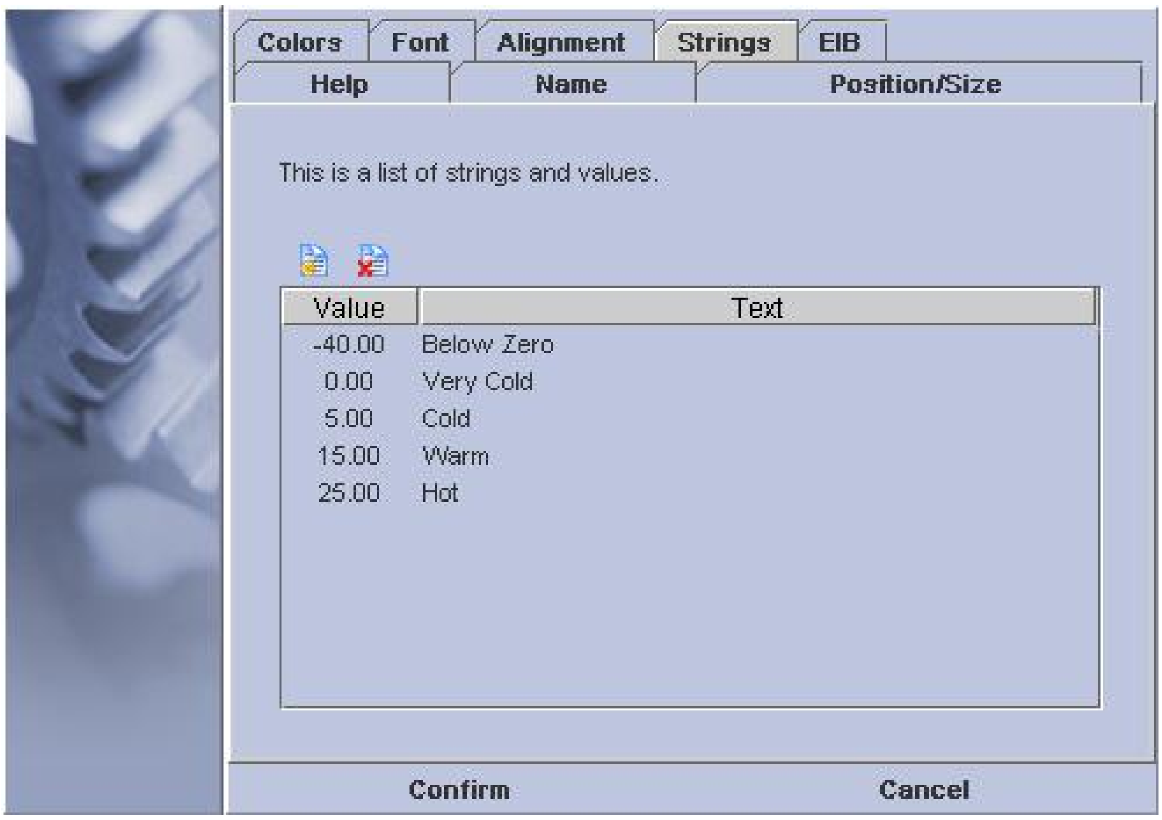


Choose a Picturelist for this Element. The Size of the Picture must be a multiple of 16. Each Picture has an associated value.

Value	Picture
0	
1	

An image list is an allocation list between images and object values. The object-values are of threshold value character. In the sahowing case the first image is shown from 0, the second from 1 and up.
If a list contains only one element, the appropriate value is sent immediately when operated (no choice available). This can be used for single telegrams (send a "1" by pressing). If a list contains two elements,the value toggles directly, when operated.
Image-lists can only be used for objects with at most 2 byte data capacity .

3.7.3.5 List of Captions and Value

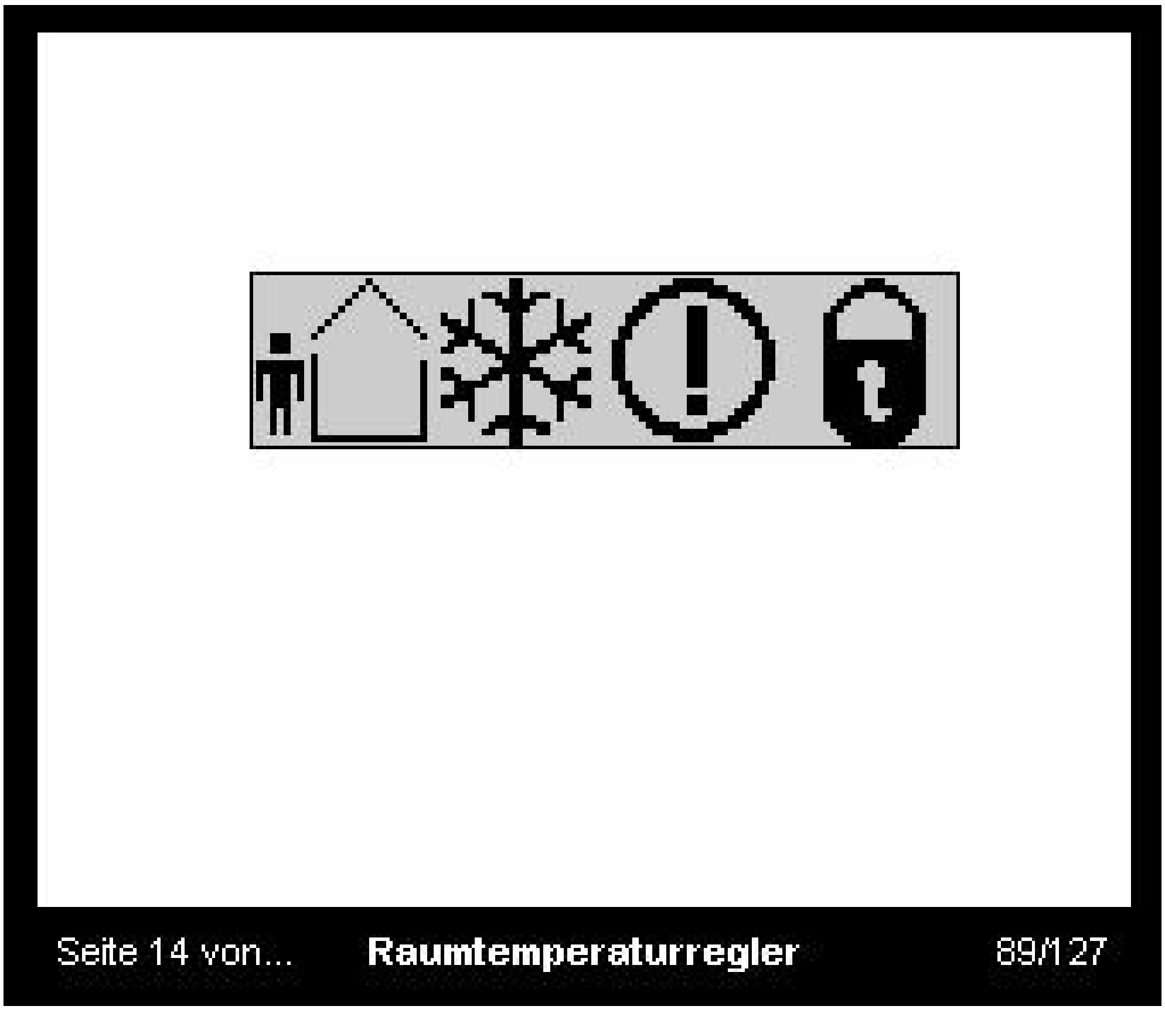


This is a list of strings and values.

Value	Text
-40.00	Below Zero
0.00	Very Cold
5.00	Cold
15.00	Warm
25.00	Hot

This list is an allocation list between text and object values. The object values have a threshold value character. In the showing case the text "sub 0" will be displayed from -40. 00, the text "cold" from 0. 00 and so on.
If a list contains only one element, the value will be sent immediately .
If a list contains only two elements, the values toggle .
Caption lists can be used only for objects with up to 2 byte data length.

3.7.3.6 Roomtemperatur



The room temperature thermostat presents the conditions of an external RTR with symbols . The following symbols are integrated:

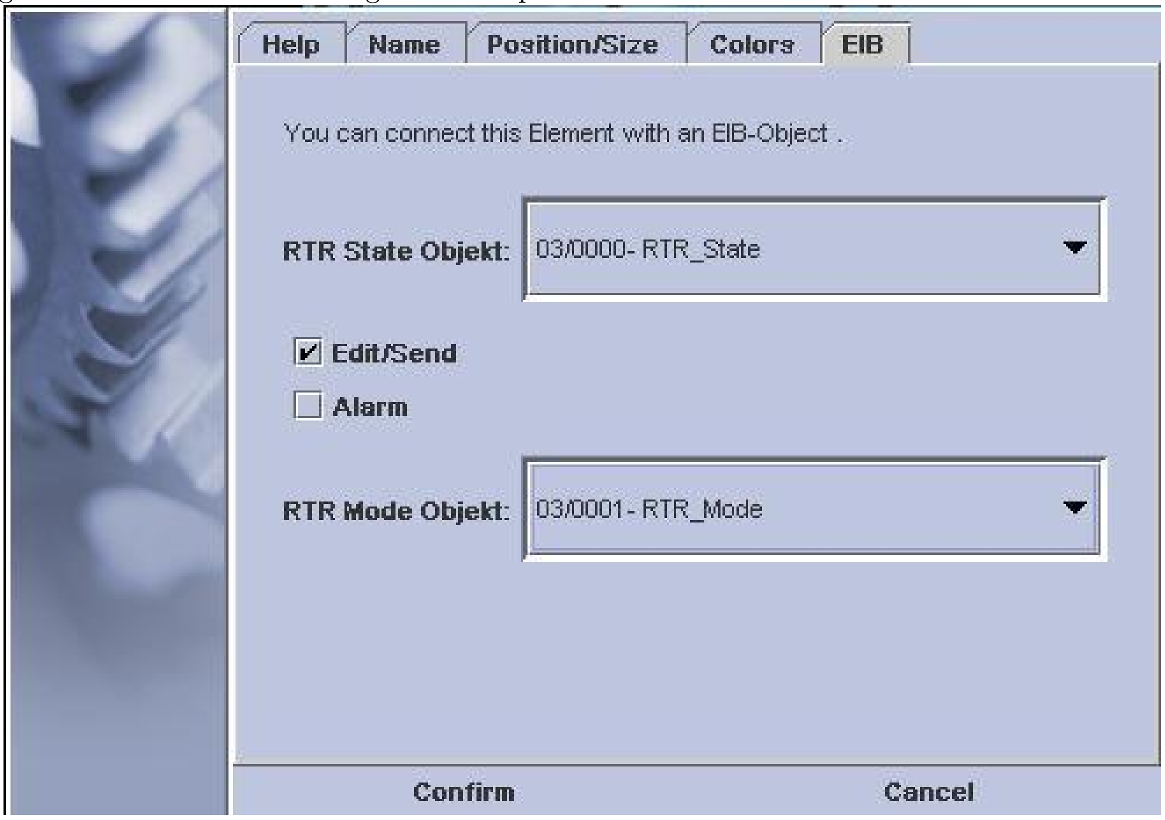
1. Symbol

- Comfort service
- Standby service
- Night service
- Frost/heat protection service

2. to 4. Symbol

- Heating/cooling
- Frost/Heat alarm
- Night service

In the Componenten Dialog you can enter a second object for the controller-mode. This object can change the controller mode setting if the component is marked as editable.



You can connect this Element with an EIB-Object .

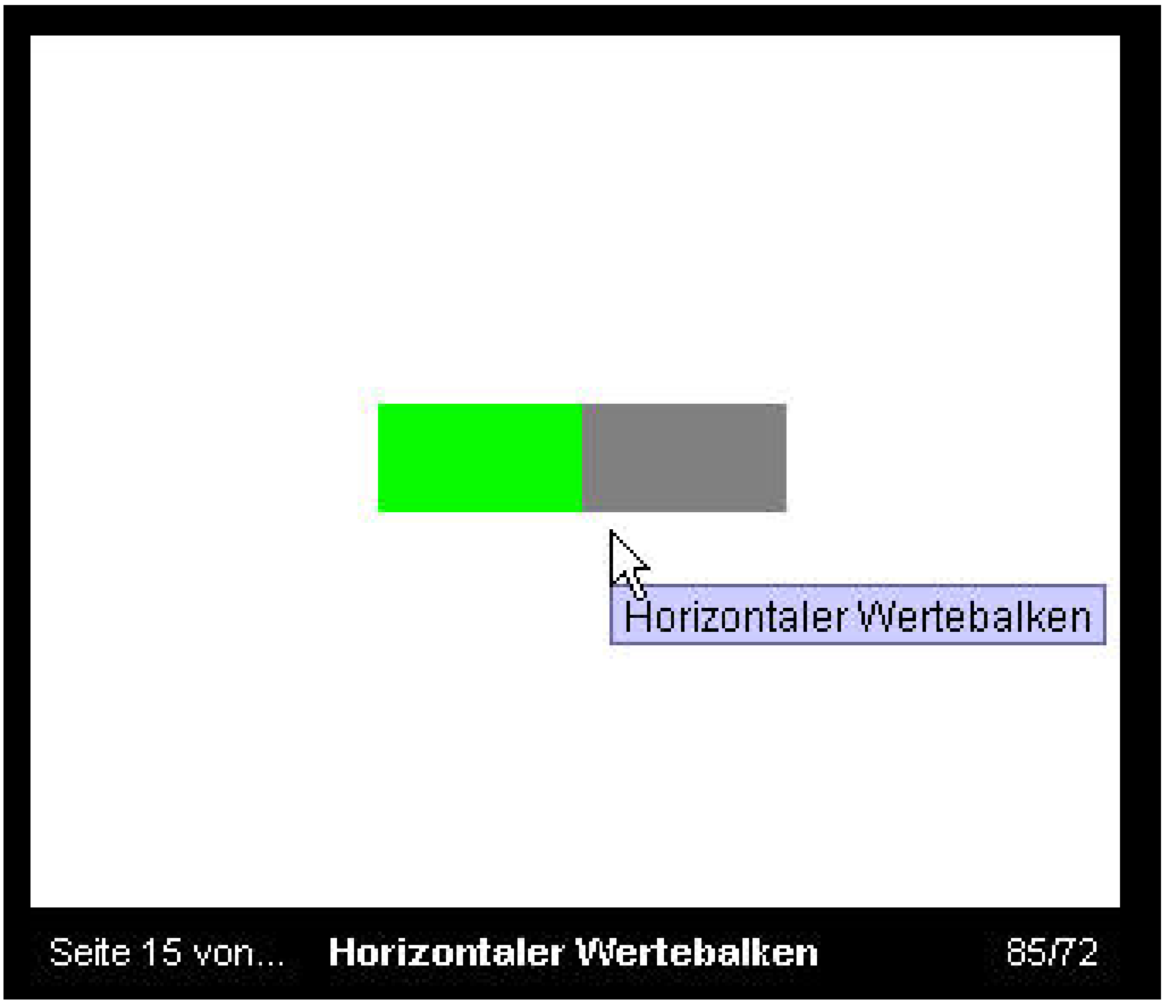
RTR State Objekt: 03/0000- RTR_State

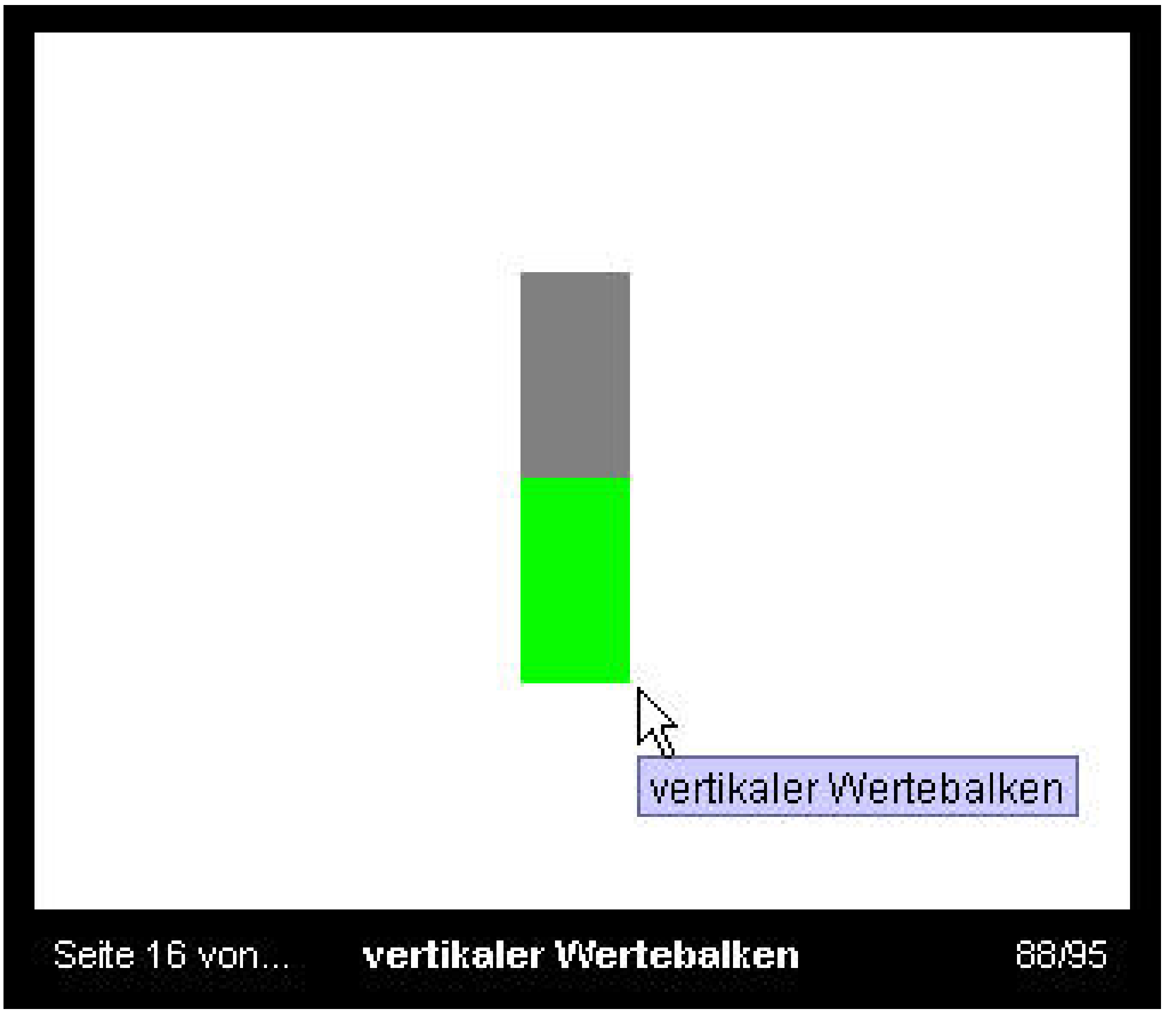
[✓] Edit/Send

[ ] Alarm

RTR Mode Objekt: 03/0001- RTR_Mode

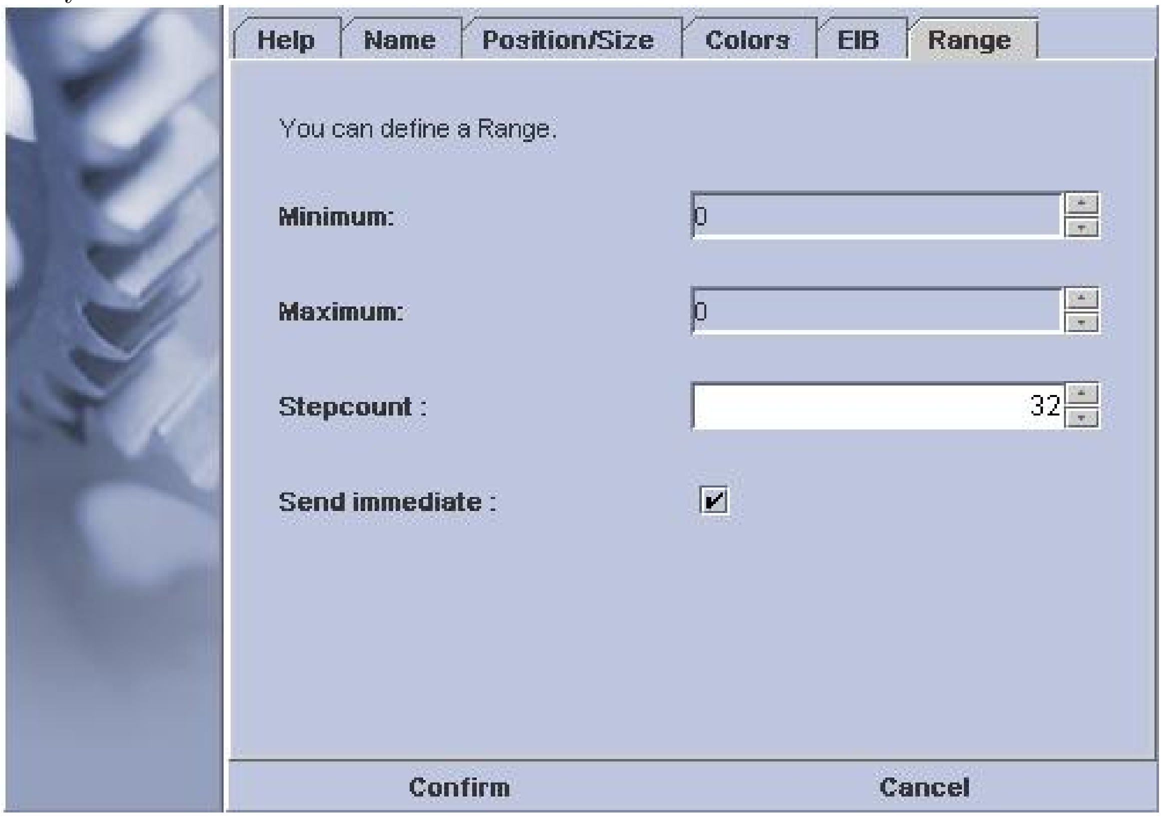
3.7.3.7 Value Bars





Bargraphs are displays and buttons with eligible colours and minimum- maximum settings.
If the bargraph has a send function you can choose: sending by confirmation (only once) or by rotation

immediatly.



You can define a Range.

Minimum: 0

Maximum: 0

Stepcount : 32

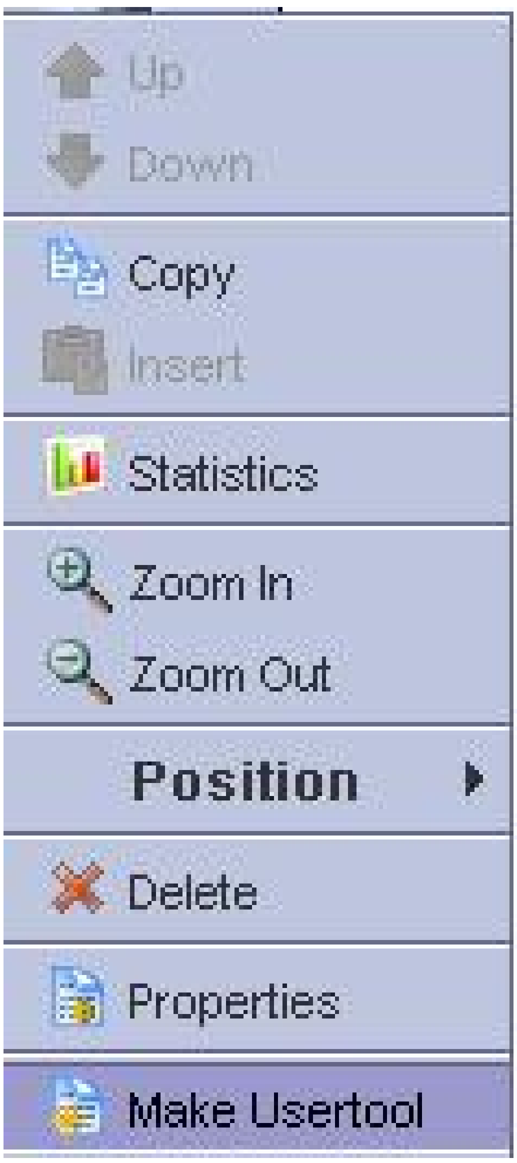
Send immediate : ✓

The indication: step-number shows how many rotation-steps are located between the minimum and maximum value.

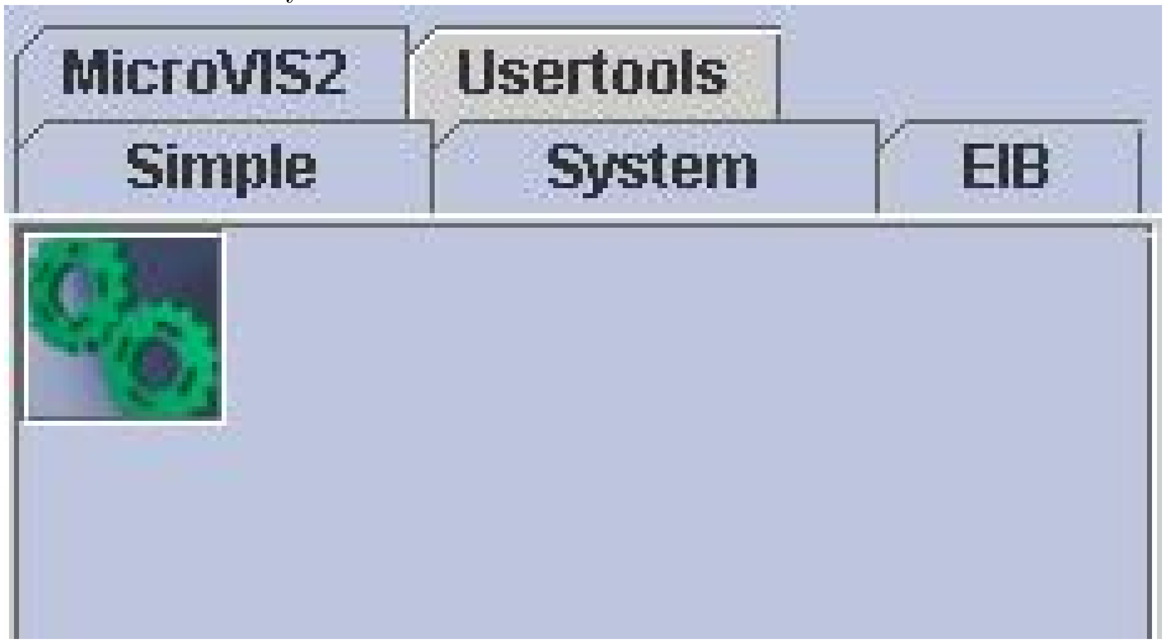
3.7.4 Userelemts

Userelements can be made from any element that is already configured by the user. As such it can be stored and recalled at any time without reconfiguration.
Creating a userelement:
By pressing the right mousebutton with the cursor pointing to an element, you get a pulldown menu.

There you can choose ‘new userelement’.



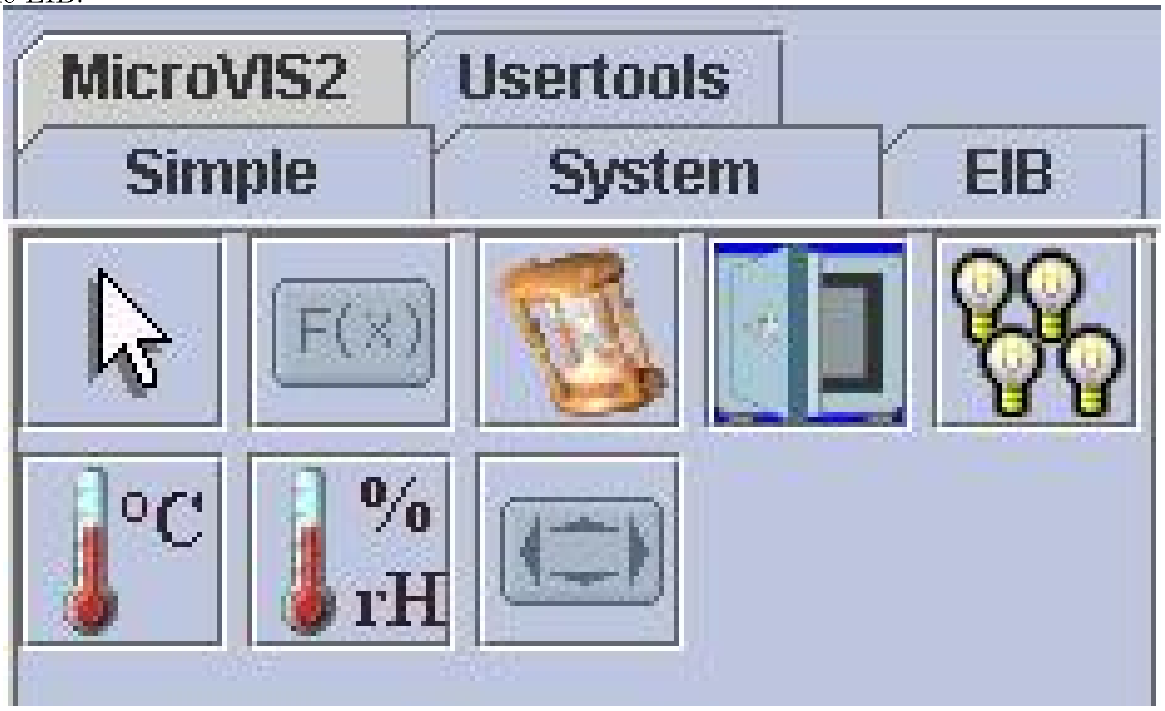
After defining a name and an icon, the new userelement is placed in the ‘**Userelements**’-Tabber with the icon that was defined by the user at the time of creation.



This element can be placed anywhere as usual and is a clone of the original element.
As this it can be used in any project.

4 The MicroVIS2-Logic Modul

The **MicroVIS2-Logic** is an extension to the **MicroVIS**. All elements and functions available in **MicroVIS** are available in **MicroVIS2-Logic** as well. **Additional Elements and Extensions:**
Up to 128 Groupaddresses
128KByte Memory for pages, images and fonts.
Freely programmable due to integration of the logical elements of the MicroFM Module with up to 48KByte of memory for code and data.
Userfunctions, Timers, Security PIN as well as **SceneManagement** and **Complex Controls** are available as additional Elements.
As an add-on for the newest generation of MicroVIS2 a separate **Temperature/Humidity**-sensor is available. The measured values of this sensor are available as values for the pages and can be sent to the EIB.



4.1 Introduction

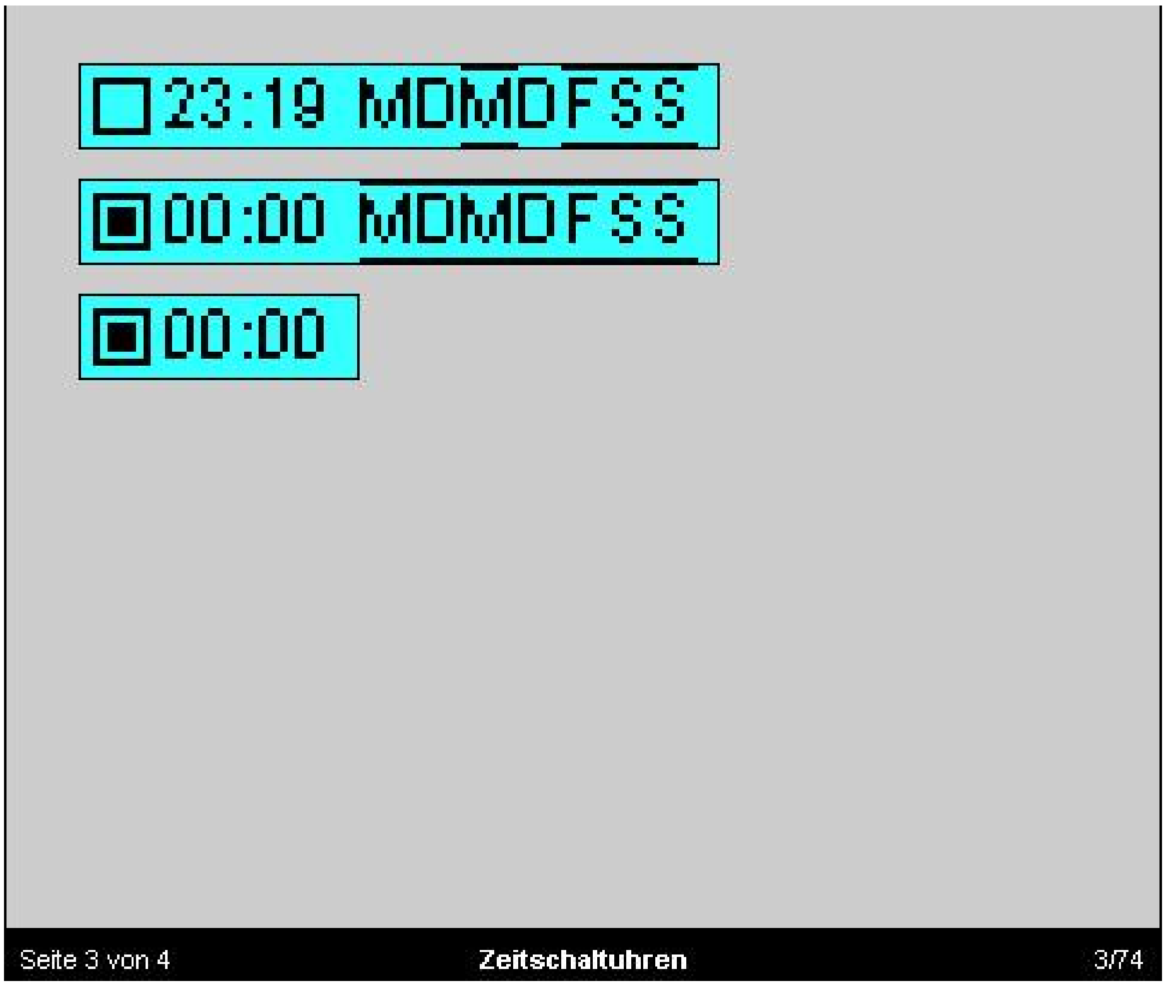
This document contains information such as data, illustrations, measurements and others which are subject to change without prior notice. Additional information is available at http:/www.arcus-eds.de

Technical changes preserved!

4.2 Userfuncitons

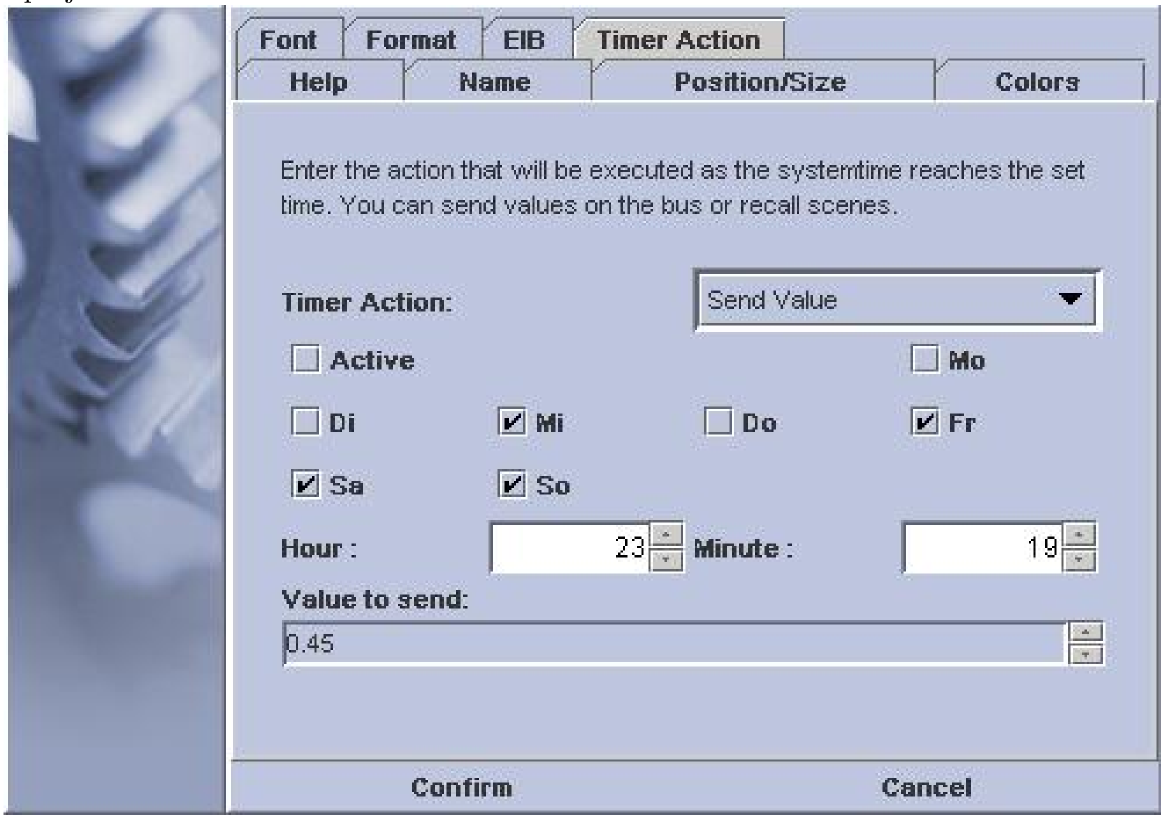
A Userfunction is a virtual element, which can be placed anywhere on any page in a project.
Under the name of this element there is a function created in the logic part of MicroVIS2-Logic.
This function can react on events like "Show the page", "Leave the Page", "Enter the Element" and "Turn the Knob".
The given Properties of the Element like Colours, Position, and Font are available to the Logic.
Without filling this Function with some executable code, this element simply does nothing.
About the Programming of the MicroVIS2-Logic there are additional informations soon available from our website www.arcus-eds.com.

4.3 Timer



Timers are elements that can automaticly do some actions like sending data to the bus or recalling scenes at user defineable times.
The settings "Active", "Execution Time" and "Weekdays" can be altered by the user but are predefined

in the project and are transfered to the device with it.



Enter the action that will be executed as the systemtime reaches the set time. You can send values on the bus or recall scenes.

Timer Action: Send Value

- [ ] Active
- [ ] Mo
- [ ] Di
- [x] Mi
- [ ] Do
- [x] Fr
- [x] Sa
- [x] So

Hour : 23 Minute : 19

Value to send:

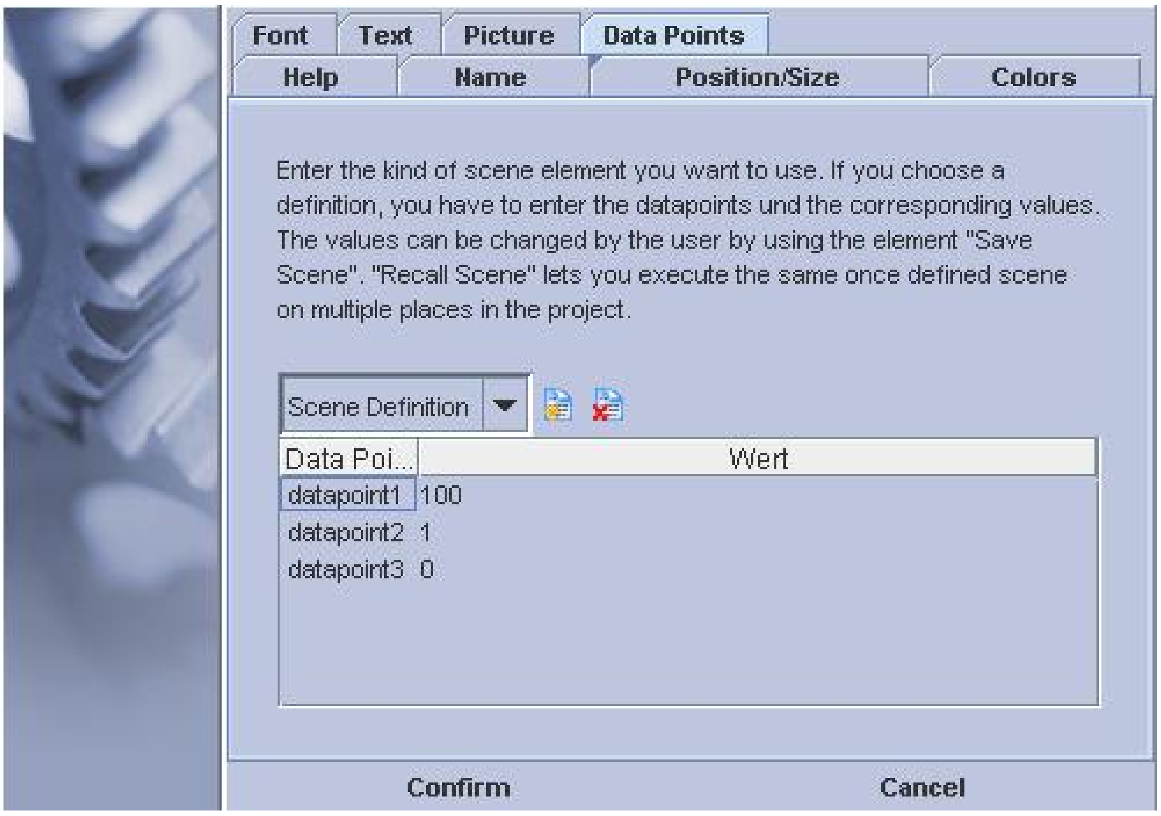
0.45

Timers are possible as daily timers or as weekly timers, this is set in the format dialog.
The value to send or the scene to execute are not to be changed by the user.
When sending values to the bus the maximum datawidth is 2 byte.
For the useage as weekly timer, the weekday setting in the systemtime must be set correctly.
Best results are obtained when getting the systemtime periodically from a timemaster on the EIB/KNX bus like the MicroFM-DCF77.

4.4 Security PIN

The Security PIN is an element on the page which can not be skipped without entering the correct PIN number.
Once entered, the pin is valid until the page is left again.
The PIN can range from 0 to 65535, it is defined in the project.
You can use as many different PIN elements on different (or the same) pages as you want.

4.5 Scene Management



Enter the kind of scene element you want to use. If you choose a definition, you have to enter the datapoints und the corresponding values. The values can be changed by the user by using the element "Save Scene". "Recall Scene" lets you execute the same once defined scene on multiple places in the project.

Scene Definition

Data Poi...	Wert
datapoint1	100
datapoint2	1
datapoint3	0

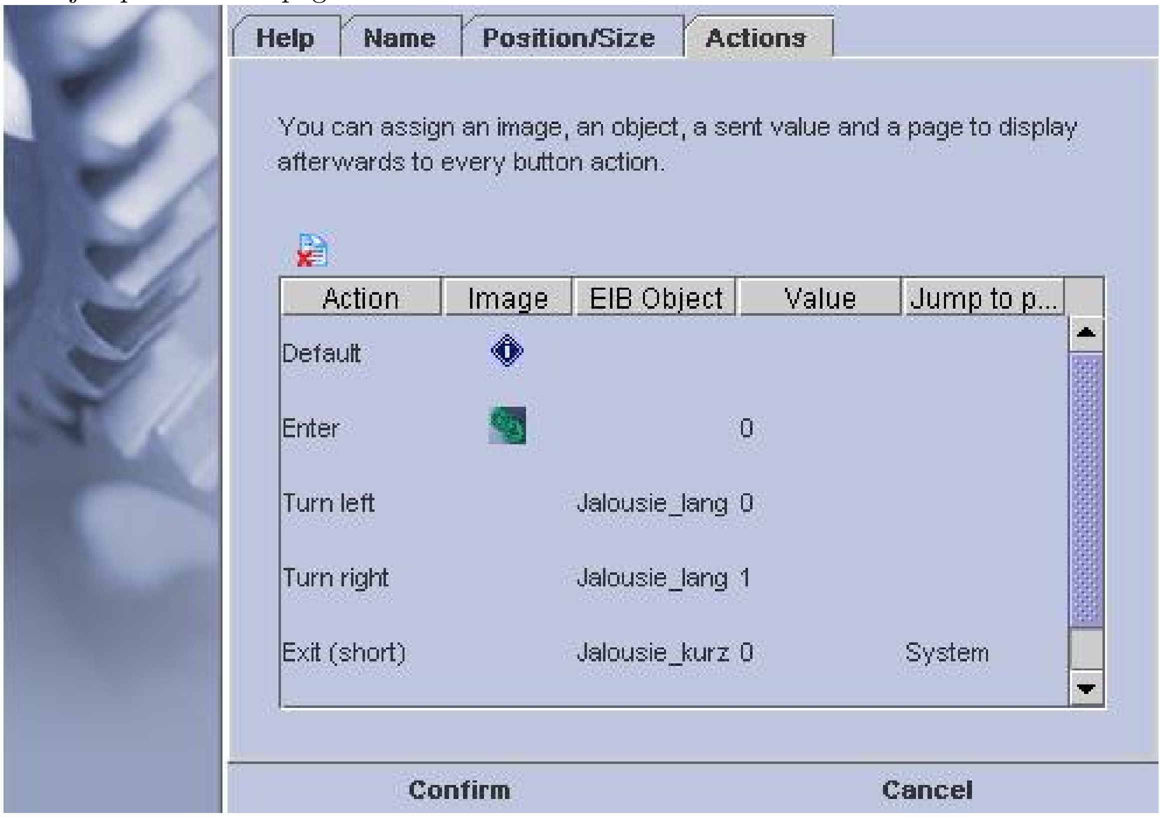
A Scene is a list of Datapoints and values for that datapoints. These values are sent to the EIB/KNX bus as the scene is recalled.
There are 3 types of scene elements:
Scene Definitions with Recall, Scene Recall only, and Scene Save.
A **Scene Difinition** defines the datapoints and the default values for that datapoints. Executing this element recalls the scene.
With **Scene Recall** you can execute the same scene on multiple pages in your project.
With **Scene Save** you can overwrite the predefined values of your scene with the actual values which were set over the EIB/KNX bus.
Scene elements can be shown as buttons with a text inside or as an image. The scene ist referenced by ist name property to other elements like timers etc.
When sending values to the bus the maximum datawidth is 2 byte.

4.6 Temperatur/Humidity

Temperature/Humidity: Temperature and Humidity-elements are available if the additional sensor element is connected to the extension connector of the MicroVIS2-Logic of the newest generation. Configuration is similar to the standard valueelements.
The values are available on the surface and can as well be sent to the EIB.

4.7 Complex Control

With complex controls you can create elements that do special tasks on specified events (enter , turn , leave , push-long). You can send specified values to different objects, change the appearance of the control or jump to defined pages.

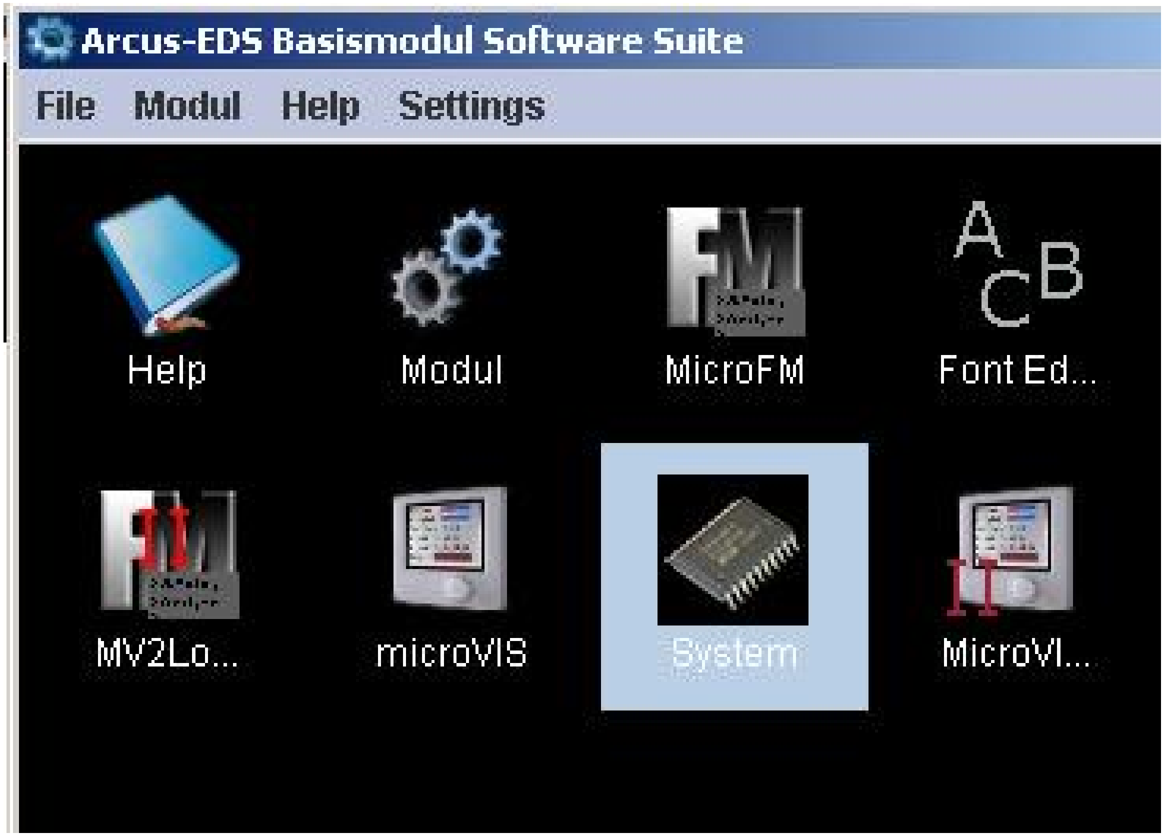


You can assign an image, an object, a sent value and a page to display afterwards to every button action.

Action	Image	EIB Object	Value	Jump to p...
Default	◆			
Enter	▨		0	
Turn left		Jalousie_lang	0	
Turn right		Jalousie_lang	1	
Exit (short)		Jalousie_kurz	0	System

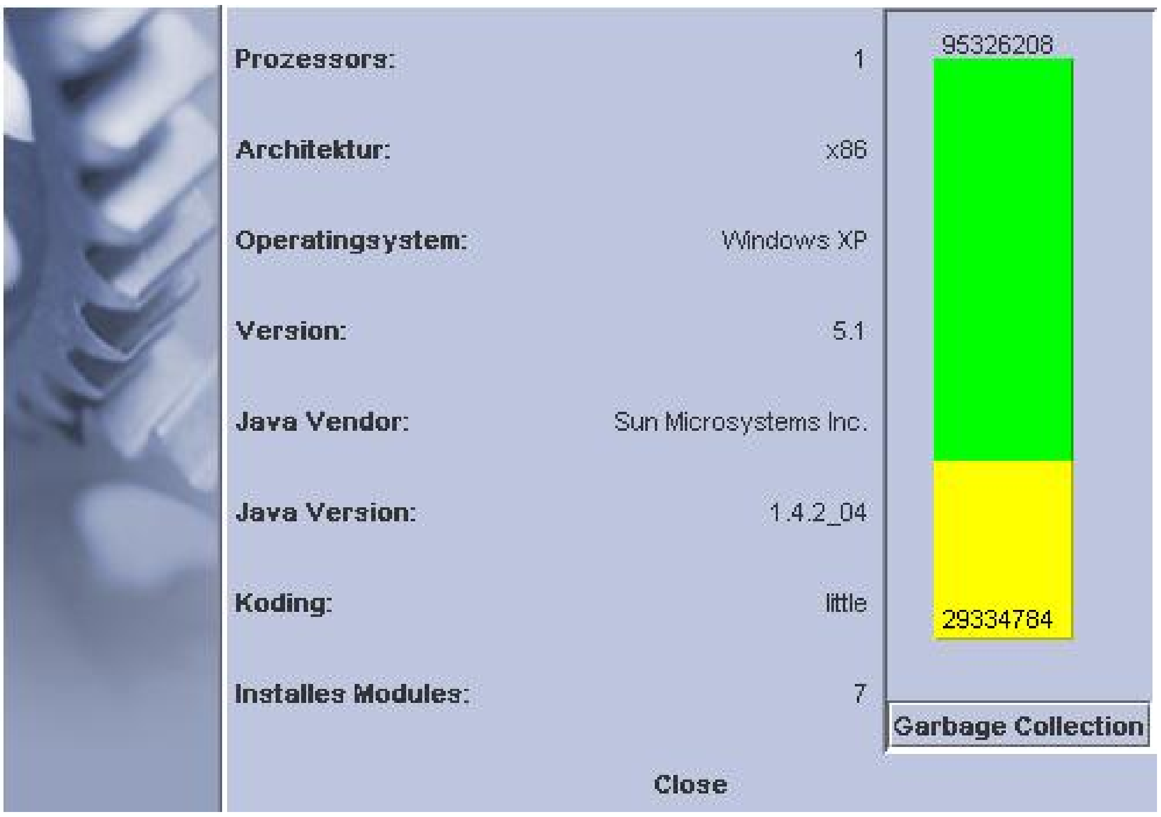
5 The System Module

5.1 Starting the Module



The System Modul shows you important software settings of your computer configuration helpful for support questions.

5.2 About this Version



Prozessors:	1
Architektur:	x86
Operatingsystem:	Windows XP
Version:	5.1
Java Vendor:	Sun Microsystems Inc.
Java Version:	1.4.2_04
Koding:	little
Installes Modules:	7

In the system module you can see important software-qualities of your computer configuration, which could be helpful for supporting questions.