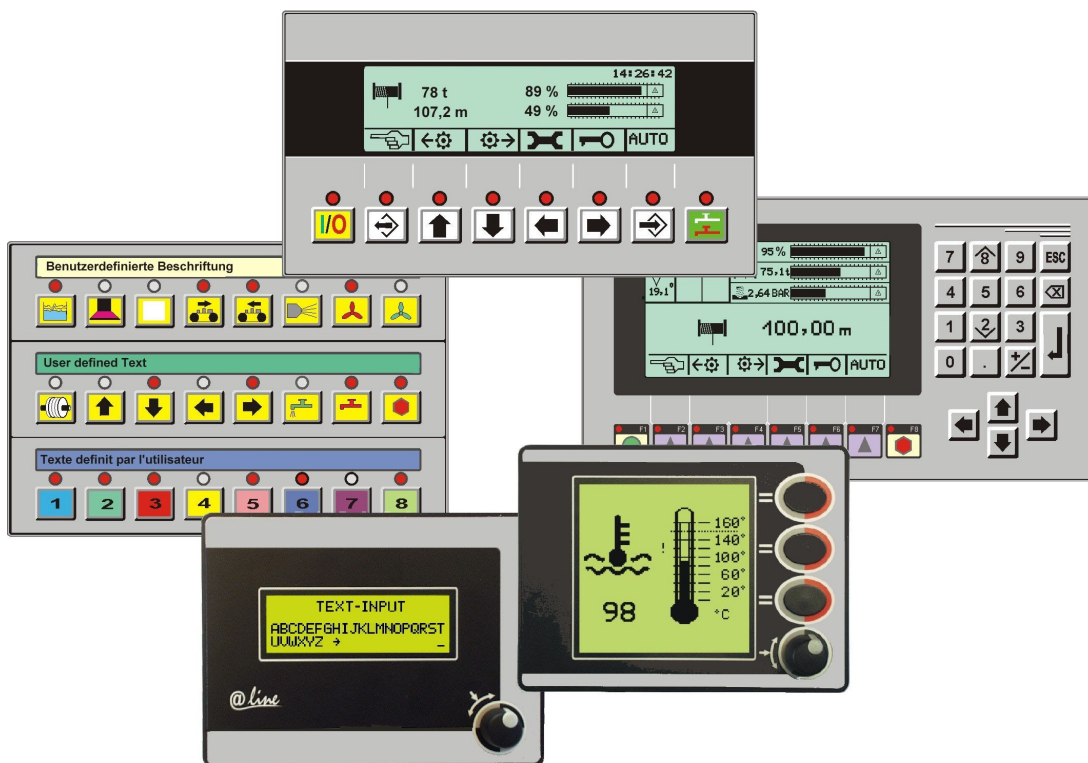




# GRAF-SYTECO

## Manual

### Controlling with Ladder Logic



Document: H128A0  
Status: released  
Issued: June 2003

**SY**steme **TE**chnischer **CO**mmunikation

# Operating panel manual

---

## 1 Introduction

### 1.1 Information on this manual

This manual describes the KOP.EXE program, which is used for developing simple, non-time critical control programs based on ladder diagrams.

### 1.2 Version information

This H003A06 manual applies to the ITE 6D18A00 version of the ladder diagram.

The operating panels must be equipped with the following program versions:

ITS6 series: BIOS:	as of IB054Sxx.HEX
TOS:	as of IO040Sxx.HEX
ITS7 series: BIOS:	as of IB155Sxx.HEX
TOS:	as of IO154Sxx.HEX

If you would like to create programs using the LD editor, you have to activate version D. For this purpose, plug in the copy protection module (dongle) when starting the ITE editor after installation. The freeware version C only offers demo programs.

### 1.3 Scope of services of the control program

The following list contains the various function features of the control program:

- *Initialisation part, cyclically-processed and time-controlled program part.*
- *Standard switching functions similar to IEC 1131-3.*
- *Flag field for intermediate results.*
- *Remanent flags.*
- *Non-remanent flags.*
- *Direct access to the CAN-I/O-level.*
- *Programming logic "AND", "OR", "NOT" operations.*
- *Programming page and message call-ups.*
- *Actuating LEDs and outputs.*
- *Requesting keys and inputs.*
- *Comparative operations for analogue parameters.*
- *Arithmetic operations for scaling functions.*
- *Direct CAN telegram functions.*

All function features can be generated and processed with the LD editor.

### 1.4 Basic mode of operation

After the operating panel has been switched on, it runs the initialisation part of the loaded control program.

Non-time-critical functions are controlled via the cyclic program part.

For functions with fast response times, a program part is available which can be set to a time-controlled operating mode. It runs within fixed, guaranteed intervals, which can be set in steps of 10 msec.

### 1.5 Process image

The control program generates a process image for digital and analogue inputs and outputs as well as the internal flag field. The operating panel always works with internal variables and system variables, which can be used for calculations, comparisons etc. within the control program.

Additionally, you have the possibility to directly use the inputs and outputs of the CAN modules with addresses 1-8. In the time-controlled program part, this "CAN-I/O-level" must be used, as otherwise the CAN bus cannot be operated fast enough. For the cyclic program part, both CAN-I/O level and variables may be used.

#### 1.5.1 Inputs process image

This process image depends on the currently active program part.

When the cyclic program part is running, variables are not externally modified, i.e. they are consistent.

On the other hand, the CAN-I/O-level of the 8 modules is reset whenever a module reports its status. During the time-controlled program part, the CAN-I/O-level is not modified and variables keep their value.

#### 1.5.2 Outputs process image

For the outputs process image as well, a differentiation must be made on the basis of the active program part.

During the cyclic program part, modified variables are written into an output buffer and only output to the respective devices when the cycle has been completed.


If you use the time-controlled program, you should not write onto CAN-I/O level outputs.

# Operating panel manual

## 2 The LD editor

The LD editor (KOP) is called up from the ITE project editor. Prior to calling up the LD editor, a project must be loaded or the currently processed project must be saved.

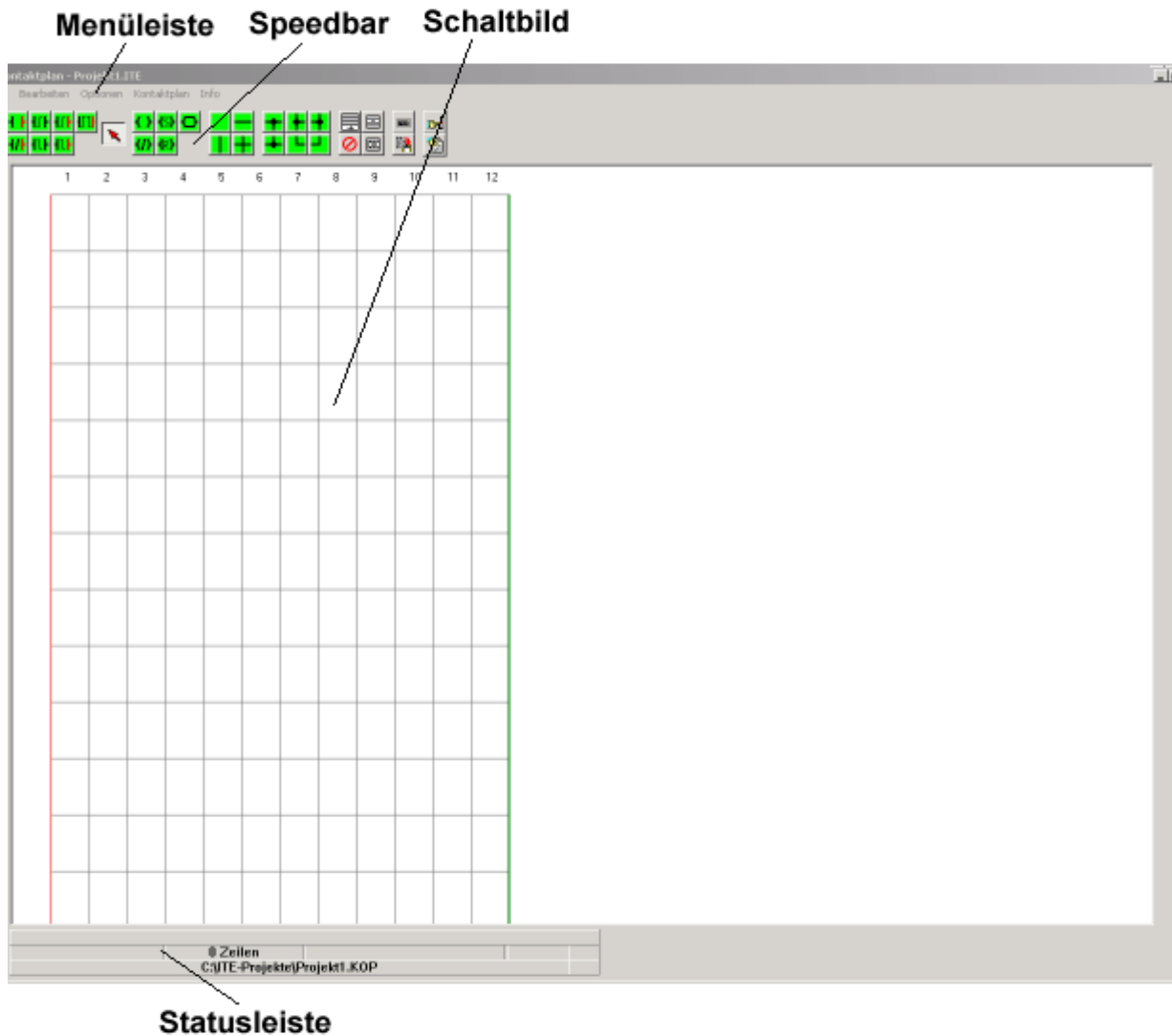
Via the Program>Edit ladder diagram... menu

items ... or the  button, the project editor is

closed and the LD editor activated. After the ladder

diagram has been edited, saved and compiled, the LD editor can be closed again. Then, the program returns to the project editor.

The main window of the LD editor is dominated by the "circuit diagram". Up to approximately 600 lines can be used for the creation of a ladder diagram. The window can be scrolled up and down by means of scroll bars. After the LD editor has been started with an empty project, the respective display looks as follows:



# Operating panel manual

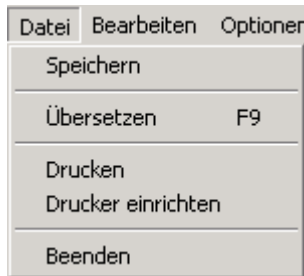
Most functions can be activated via a Windows-typical menu bar.

A speed bar, a tool bar as well as switching and wiring elements are available on top of the circuit diagram.

Buttons for frequently used functions are located on the left side of the circuit diagram.

## 2.1 Menu bar

### 2.1.1 "File" menu



#### 2.1.1.1 Save file

Saves the created ladder diagram to a file, which adopts the name of the ITE-project with the file extension .KOP. For example, if your project is called KRAN.ITE, the ladder diagram is automatically saved as KRAN.KOP.

#### 2.1.1.2 Compile

Converts the created ladder diagram into the format required for the operating panel. This function generates a file which has the same file name as the project. Depending on the file size, the file extension for ITS devices is ".HEX", ".HE3" or ".HE4". For AT devices, the file extension is ".ATX" or ".AT2" and ".AT3" respectively.

The LD editor indicates the compilation status in a window and reports possible compilation errors with the respective line and column. An error number is to help you to easily determine the error. Error codes: See "Compiling LD" on page 29.

#### 2.1.1.3 Print

Prints the ladder diagram on the standard printer configured under Windows.

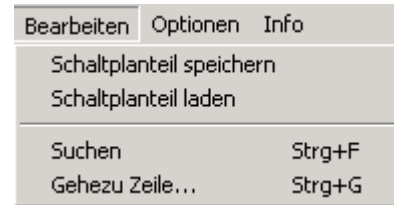
#### 2.1.1.4 Printer configuration

Opens the standard Windows dialogue for printer selection and configuration.

#### 2.1.1.5 Close

Closes the program after you have been asked whether the ladder diagram is to be saved and compiled.

### 2.1.2 "Edit" menu



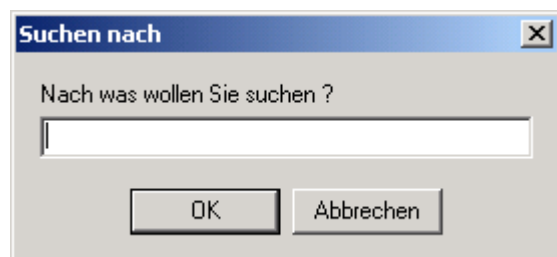
#### 2.1.2.1 Save block

Saves a part of the circuit diagram as block to a file. If this menu item is selected, the status line displays "Click on first line!". After clicking the first line to be saved, a dialogue shows up which asks you to enter a file name. Just enter the desired name; the .KOP file extension is automatically assigned. The program then asks you to click the end line. This way, the ladder diagram area between the start and the end line is saved to the file.

#### 2.1.2.2 Load block

Loads a stored circuit diagram part as block into the circuit diagram. If this menu item is selected, the status line displays "Click on destination line!". After clicking the destination line, a dialogue shows up where the file to be loaded can be selected. The dialogue only contains files with .KOP extensions. After the respective file has been selected, the circuit diagram block is inserted into the destination line and the following lines.

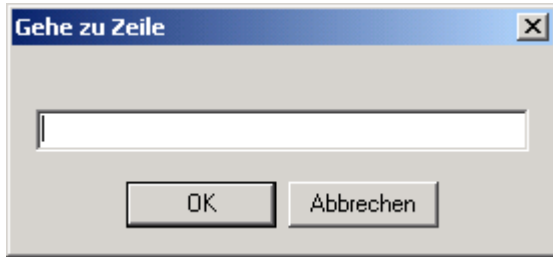
#### 2.1.2.3 Search



Enter the term you are looking for. The ladder diagram editor jumps from the current line position to the first position where the search term was found.

# Operating panel manual

## 2.1.2.4 Go to line



Jumps to the indicated line in the circuit diagram.

## 2.1.3 "Options" menu

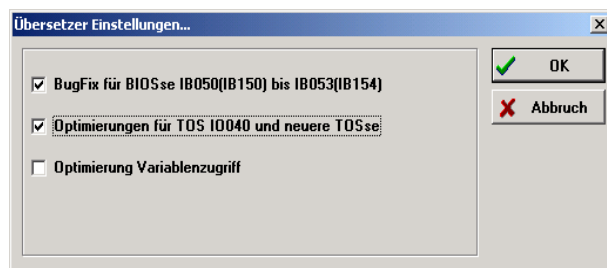


### 2.1.3.1 LD font

Select the fonts for switch designations, lines and column numbering. All switches and relays are displayed in the same font.

### 2.1.3.2 Compiler settings...

Opens a dialogue for the compilation settings.



### BugFix for BIOSse...

Provides for a trouble-free execution of the LD editor even with BIOS versions IB050 to IB053 and IB150 to IB153. This setting has no negative impact on other BIOS versions and should therefore remain active.

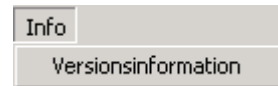
### "Optimization for TOS IO040 or newer one" box

This box must also remain active. Only deactivate this option when using a TOS version older than IO040.

## Optimize variables access

This option should only be ticked if the ladder diagram overstrains the operating panel. This is the case when pages are built up very slowly. Please observe the following when activating this setting: If you modify the table of variables (in the editor, LD or CAN configurator), you must re-compile the circuit-diagram before transferring the project to the operating panel. Otherwise, variables may not be properly accessed.

## 2.1.4 "Info" menu



The "Version information" menu item opens an external program which displays contact information and the version of the individual programs of the software package. In cases of software problems, please contact our customer support and state the version information contained in this window.

## 2.2 Tool bar

The tool bar contains buttons for inserting switch elements,



relays,



and wiring elements



as well as some editing functions.



A more detailed description of switch, relay and wiring elements is available in chapter 4.

# Operating panel manual

## 2.2.1 Frequently used functions



### Edit cyclic program part

Displays the section of the cyclically running program part in the ladder diagram.



### Edit initialisation part

Displays the initialisation part of the control program. This part is executed one time after the operating panel has been switched on or reset.



### Edit time-controlled program part

Displays the time-controlled program part in the ladder diagram.



### Translate ladder diagram



### Open file

Opens an existing ladder diagram file. This button is not available if the LD editor was called up from the ITE.



### Save file

Saves the currently processed ladder diagram.



### Insert LD block

Inserts a block saved as .KOP into the circuit diagram.



### Save LD block

Saves a part of the circuit diagram as block.



### Print ladder diagram

Prints the currently active ladder diagram on the standard Windows printer.



### Configure printer

Calls up the standard Windows dialogue for printer configuration.

## 2.3 Ladder diagram

The ladder diagram is structured like a tile wall which is divided into lines and columns. On each tile, a switch, relay or wiring element can be positioned.

The initialisation part, the time-controlled part as well as the cyclic part can comprise a total of approx. 600 lines. The lines are shared dynamically. With the current LD version, the width - 12 switch elements per line - cannot be modified.

If the window does not have sufficient space for all lines, scroll bars are automatically attached to the circuit diagram. This way, you always have access to the ladder diagram area to be edited.

## 2.4 Status line

Relaisfunktion		Zykluszeit zeitgesteuerter Teil	
1 Zeilen	Keine Fehler im Schaltplan	500 ms	
C:\ITE-Projekt\Projekt1.KOP			
Änderungsstatus	Kontaktplanlänge	Dateiname mit Pfad	Fehlerstatus

The status line contains a number of information which are helpful when working with the LD editor:

- The first line always shows the function of the switch element on which the mouse is positioned. This is particularly helpful if the function name is very long and partially covered by the following switch element.
- The first field of the second line (left) informs you, whether you have modified the circuit diagram.
- The second field displays the number of lines of the currently edited program part.
- The third field informs you about possible errors after the circuit-diagram has been compiled. An additional text window lists the compilation results in detail.
- The fourth field is only active if you edit the time-controlled program part. It displays the time interval you have adjusted for the time-controlled program. If you double-click this field, you can set the interval.
- The last line shows the file name of the currently edited ladder diagram with the respective drive and path.

# Operating panel manual

## 3 Switch, relay and wiring elements

Three types of ladder diagram elements are available for the operating panel: switches, relays and lines.

### 3.1 Switch types

Switches are used to only control downstream relays if certain conditions are fulfilled. By connecting switch elements in series or in parallel, a connection logic can be established (also refer to chapter 4). The connection program always runs along the lines from the left to the right and from the top to the bottom unless a switch element with cancel is selected. In this case, the connections located further right of the switch elements and relays are not realised if a condition is not fulfilled.

#### 3.1.1 Switch elements without cancel

As regards switch elements without cancel, the switch program at the right of the switch element is executed first, no matter whether the condition is fulfilled or not. Only then is the next line processed.



##### 3.1.1.1 Closing switch

Via this button, a closing switch can be inserted into the ladder diagram. The closing switch "carries current" if the condition you have entered in the parameterisation mask is fulfilled.

```
CODE IF
      SWITCH_CLOSED
      AND
      CONDITION_FULFILLED
      THEN
      CARRYING_CURRENT
      OTHERWISE
      INTERRUPTING_CURRENT
```

**Continue switch program to the right**  
**Continue switch program downwards**



##### 3.1.1.2 Opening switch

Via this button, an opening switch can be inserted into the ladder diagram. The opening switch "carries current" if the condition you have entered in the parameterisation mask is fulfilled.

```
CODE IF
      SWITCH_OPEN
      AND
      CONDITION_NOT_FULFILLED
      THEN
      CARRYING_CURRENT
      OTHERWISE
      INTERRUPTING_CURRENT
```

**Continue switch program to the right**  
**Continue switch program downwards**



##### 3.1.1.3 Positive edge closing switch

Via this button, a closing switch for positive edges can be inserted into the ladder diagram.

The "positive edge closing switch" starts to carry current exactly when the condition you entered in the parameterisation mask changes from "not fulfilled" to "fulfilled".

```
CODE IF
      SWITCH_CLOSED
      AND
      CONDITION_CHANGES_FROM
      NOT_FULFILLED_TO_FULFILLED
      THEN
      CARRYING_CURRENT
      OTHERWISE
      INTERRUPTING_CURRENT
```

**Continue switch program to the right**  
**Continue switch program downwards**



##### 3.1.1.4 Negative edge closing switch

Via this button, a closing switch for negative edges can be inserted into the ladder diagram.

The "negative edge closing switch" starts to carry current exactly when the condition you entered in the parameterisation mask changes from "fulfilled" to "not fulfilled".

```
CODE IF
      SWITCH_CLOSED
      AND
      CONDITION_CHANGES_FROM
      FULFILLED_TO_NOT_FULFILLED
      THEN
      CARRYING_CURRENT
      OTHERWISE
      INTERRUPTING_CURRENT
```

**Continue switch program to the right**  
**Continue switch program down-**  
**wards**

### 3.1.2 Elements with cancel

When setting switch elements with cancel, the next line is processed if the condition is not fulfilled. The switch program at the right of the switch is not executed. Instead the branch at the left of the switch is processed.

Switch types with cancel are preferably used to optimise the LD program runtime. If, for example, the individual flags of a flag word are used for message call-ups, the flags are always called up individually. If, however, a switch with cancel is set in front of the "message flag circuit", the downstream circuit is only processed if the switch with cancel is operated. In such cases, a reasonable call-up such as "If flag word has changed" is used. The number of call-ups is reduced from 16 to 1 if no change has taken place.



#### 3.1.2.1 Closing switch with cancel

Via this button, a closing switch with cancel can be inserted into the ladder diagram. The closing switch with cancel "carries current" if the condition you have entered in the parameterisation mask is fulfilled.

```
CODE IF
      SWITCH_CLOSED
      AND
      CONDITION_FULFILLED
      THEN
      CARRYING_CURRENT
```

**Continue switch program to the right**  
**Continue switch program down-**  
**wards**



#### 3.1.2.2 Opening switch with cancel

Via this button, an opening switch with cancel can be inserted into the ladder diagram.

The opening switch "carries current" if the condition you have entered in the parameterisation mask is fulfilled.

```
CODE IF
      SWITCH_OPEN
      AND
      CONDITION_NOT_FULFILLED
      THEN
      CARRYING_CURRENT
```

**Continue switch program to the right**  
**Continue switch program down-**  
**wards**



#### 3.1.2.3 Positive edge closing switch with cancel

Via this button, a closing switch for positive edges with cancel can be inserted into the ladder diagram.

The "positive edge closing switch with cancel" starts to carry current exactly when the condition you entered in the parameterisation mask changes from "not fulfilled" to "fulfilled".

```
CODE IF
      SWITCH_CLOSED
      AND
      CONDITION_CHANGES_FROM
      NOT_FULFILLED_TO_FULFILLED
      THEN
      CARRYING_CURRENT
```

**Continue switch program to the right**  
**Continue switch program down-**  
**wards**



#### 3.1.2.4 Negative edge closing switch with cancel

Via this button, a closing switch for negative edges with cancel can be inserted into the ladder diagram.

The "negative edge closing switch with cancel" starts to carry current exactly when the condition you entered in the parameterisation mask changes from "fulfilled" to "not fulfilled".

```

IF
  SWITCH_CLOSED
AND
  CONDITION_CHANGES_FROM
  FULFILLED_TO_NOT_FULFILLED
THEN
  CARRYING_CURRENT
  
```

Continue switch program to the right  
Continue switch program down-wards



### 3.1.2.5 Both edges closing switch with cancel

Via this button, a closing switch for both edges with cancel can be inserted into the ladder diagram.

The "both edges closing switch with cancel" starts to carry current exactly when the condition you entered in the parameterisation mask changes from "fulfilled" to "not fulfilled" or from "not fulfilled" to "fulfilled".

```

CODE  IF
      SWITCH_CLOSED
      AND
      CONDITION_CHANGES_FROM
      FULFILLED_TO_NOT_FULFILLED
      THEN
      CARRYING_CURRENT
      Continue switch program to the right
      OR
      CONDITION_CHANGES_FROM
      NOT_FULFILLED_TO_FULFILLED
      THEN
      CARRYING_CURRENT
  
```

Continue switch program to the right  
Continue switch program down-wards

## 3.2 Relay types

The functions of the operating panel can be controlled by means of relays. Depending on the relay type, you can trigger functions in the closed or/and open state. If the red arrow symbol is active, the relay functions can be adjusted via a double-click on the respective relay.

### 3.2.1 Standard relays



Via this button, a standard relay can be inserted into the ladder diagram.

#### Operating mode of standard relays

The operating mode of standard relays depends on your parameterisation. If you parameterise a variable setting / calculation, you can carry out different calculations for "relay open" and "relay closed". Please refer to the description of the relay parameterisation mask.

If you parameterise other functions, e.g. "page call-up", the relay calls up the respective page if it is closed and removes the page if it re-opens.

#### Application example:

You use a closing switch and formulate, for example, an "IsSpeed>60" condition. A standard relay is to call up a message for status indication. This way, you only need 2 switch elements for both the message call-up and the message output. Isn't this practical?

If the standard relay was represented as program, the following equations would result:

- For set variable/calculate or set timer (upper field of the "Parametrize relay" mask):  
if RELAY\_CLOSED,  
set\_value (from the "relay closed" field)  
otherwise set\_value (from the "relay open" field)
- For CAN functions (lower field of the "Parametrize relay" mask) and the functions in the third column of the centre field:  
if RELAY\_CLOSED,  
execute\_function
- For the functions of columns 1 and 2 in the centre field of the "Parametrize relay" mask:  
if RELAY\_CLOSED,  
execute\_function  
otherwise execute\_reverse\_function

The "reverse" function is usually available in the respectively other column (1 <--> 2) in the same line, e.g. the "switch off LED" function for "switch on LED".

# Operating panel manual

## 3.2.2 Inverted relays



Via this button, an inverted relay can be inserted into the ladder diagram.

### Operating mode of inverted relays

Inverted relay functions are basically characterised by the same operating mode as standard relays, only "open" and "closed" are exchanged.

If the inverted relay was represented as program, the following equations would result:

- For set variable/calculate or set timer (upper field of the "Parametrize relay" mask):  
if RELAY\_OPEN,  
set\_value (from the "relay closed" field)  
otherwise set\_value (from the "relay open" field)
- For CAN functions (lower field of the "Parametrize relay" field) and the functions in the third column of the centre field:  
if RELAY\_OPEN,  
execute\_function
- For the functions of columns 1 and 2 in the centre field of the "Parametrize relay" mask:  
if RELAY\_OPEN,  
execute\_function  
otherwise execute\_reverse\_function

The "reverse" function is usually available in the respectively other column (1 <--> 2) in the same line, e.g. the "switch off LED" function for "switch on LED".

## 3.2.3 Hold relays



Via this button, a hold relay can be inserted into the ladder diagram.

### Operating mode of hold relays

This relay is "part" of the standard relay. In the closed state, it only executes the function which is executed by the standard relay in the "relay closed" state.

As a consequence, this relay only executes the "relay closed" path if variables are to be set. "Relay open" is not applicable for this relay.

As regards message call-ups and similar functions, the message is called up but not output.

If the hold relay was represented as program, the following equations would result:

- For set variable/calculate or set timer (upper field of the "Parametrize relay" mask):  
if RELAY\_CLOSED,  
set\_value (from the "relay closed" field)

- For CAN functions (lower field of the "Parametrize relay" mask) and the functions in the third column of the centre field:  
if RELAY\_CLOSED,  
execute\_function
- For the functions of columns 1 and 2 in the centre field of the "Parametrize relay" mask:  
if RELAY\_CLOSED,  
execute\_function

## 3.2.4 Reset relays



Via this button, a reset relay can be inserted into the ladder diagram.

### Operating mode of reset relays

If the relay is open, the function is carried out as a "resetting" function. Example: If you indicate "activate message" as action, the "deactivate message" function is carried out.

If the reset relay was represented as program, the following equations would result:

- With set variable/calculate or set timer (upper field of the "Parametrize relay" mask):  
if RELAY\_OPEN,  
set\_value (from the "relay open" field)
- For CAN functions (lower field of the "Parametrize relay" mask) and the functions in the third column of the centre field:  
\*\*\*\* Relay is not applicable\*\*\*\*
- For the functions of columns 1 and 2 in the centre field of the "Parametrize relay" mask:  
if RELAY\_OPEN,  
execute\_reverse\_function

The "reverse" function is usually available in the respectively other column (1 <--> 2) in the same line, e.g. the "switch off LED" function for "switch on LED".

## 3.2.5 Function relays



Via this button, a function relay can be inserted into the ladder diagram.

### Operating mode of function relays

Function relays operate like "hold relays" with edge evaluation. This function is executed if a positive edge is recognised at the relay.

If the function relay was represented as program, the following equations would result:

- For set variable/calculate or set timer (upper field of the "Parametrize relay" mask):  
if POSITIVE\_EDGE\_CURRENT,  
set\_value (from the "relay closed" field)

# Operating panel manual

- For CAN functions (lower field of the "Parametrize relay" mask) and the functions in the third column of the centre field:  
if `POSITIVE_EDGE_CURRENT`,  
`execute_function`
- For the functions of columns 1 and 2 in the centre field of the "Parametrize relay" mask):  
if `POSITIVE_EDGE_CURRENT`,  
`execute_function`

## 3.3 Connection elements

Connection elements are self-explanatory. By means of these elements, you can establish connections between switch elements.



This element creates an empty tile and deletes already existing elements.



### Connection lines

These elements need not be explained. They are just "lines".



### Crossing element

This crossing element represents a non-conducting crossing of two lines.



### Branches

The point symbolises that lines are connected to each other.



### 3.3.1 Insert connection elements

If you click one of the connection element buttons, the respective button remains down and symbolises that this connection element can now be inserted into the ladder diagram. Click the mouse onto the ladder diagram position where the element is to be inserted. As opposed to switch elements, the element button remains down, i.e. you may insert several connecting elements of the same type by simply clicking the ladder diagram.

### 3.3.2 Overwrite connection elements

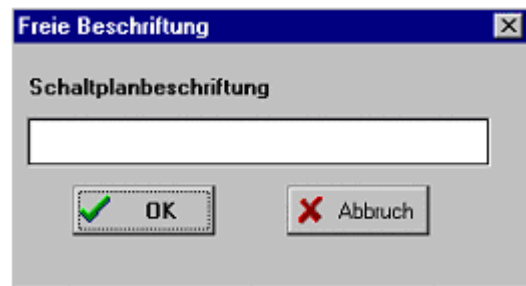
Select the button of the desired new element and click the ladder diagram element you want to overwrite.

### 3.3.3 Label connection elements

If you want to add comments to the ladder diagram for documentation purposes, the ladder diagram allows you to label wiring elements or empty fields. Please proceed as follows:

1. Select the "arrow" tool in the tool bar.
2. Double-click the element you want to label.

The following input mask shows up:



3. Enter the desired label text in the entry line.
4. Press OK and the labelling is taken over into the ladder diagram.

### 3.3.4 Delete element labelling

In cases element labellings are to be deleted, please proceed as described under 3.3.3 and delete the entire text in the input mask.

### 3.3.5 Insert relay/switch

If you press one of the switch element buttons, the respective button remains down and symbolises that this switch element can now be inserted into the ladder diagram. Click the mouse onto the ladder diagram position where the element is to be inserted. As opposed to connection elements, the switch/relay button does not remain down, instead the arrow is selected. This way, you can immediately call up the parameterisation mask via a double-click.

### 3.3.6 Replace relay/switch

Select the button of the desired new element and click the element in the ladder diagram you want to overwrite. With switches, the request remains valid; with relays, the action as well. Please check the parameterisation in any case!

### 3.3.7 Delete elements

Select the empty button from the connection elements and click the element to be deleted in the ladder diagram.

# Operating panel manual

## 4 LD variables

Apart from project variables, the ladder diagram editor also provides flags, which can be used as result buffer. In addition to this, direct access to the inputs and outputs of the CAN modules is possible with node addresses 1-8.

If you want to access these variables, you have to "manually" enter them into the parameterisation masks of the switches or relays.

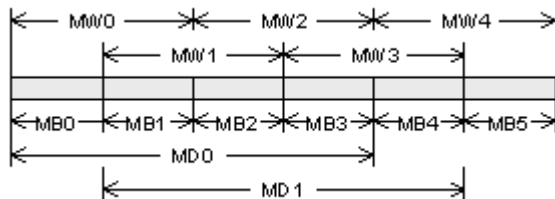
The third variable type are timers. The editor can work with up to 10 timers which allow for a realisation of switch-on or switch-off delays.

### 4.1 Flag field

The flag field of the LD contains 128 flag bytes. These flags can be accessed by entering the flag name into the variable fields. You have the following options:

<b>M0.0 to M127.7</b>	flag bit
<b>MB0 to MB127</b>	flag byte
<b>MW0 to MW126</b>	flag word
<b>MD0 to MD124</b>	flag double word

The flag area is characterised by an overlapping structure:



With MW/MD, the MB with the lowest number also has the lowest rating.

The flag bytes 0-63 are deleted with every device re-start (set to 0). The flag bytes 64-127 keep their value if a battery has been integrated (device with real-time clock).

#### 4.1.1 Standard addressing of the flag field

The flags must begin with an "@", which is followed by the flag type and the number. Examples:

<b>@M3.5</b>	flags 3.5
<b>@MW2</b>	flag word 2
<b>@MB7</b>	flag byte 7

Enter this name into the "variable" field in the respective parameterisation masks.

#### 4.1.2 Indirect addressing of the flag field

Sometimes, if several measured values are to be subsequently read into a flag field, the flag word/byte should be addressed via a counter as with an

index.

Exactly this can be done here. This procedure, the indirect addressing only works with flag bytes, flag words and flag double words. The index or number of the flag byte is stored in the flag byte which is then used for processing. For example, you enter:

**@MB[10]**

which means: Take the flag byte whose number is stored in MB10.

Consequently, if flag byte 10 contained the value 12, the variable @MB[10] would address @MB12.

### 4.2 CAN and internal I/O level

The editor has direct I/O access to the first 8 CAN modules, i.e. you receive additional variables which represent the inputs and outputs of the CAN modules.

The editor does not (yet) know your CAN module configuration and can therefore not check your entries.

The following "I/O variables" are available:

<b>@DI0.0 to @DI0.15</b>	internal digital inputs
<b>@DO0.0 to @DO0.15</b>	internal digital outputs
<b>@AI0.0 to @AI0.1</b>	internal analogue inputs
<b>@DI1.0 to @DI8.55</b>	digital inputs GCM
<b>@DO1.0 to @DO8.31</b>	digital outputs GCM
<b>@AI1.0 to @AI8.3</b>	analogue inputs GCM
<b>@AO1.0 to @AO8.3</b>	analogue outputs GCM

Please consider that the analogue inputs and outputs are not scaled and treated as 12-bit values. The "real" values of the CAN modules are provided.

Enter an "@" in front of the I/O-name in the parameterisation masks, e.g.

**@DI3.6** digital input No. 6 at the module with address 3

**@DO2.13** digital output No. 13 at the module with address 2

**@AI2.1** analogue input No. 1 at the module with address 2

**@AO7.0** analogue output No. 0 at the module with address 7

The editor only checks the limits for module numbers and input/output numbers.

# Operating panel manual

## 4.3 Timers

The editor offers a total of 10 timers for special time functions.

These timers operate on the basis of a 100 msec step range. Internally, the LD works with a step range of 20 msec, which results in an inaccuracy of 20 msec related to the entire timer time.

Timers can be set by relays. Within this context, you must enter how many 100 msec steps are to be executed.

Switches can be used to check whether a timer still runs or whether it has already expired.

The timers always count down from the set value to 0. As soon as 0 has been reached, they are in the "stop" status.

The timers are directly available in the parameterisation masks. You just need to enter the respective number (0-9).

What is the difference between an hour counter and a timer?

Quite easy: An hour counter is a variable which must be created (in the table of variables). Hour counters operate in a second-based mode.

A timer cannot be displayed. It exclusively serves for the realisation of time-dependent switching functions in the LD.

## 4.4 System variables

What are system variables? Roughly spoken: The operating system (TOS) of the operating panel must internally save certain values, e.g. the currently displayed page or the number of active messages.

Many of these information are also available to the LD via system variables.

Some system variables only provide for a read-only access, i.e., the cannot modify the value of these system variables.

The variables are currently known:

Variable	Function	Access
SYS(Page No)	Indicates the currently displayed page.	RO
SYS(Message No)	Indicates the currently displayed message 0=no message	RO
SYS(Status)	Indicates the current device status. Refer to CAN telegram 0x0A, "operating panel status"	RO

SYS(Time zone)	Setting of the time zone: 0=winter 1=summer 2=no active time zone Can be modified by means of the CAN telegram. D0=0x15 (PARAM) D1=0x0D D2=value Can be realised via /relay CAN-function	RO
SYS(Menu index)	Indicates the menu index which has been delivered via the "menu index to LD" function. Only valid for 1 cycle and deleted afterwards (0).	RO
SYS(CAN_TRANSMIT_ID)	Transmission identifier in the final format.	RW
SYS(CAN_RECEIVE_ID_0)	Standard receive identifier in the final format.	RW
SYS(CAN_RECEIVE_ID_1)	Multi-master receive identifier 1.	RW
SYS(CAN_RECEIVE_ID_2)	Multi-master receive identifier 2.	RW
SYS(CAN_RECEIVE_ID_3)	Multi-master receive identifier 3.	RW
SYS(CAN_RECEIVE_ID_4)	Multi-master receive identifier 4.	RW
SYS(CAN_RECEIVE_ID_5)	Multi-master receive identifier 5.	RW
SYS(CAN_RECEIVE_ID_6)	Multi-master receive identifier 6.	RW
SYS(CAN_RECEIVE_ID_7)	Multi-master receive identifier 7.	RW
Final format for multi-master IDs: 15.....0 bit No.      --- 0x08 = master channel -- 0x08 = master channel       0x0F = channel       others = invalid !      +---- = RTR-bit (fixed to 0) +++++----- identifier (11-bit)		
SYS(CAN_UNIT)	SELECAN device address	RW
SYS(GCM_MASTER_ENABLE)	0=no master 1=GCM master	RO
SYS(SOFTKEY_MASK)	Soft key mask in terms of bits	RW
SYS(EXT_KEY_COUNT)	Number of keyboard extensions	RO
SYS(PLC_DRIVER)	Active PLC driver: 0=no driver 1=Siemens S5/PG 2=Mitsubishi FX 3=Request/Response 4=Telegram driver 5=VT100	RO
SYS(PAGE_ALTERNATING TIME)	Page alternating time in seconds	RW
SYS(MESSAGE_ALTERNATING TIME)	Message alternating time in seconds	RW
SYS(PRINTER_IF)	Printer connection 0=no printer 1=serial 2=to CAN-module	RW
SYS(CONTRAST)	Contrast Value 0 - 23	RO
SYS(BRIGHTNESS)	Brightness. Value 0 - 7	RO
Contrast and brightness can be set via the CAN PARAM function: D0 = (0x15) D1 = (0x01 = contrast; 0x02 = brightness) D2 = brightness level/contrast value Can be realised via /relay "CAN function"		

# Operating panel manual

SYS(RS232_STATUS)	If <0, a character was serially received. If the character has been read out, this system variable must be reset to 0.	RW
SYS(RS232_BUFFER)	Contains the received character.	RO
SYS(DIB0)	Inputs 0-7 of the I/O extension as byte	RO
SYS(DIB1)	Inputs 8-15 of the I/O extension as byte	RO
SYS(DIW0)	Inputs 0-15 of the I/O extension as byte	RO
SYS(DOB0)	Outputs 0-7 of the I/O extension as byte	RO
SYS(DOB1)	Outputs 8-15 of the I/O extension as byte	RO
SYS(DOW0)	Outputs 0-15 of the I/O extension as word	RO
SYS(X_Down)	Touchscreen, X coordinate when pressing	RO
SYS(Y_Down)	Touchscreen, Y coordinate when pressing	RO
SYS(X_Move)	Touchscreen, X coordinate when pushing	RO
SYS(Y_Move)	Touchscreen, Y coordinate when pushing	RO
X coordinates:0-239,255=invalid (not pressed) Y coordinates:0-127,255=invalid (not pressed)		
<b>Evaluation only after RESET:</b>		
SYS(CAN_BUSRATE)	CAN bus rate. 0 = 10 kBit/sec 1 = 20 kBit/sec 2 = 50 kBit/sec 3 = 100 kBit/sec 4 = 125 kBit/sec 5 = 250 kBit/sec 6 = 500 kBit/sec 7 = 1000 kBit/sec	RW
SYS(RS232_PARAM)	Parameter interface in terms of bits	RW
The bits indicated with X are freely selectable, the permanent bits have to be set in the predefined way. 7 0 bit 00XXXX01         *---- 0 = 1 stop bit     1 = 2 stop bits   +---- 0 = 7 data bits    1 = 8 data bits  +----- 0 = odd parity (uneven)   1 = even parity (even) +----- 0 = parity disable (no parity) 1 = parity enable		
SYS(RS232_BAUD)	Serial interface baud rate 0 = 9600 bauds 1 = 4800 bauds 2 = 2400 bauds 3 = 1200 bauds 4 = 600 bauds 5 = 300 bauds 6 = 150 bauds 7 = 110 bauds	RW
SYS(INIT_PARAMS)	Device switch-on behaviour; in terms of bits	RW
7.....0 bit XXXx0XXX      +---- delete internal variable      +---- variable 0-19 resident into FLASH      +---- skip publicity page     +----- always 0    +----- not used   +----- 0 to TOS IO031...    1 from TOS IO032... ++----- 00 display variable by 00000 01 display variable by ***** 10 display variable by BLANK		

SYS(CAN_ACCEPTANCE_CODE)	Identifier filter. Refer to the 82C200 or SJA1000 controller manual.	RW
SYS(CAN_ACCEPTANCE_MASK)	Identifier filter mask. Refer to the 82C200 or SJA1000 controller manual.	RW

## 4.5 Project variables from the table

The editor has access to the internal project variables. External variables are excluded, as they cannot be accessed at any time. External variables are only available if they are included on the currently displayed page.

The table of variables can be edited by clicking the button. The respective screen masks are identical with the masks of the project editor. For more detailed descriptions, please refer to the project editor. In short:

The table of variables is displayed on the screen:

Variablenabelle								
Loc	Name	Format	Handle	Ty	Untergrenze	Obergrenze	Schritt	Kommentar
S	I_Datum	@@:@@:@@	65534	15				
S	I_Uhrzeit	@@:@@:@@	65533	14				
S	I_Wochentag	@	65532	3				
S	I_Laufzeit	@@:@@:@@	65531	13				
S	I_Sprache	@	65530	3	0	9		
S	I_Priority	@@	65528	3	0	15		

OK    verwerfen    Neue Var    Var löschen

If you double click a variable now, you can modify the variable settings. Set your desired values.

To enter a new variable into the table of variables, click the "New var..." button.

If you want to delete a variable, first click the mouse in the line of the respective variable and click the "Delete var" button.

When clicking the "reject" button, you exit the variable list and undo all modifications in the table. Finally click "OK" to save your entries/modifications. For more details regarding the variables, please refer to the "Operating and Monitoring" operating panel manual.

## 5 Creating ladder diagrams


This chapter provides an overview of the creation and editing of ladder diagrams.

### 5.1 Ladder diagram structure

The ladder diagram comprises three parts, which can be programmed individually.


#### 5.1.1 Initialisation part

This part of the ladder diagram is processed whenever the operating panel is switched on or a device reset is executed via a command. This command can, for example, originate from the CAN bus.

If you click the  button, the LD displays the initialisation part for editing.

#### 5.1.2 Cyclic part

This part of the ladder diagram is processed more or less regularly. Depending on the processor utilisation, the time intervals range between 40 msec and 500 msec. The programming of this part of the ladder diagram is designed for the control of non-time-critical functions and processes.

The cyclic part can be edited by clicking the  button.

#### 5.1.3 Time-controlled part

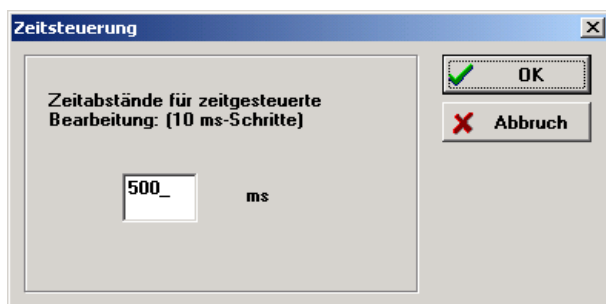
If you really want to control time-critical functions with the operating panel, the operating panel provides a time-controlled part.

To call up the edit mode for the time-controlled

part, click the  button.

The execution interval can be set by the user. It can be set in steps of 10 msec. The minimum time interval is 20 msec.

The center line of the status bar displays the cycle time at the right. If you double-click this field, the input mask for the time setting is called up:



Enter the desired time interval. Independent of the processor utilisation, the operating panel will then always execute this program part within these intervals.

To prevent the operating panel from processing

only the time-controlled program and from neglecting the page build-up, you should not create more than 64 lines. Make also sure to select an appropriate interval time. It is useless, for example, to execute the program part every 20 msec if you receive new values every 500 msec.

### 5.2 Working in the ladder diagram

First of all, you have to select the desired program part (initialisation, cyclic or time-controlled part) by clicking the button of the desired program part. The ladder diagram immediately opens the respective part of the ladder diagram.

To be able to modify a ladder diagram by setting switch, relay or wiring elements, at least one line must be inserted. For this purpose, use the editing functions of the tool bar.

#### 5.2.1 Editing functions

The tool bar contains a number of important editing functions which are required for editing the ladder diagram:



##### Add LD line

Via this button, a new line is added to the bottom line of the ladder diagram.

##### Important:

You always have to add an empty line to your ladder diagram before you can insert switch, relay or wiring elements!



##### Insert LD line

Click this button, it remains down (nothing has happened yet). Move the mouse cursor to the ladder diagram line in front of which you want to insert the new line and press the left mouse key. The line is inserted.



##### Insert comment line

Click this button, it remains down (nothing has happened yet). Move the mouse cursor to the ladder diagram line in front of which you want to insert the new comment line and press the left mouse key. The line is inserted and a dialogue field appears in which you can enter a comment to improve the transparency of your ladder diagram.



##### Delete LD line

Click this button, it remains down (nothing has happened yet). Move the mouse cursor to the ladder diagram line to be deleted and press the left mouse key. You are asked whether you really want to delete this line. If you select "Yes", the line is deleted.

# Operating panel manual

---

## 5.2.2 Special functions

Apart from the editing functions, the tool bar contains the following special functions to facilitate the ladder diagram work.



### Edit table of variables

This button calls up the table of variables, which can then be edited as in the editor by adding, deleting and modifying variables ...



### Copy LD line

You may only copy the content of one line into another (existing) line. Click this button, it remains down. Then, click the line to be copied (source). The line number of this line is displayed in red. Click the line into which you want to copy the source line. The line is then copied.



### Find...

If you want to activate the "Find" function, click this button and enter the element to be searched for. You can search for a variable name, an @BILD page call-up, etc. A new window shows up and displays all tiles where the search term has been found. Please observe: The "Find" functions takes capitalisation/use of small letters into account.



### Go to line

Jumps to the indicated line. This function is useful if a large number of lines has been programmed.



This tool has been designed for the parameterisation of switches and relays. Activate this tool and double-click the desired element.

## 5.2.3 Insert elements

Proceed as follows to insert elements:

1. Go to the tool bar above the ladder diagram and click the button for the desired element. The button remains down.
2. Place the mouse cursor on the ladder diagram position where the element is to be inserted and press the left mouse key. The element appears at the respective position.
3. With wiring elements, the button of the element selected last remains down; with switch elements, the arrow is activated.

## 5.2.4 Move elements

Elements can be moved in the following way:

1. Click the left mouse key on the element and keep the key pressed.
2. Move the mouse cursor (with the key pressed) to the desired position and release the mouse key.
3. The switch element is inserted at the new position and the old position is usually filled with a horizontal line.

## 5.2.5 Replace element

You also have the possibility to replace switch elements. Please proceed as follows:

1. Select the element from the tool bar which is to replace another element in the ladder diagram.
2. Simply click the tile where you want to replace the element.

If you replace a switch or a relay by another switch (or relay), the condition (or function) of this switch/relay is as far as possible maintained.

**In any case, check the settings!**

## 5.2.6 Enter comments


For documentation purposes, you may want to enter comments into the ladder diagram.

Within the ladder diagram, wiring elements or blank fields (not switches and relays) may be labeled. Please proceed as follows:

1. Select the "arrow" tool from the tool bar.
2. Double-click the element you want to label. The following input mask shows up:
3. Enter the desired label text in the entry line of the dialogue field. By clicking OK, the labeling is taken over into the ladder diagram.

## 5.2.7 Delete comment

If you want to delete a free labeling, delete the text from the entry line of the dialogue field.

If you want to delete a comment line, delete the whole line via the  button.

## 5.3 Logic operations

Logic operations can be realised by connecting switches in parallel or in series. Principally, such functions are not subject to any restrictions. A line can only take up a maximum of 12 elements. Of course, at least one relay is used for each switch element connection!

In the following, we will present some simple diagrams as examples.

### 5.3.1 Serial operation

A LED is to indicate when a certain message is active. For this purpose, a switch and a relay are required which are connected as follows:

# Operating panel manual



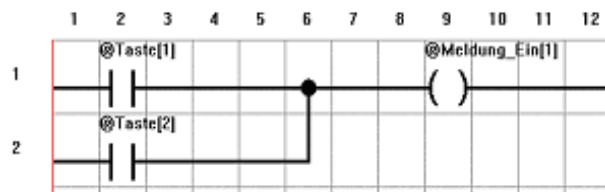
First, create a blank line before you set a switch and a relay. Double-click the switch and select "message active". Enter the message number in the "number" field. Select OK.


Double-click the relay. Select "turn on LED" and enter the LED number in the "number" field.

This procedure is always identical for all ladder diagram parts.

## 5.3.2 OR operation

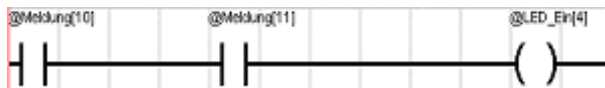
A message is to be displayed if one key or another key is pressed. The following ladder diagram realises this operation:



The blank fields in line 2, column 7 and the following are created via the  button.

## 5.3.3 AND operation

A LED is to indicate when two messages are active at the same time. This can be realised as follows:



Only if both switches close (i.e. both messages are active), current is carried into the relay and the LED is illuminated.

AND operations underly an additional rule in connection with cancel and edge controlled closing switches:

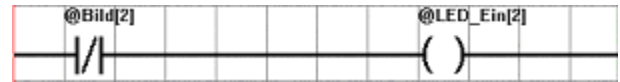
**Do not set any edge-controlled elements downstream a cancel closing switch!**

This could have negative effects in some cases, e.g. if the edge-controlled switch element had a conditional status change and if it was not actuated due to the cancel closing switch.

**Therefore: No edges after cancel.**

## 5.3.4 NOT operation

A NOT operation is realised with an opening switch instead of a closing switch. Example: A LED is to display when a page is not active:



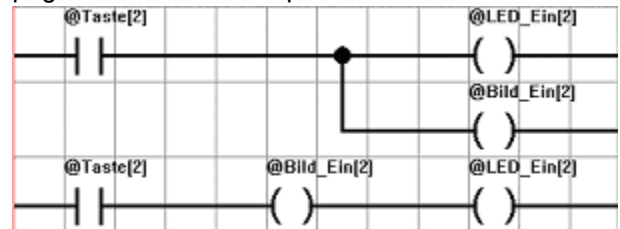
The same solution but in reverse order would result in the following ladder diagram. In this case, "turn off LED if page is active" is more suitable:



Nevertheless, we receive the same result.

## 5.3.5 Several operations

Sometimes several operations must be carried out if a condition is fulfilled. In such cases, you can connect relays in series or in parallel. Example: If key 4 is pressed, LED 4 is to be switched on and page 2 is to be called up:



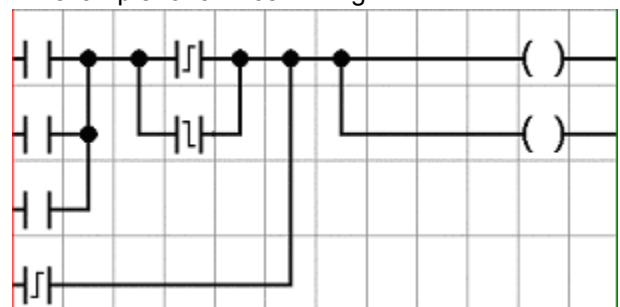
Both paths have the same result.

## 5.3.6 Complex operations

The setting options for switches and relays are almost unlimited. You can implement AND, OR and NOT operations in accordance with your individual requirements.

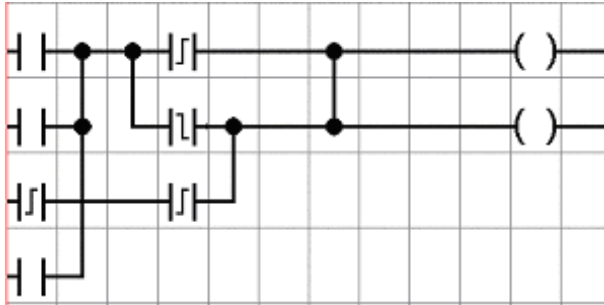
You should, however, set a joint result point to which you attach the relay(s). Otherwise, the switching logic may fast become intransparent.

An example for a "nice" wiring:



# Operating panel manual

Here, the connection logic is still transparent. The following illustration shows the same logic, however, with a chaotic wiring:



You have to look carefully to understand how it works. The transparency of the wiring has of course no impact on the processing in the control program. The LD translates both ladder diagrams into a functioning program.

## 5.4 Parameterisation of switch elements

All switch elements must be parameterised, i.e. their input conditions must be defined. First, activate the button with the red arrow. Then, move the mouse cursor over the switch element to be parameterised and double-click the switch element. The respective element parameterisation mask shows up. You can also call up parameterisation masks by simply clicking the right mouse key on the switch element. The following dialogue appears.

Enter the desired request. Within this context, the following rules must be observed:

- First, select the request type by clicking the respective button. Depending on the request type, further fields appear.
- Fill in the respective fields.

The different fields contain different entry options.

### 5.4.1 Variable fields

Here you can indicate all internal variables from the table of variables, which are listed in the respective field. Additionally, you may also indicate the LD variables.

The variables can also be accessed indirectly by putting the variable name into parenthesis, e.g.: (INDEXVAR)

You may also indirectly access flag bytes, flag words and flag doubled words if you keep the number of the desired flag word in a flag byte and put the number of this flag word into square parenthesis.

@MW[3] means: flag word, whose number is contained in @MB3.

### 5.4.2 Value fields

Enter constant values into these fields. The following formats are available:

**Decimal**, no format indication required.

Example: 234

Observe that the LD converts the value 12.5 to 125. Internally, LD always calculates with integers.

**Hexadecimal**, format indication: prefix 0x

Example: 0x33

**Date indication**: prefix D

Example: D05.08.02 (Ddd.mm.yy)

**Time indication**: prefix U

Example: U08:10:00 (Uhh.mm.ss)

LD converts the format in accordance with the comparison variable. Date/time variables are BCD coded, while integer variables are binary coded. LD considers the different coding systems for conversion.

#### "Number/variable/SDO" field

A value, a variable or an indirect addressing may be entered here.

The following sections describe the functions and respective fields.

#### 5.4.2.1 Variable <compare> variable

This function compares the value of two variables. Select the variables to be compared via the variable fields and the respective comparison.

#### 5.4.2.2 System variable <compare> variable

This function compares the value of a system variable with the value of a variable. Select the system variable in the first field, then the comparison and finally the variable.

# Operating panel manual

## 5.4.2.3 Variable <compare> value

Compares a variable with a fixed value. Fill in the fields respectively.

## 5.4.2.4 System variable <compare> value

Compares a system variable with a fixed value. Fill in the fields respectively.

## 5.4.2.5 Nominal variable is sent

Using this function, you can check whether a nominal value which was modified and fitted into the transmission queue has already left the device. This way, e.g., password entries can be immediately deleted after the variable has been output.

## 5.4.2.6 Message active

This function checks whether a message is active. The message does not need to be visible on the display, e.g. if several messages are active at the same time. Enter the message number in the "number/variable/SDO" field.

## 5.4.2.7 Page active

This function checks whether a page is active. The page does not need to be visible on the display, e.g. if several pages are active at the same time. Enter the page number in the "number/variable/SDO" field.

## 5.4.2.8 Priority page

This function works like the "page active" function, though it checks whether the page has been activated with priority.

## 5.4.2.9 LED is on

The "LED is on" function checks whether the LED indicated in the "number/variable/SDO" field is switched on.

## 5.4.2.10 Key touched

This is one of the most frequently used functions. Indicates whether the key displayed in the "number/variable/SDO" field is touched.

## 5.4.2.11 Output is on

This functions checks whether the integrated output is on.

## 5.4.2.12 SDO status check

This option is required for CANopen-master functions. Enter the number you have entered in the "SDO number" field of the SDO transmission function into the "number/variable/SDO" field. Further information on this topic is available in the Com-

munication manual.

## 5.4.2.13 Timer running

The LD is equipped with more than 10 timers (0-9), which operate with a 100 msec resolution, internally with an accuracy of 20 msec (inaccuracy). This call-up is to check whether a timer is set to 0 (timer stopped) or whether it is unequal 0 (timer running). Enter the timer number in the "number/variable/SDO" field.

## 5.4.2.14 Timer stops

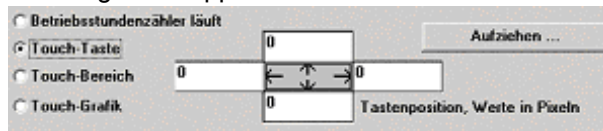
Inverse function to "timer runs".

## 5.4.2.15 Hour timer running

Each internal variable can be used as an hour timer. A variable can be started as an hour timer via a relay. In this case, you can check via the "hour timer running" function, whether a variable is processed as hour timer. Enter the variable to be checked in the "number/variable/SDO" field.

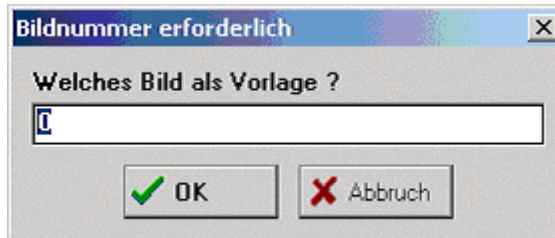
## 5.4.2.16 Touch key

With touchscreen equipped devices, this function checks whether a "key stroke" has been registered in the indicated area. Via this function, a particular touch screen surface can be used as a key. The following fields appear:



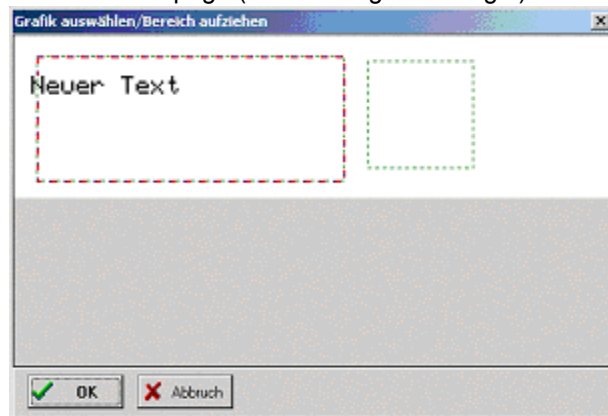
Enter the limitations of the key in pixels (!). Left field = left limitation, right field = right limitation etc. You can use fixed values, variables or indexed variables.

When clicking the "Drag area" button, a window shows up where you have to enter the page for the key:



# Operating panel manual

Enter the page to be displayed to position the touch key. Then, you can draw the key directly with the mouse on the page (like drawing a rectangle):



The red broken line indicates the initial setting, the green one the new drawn area. Via OK, the limitations are automatically taken over into the touch fields.

## 5.4.2.17 Touch area

Checks whether a point is touched within a certain screen area. As opposed to the touch key, this point may also move and dynamic elements such as slide controllers can be implemented. The fields are to be adjusted as with the touch key or filled via the "Drag area" button.

## 5.4.2.18 Touch bitmap

Particular bitmaps are frequently used as touch key for particular pages. This function realises this task in a comfortable way:



Enter the number of the page which contains the bitmap into the "number/variable/SDO" field and touch "select bitmap":



The currently adjusted bitmap appears with a red broken frame. If you click the mouse onto another bitmap, this bitmap appears with a green frame and indicates the new selection. The bitmap name is displayed underneath the "select bitmap" button.

Basically, this function is a combination of the request "page (number) is displayed and the touch key with area above the graphic", i.e. you only have to parameterise one closing switch ....

## 5.4.2.19 User/Flash available

If the project does not occupy bank 11 in the memory, 32 kb flash memories are freely available for the user, which start with address 0 in bank 11. Via this request, you can make sure that bank 11 is not occupied with project data.

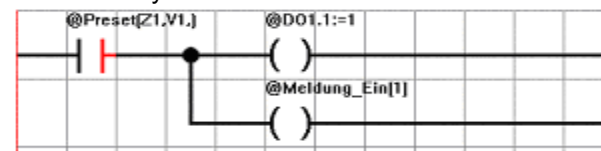
## 5.4.2.20 Preset of fast counter reached

This is a special function which is only available in devices with internal I/Os and counters.

As counters can count with a frequency of up to 10 kHz, it would be nonsense to request them with the smallest time-controlled interval of 20 msec. The inaccuracy would be excessive.

For this reason, the program offers the possibility to define 2 presets per counter channel via system variables which initiate a fast response.

By setting a "preset achieved" closing switch, you create an "interrupt" program, i.e. the response program to "preset achieved". This program should always be created as follows:



The LD part downstream the closing switch is executed with a delay of a maximum of approx. 2 msec.

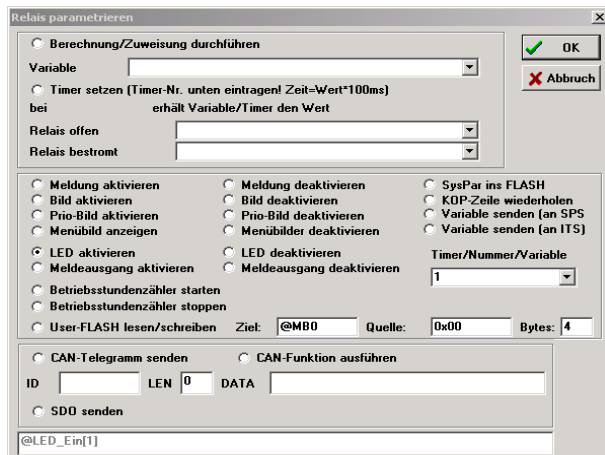
To ensure a correct function, the following rules must be adhered to:

1. Only 1 closing switch of this type is allowed to appear with every preset.
2. A "closing switch with cancel" must be used.
3. The closing switch must always be connected directly left to the busbar. No switch element must be in front of it.
4. Behind the closing switch, the number of functions to be processed should be reduced to a minimum. Consider that each function requires time.

# Operating panel manual

## 5.5 Relay parameterisation

Apart from the switch elements, relays must be parameterised as well, i.e. their reactions towards the input conditions must be defined. First, activate the button with the red arrow. Then move the mouse cursor over the relay to be parameterised and double-click the relay. The respective relay parameterisation mask shows up. The following dialogue appears.



1. Select the desired function. Within this context, the following rules must be observed:
2. First, select the type of action and click the respective selection button. Then, fields may appear which are only visible with the selected function.
3. The field in the lower area of the mask indicates the adjusted action in text form. This field is automatically updated during the processing of the action and cannot be edited by the user.
4. If you assign a value, a relay can assign a value depending on its status (open or closed).
5. Enter the value to be assigned to a variable in the "relay open"/"relay closed" fields. Observe: The "relay closed" field must always be filled in, while the "relay open" field may also stay free. Also arithmetic operations are possible here.
6. Calculations: Currently, you can execute the following calculations:

Variable+value (addition)  
Variable-value (subtraction)  
Variable\*value (multiplication)  
Variable/value (division)  
Variable AND value (blank out bits AND)  
Variable OR value (set bits OR)  
Variable XOR value (invert bits XOR)  
Variable+variable (addition)  
Variable-variable (subtraction)  
Variable\*variable (multiplication)

Variable/variable (division)  
Variable AND variable (bit-coded AND)  
Variable OR variable (bit-coded OR)  
Variable XOR variable (bit-coded XOR)  
Variable (restore)

The variable name must be typed in "manually". Observe the use of capitalised/small letters! With arithmetic operations (+,-,\*,/) no blanks must be positioned between variable, operator and value. In case of divisions or multiplications, the value is considered as real decimal number, i.e. the decimal point is fully evaluated and the arithmetic operation is executed with this fraction. With additions/subtractions, the decimal point is deleted and the number resulting thereof is added/subtracted to/from the variable as an integer. With logic operations (AND, OR, XOR), a blank must be set between variable, operator and value. If you want to enter a hexadecimal number, mark this number with "0x", e.g. 0x13D.

7. Variables can also be directly addressed, e.g. (INDEX) or @MB[2].  
Example: (INDEX1)+(INDEX2) adds up the variables whose handles are stored in the variables INDEX1 or INDEX2.
8. For the "start hour timer" or "stop hour timer" actions, select the appropriate variable from the list box.
9. With all other actions, enter the number into the "number" field.
10. With CAN telegrams, you have to enter the identifier "ID" in decimals or as variable; the length of the telegram ranges from 0 to 8. Fixed entered values can be transmitted as data. The entries must be made in HEX form or as variables. The low-byte of the variable is transmitted as D0, the next byte as D1 etc. A variable may have a length of a maximum of 4 bytes.

### 5.5.1 Description of actions

Depending on the "closed" / "open" status, the individual functions carry out different actions. It must be observed that different relay types initiate different reactions.

Some actions, for example, imply an "inverse action", which we refer to as "implicit" in the following sections. The other actions are called "simple".

# Operating panel manual

---

## 5.5.1.1 Set variable/calculate

**Type of action:** simple

This action enables the calculation or restorage of values. The possible calculation types are listed above.

Enter the location in the "variable" field where the results of the calculations are to be stored. An indirect addressing is also possible here.

Enter the type of calculation to be executed if the relay is in its open state in the "relay open" field. This field may also be skipped.

In the "relay closed" field, enter the type of calculation to be executed if the relay is in its closed state. This field must be filled in.

Example: Calculation of a variable value. If you use a switch with the request "VALUE smaller than 0", you can enter the calculation "0-VALUE" into the downstream relay in the "relay closed" branch and the calculation "VALUE" into the "relay open" branch. The result of this calculation is the value of the "VALUE" variable.

## 5.5.1.2 Set timer

**Type of action:** simple

10 timers, 0 - 9 are available, which can be set in steps of 100 msec.

The operating system always counts down the timers to 0. The LD can check whether these timers are in the "running" or "stopped (=0)" mode.

Enter the timer number into the "timer/number/variable" field. This number can also indirectly be determined via, e.g., @MB[5] or (VAR).

Enter the calculation for the number of 100 msec steps with an open relay into the "relay open" field. This field may be skipped.

Enter the calculation of the number of 100 msec steps with a closed relay into the "relay closed" field. This field must be filled in.

## 5.5.1.3 Activate message

**Type of action:** implicit

The message indicated in the "timer/number/variable" field is activated with a closed relay. If the relay is open, the message is deactivated. The message number can be addressed directly as fixed value, indirectly via a variable or double indirectly.

## 5.5.1.4 Deactivate message

**Type of action:** implicit

The message stated in the "timer/number/variable" field is deactivated in the "relay closed" case and activated in the "relay open" case(!) The message number can be addressed directly as fixed value, indirectly via a variable or double indirectly. Note: -1 deletes all messages

## 5.5.1.5 FLASH new SysPars

**Type of action:** simple

In the "relay closed" case, the values saved in the system variables are permanently taken over into the FLASH.

## 5.5.1.6 Activate page

**Type of action:** implicit

The page stated in the "timer/number/variable" field is activated in the "relay closed" case and deactivated in the "relay open" case. The page number can be addressed directly as fixed value, indirectly via a variable or double indirectly.

## 5.5.1.7 Deactivate page

**Type of action:** implicit

The page stated in the "timer/number/variable" field is deactivated in the "relay closed" case and activated in the "relay open" case. The page number can be addressed directly (fixed value), indirectly via a variable or double indirectly. Note: -1 deletes all pages.

## 5.5.1.8 Repeat LD lines

**Type of action:** implicit

Usually, a ladder diagram program is executed linearly from the left to the right, from the top to the bottom.

If you want to set, e.g., several variables with consecutive handles to one value, a program loop would be desirable. This is exactly the task of the "repeat" relay.

When using repeat relays, please observe the following:

- *Make sure that the relay can be connected only from one position of the left bus bar. If required, set a branch in column 1 and start to connect.*
- *Make sure that the relay is also actuated in the "open" state or not at all to avoid infinite loops.*

In the "closed" state, the relay continues the LD program at the start of the network at the left side of the bus bar after the entire currently active network has been processed. In the "open" state, the

# Operating panel manual

---

network is not further processed.

## 5.5.1.9 Activate prio-page

**Type of action:** implicit

A "prio-page" is a page with high priority which definitely shows up when being called up, independently of the currently displayed elements. Even the entries of an operator are interrupted if a priority page is called up! When using prio-pages, the action relays should only be actuated with edge triggering.

The prio-page indicated in the "timer/number/variable" field is activated in the "relay closed" state and deactivated in the "relay open" state. The page number can be addressed directly (fixed value), indirectly via a variable or double indirectly.

## 5.5.1.10 Deactivate prio-page

**Type of action:** implicit

The page indicated in the "timer/number/variable" field is deactivated in the "relay closed" state and activated in the "relay open" state. The page number can be addressed directly as fixed value, indirectly via a variable or double indirectly.

Note: -1 deactivates all pages.

## 5.5.1.11 Send variable (to PLC)

**Type of action:** simple

In the "relay closed" state, the variable indicated in the "timer/number/variable" field is sent with the "REPORT VARIABLE VALUE" (TA = 0x03) CAN-telegram format.

## 5.5.1.12 Show menu page

**Type of action:** simple

A "menu page" is a page where the user can navigate or carry out entries. Usually, he first has to touch a key to get the cursor displayed; the menu page call-up "shortens" this procedure and directly shows the cursor.

The menu page indicated in the "timer/number/variable" field is activated in the "relay closed" state.

The page number can be addressed directly as fixed value, indirectly via a variable or double indirectly.

## 5.5.1.13 Deactivate menu pages

**Type of action:** simple

Within menu pages, menu items can be set in a way which allows them to call up new menu pages upon selection. This results in a structure with a depth of up to 16 pages. The "deactivate menu pages" action removes a certain number (not particular menu pages) from this structure.

The action is only carried out in the "relay closed" state.

Note: -1 deletes the entire structure

The "timer/number/variable" field indicates the number of backward steps.

## 5.5.1.14 Send variable (to ITS)

**Type of action:** simple

In the "relay closed" state, the variable indicated in the "timer/number/variable" field is sent in the "SET VARIABLE VALUE" (TT = 0x03) CAN-telegram format. This way, variables can be easily exchanged between two operating panels.

## 5.5.1.15 Turn on LED

**Type of action:** implicit

The LED indicated in the "timer/number/variable" field is switched on in the "relay closed" state and switched off in the "relay open" state(!)

## 5.5.1.16 Turn off LED

**Type of action:** implicit

The LED indicated in the "timer/number/variable" field is switched off in the "relay closed" and switched on in the "relay open" state(!)

## 5.5.1.17 Turn on output

**Type of action:** implicit

The output is switched on in the "relay closed" state and switched off in the "relay open" state.

## 5.5.1.18 Turn off output

**Type of action:** implicit

The output is switched off in the "relay closed" state and switched on in the "relay open" state.

## 5.5.1.19 Start hour counter

**Type of action:** implicit

Each internal variable can be used as an hour counter. If you start a variable as an hour counter, the operating system raises this variable each second by 1.

The variable is indicated in the "timer/number/variable" field.

The hour counter is started in the "relay closed" state and stopped in the "relay open" state (!). Consequently, it counts as long as the relay is closed.

## 5.5.1.20 Stop hour counter

**Type of action:** implicit

The variable is indicated in the "timer/number/variable" field.

The hour counter is stopped in the "relay closed" state and started in the "relay open" state(!). Consequently, it counts as long as the relay is open.

# Operating panel manual

---

## 5.5.1.21 Read/write user flash

**Type of action:** simple

If the relay is closed, all bytes indicated in the "bytes" field are copied from "source" to "target".

If a flag byte is indicated as source, a fixed value must be entered as index/offset into the FLASH memory as target. Alternatively, a variable which contains the index-offset can be entered.

If an index/offset is indicated as source in the FLASH memory, a flag byte must be indicated as target. Copying is only possible from the flash into flag bytes or vice-versa.

**Direct copying within the FLASH is not possible.**

## 5.5.1.22 Send CAN telegram

**Type of action:** simple

If the relay is closed, a CAN telegram is sent to the CAN bus. Enter the ID under which the telegram is to be sent into the "ID" field. (Note: The @SER entry sends the telegram to the serial interface).

Enter the number of data bytes to be sent into the LEN field.

(Note: If you want to use the RTR-bit, add 16 in the LEN field)

Enter the data to be sent into the "DATA" field. The following options are available:

- *All bytes in series, noted in hexadecimal form, without hyphens.*

**Example:** You want to send 0x23 0x45 0x12 0x77. Enter "23451277" in the "Data" field.

- *All bytes in series from one flag byte.*

Enter the first flag byte into the "Data" field, e.g. @MB10. Starting with MB10, flag bytes are used as data for the telegram (@MB11, @MB12....)

## 5.5.1.23 Execute CAN action

**Type of action:** simple

With this function, you can activate operating panel functions which do not have individually assigned actions in the relay parameterisation mask.

Only fill in the "Data" field as described under the "send CAN telegram" action. Use the Communication manual as a reference for possible functions.

The CAN function is executed if the relay is closed.

The exact mode of operation of the different switches and relays is described in the following chapter.

# Operating panel manual

## 6 How can I...?

Now that you are familiar with the theory, let's go into practice. We would like to provide you with some tips for LD applications. The following sections will give you an impression of what lies all behind the ladder diagram.

This chapter provides support for a wide field of problems. We tried to structure the following sections into problem fields.

Let's get started!

### 6.1 Put a message onto an input?

For this purpose, we use a conventional closing switch and a conventional relay. The "activate message" function of a conventional relay includes "deactivate message" in the open state:

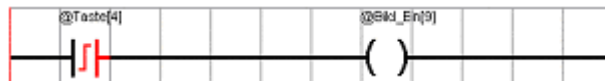


The message remains active as long as the input is active.

### 6.2 Activate pages via keys?

You will certainly need this function as the following problem comes up: If you touch a key, the page appears, but what happens if you release this key?

The solution is a "positive edge closing switch with cancel". The solution for this problem is as follows:



Page number 9 shows up if you touch key No. 4...

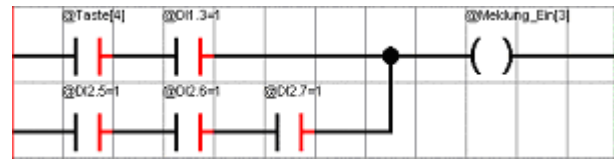
### 6.3 Activate a message via several conditions?

This is also a frequently asked question. Here, we need a closing switch with cancel:

Imagine you want to receive message No. 3 if

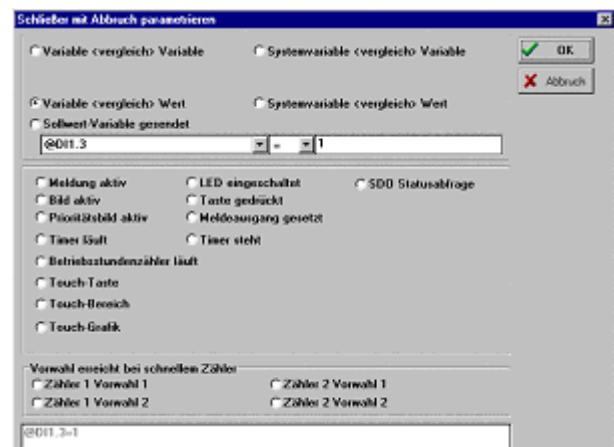
- key 4 is touched and input 3 at CAN module 1 is set at the same time
- or
- the inputs 5, 6 and 7 are set at CAN module 2.

The following ladder diagram solves the problem:



It is recommendable to use closing switches with cancel at the respective positions, as the "call up message" relay would implicitly output the message in the open state ...

The following mask is an example of an input mask for a closing switch which has been set to an input:

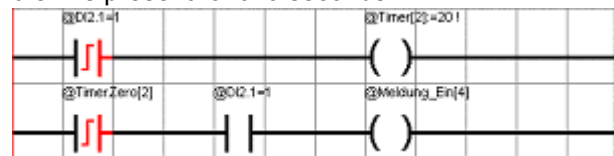


Carry out these settings for all closing switches which are to request a CAN input.

### 6.4 Realise a switch-on delay?

A switch-on delay can be realised by means of a timer. Please refer to the following example:

A message is to be displayed if input at CAN module 2 is present for two seconds:



The function relay in the first line is parameterised as follows:



Parameterise the timer request in the second line as follows:



Function description: In the first line, timer 2 is restarted with each positive edge of the input signal. If the timer expires although the input is not set yet, the message is activated (line 2).

## 6.5 Acknowledge a message via a key?

Acknowledging and deactivating - two reactions to one message.

The operating panel itself offers the possibility to acknowledge messages (i.e. the operator has to touch a key to signalise the device that he has read the message). A further acknowledgement type is the deactivation - removal from the message stack.

This is a very easy procedure:

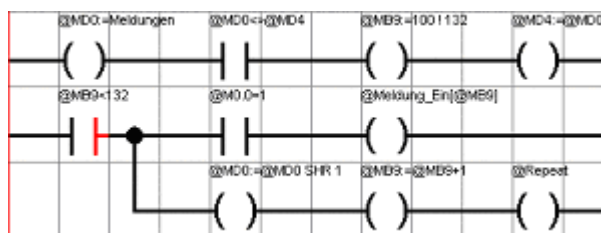


Simply select "deactivate message" in the function relay.

## 6.6 Administer variable bits for message call-ups?

This is also a frequently asked question. Fortunately, this function can be realised quite easily. Save the variable to a flag word (or flag double word) and request the bits via the flags. For this purpose, you may use the "repeat LD line" loop function.

**Example:** In the "messages" variable (a long variable with 32 bits), the bit numbers 0 to 31 are to represent messages 100 to 131. The ladder diagram illustrated below already optimises the run-time:



At first sight, the diagram appears to be confusing. It is, however, quite easy to understand:

First line:

In the MD0 flag double word, the bit pattern of the messages variable is stored and compared with the bit pattern of the previous cycle. If the bit patterns are not identical, the start message number in MB9 is set to 100 and the new status is restored under MD4. If the bit patterns are identical, the start number is set to 132.

Second line:

The network is executed if the start number is smaller than 132. Bit 0 is requested from MD0 and the message whose number is currently kept in MB9 is activated or deactivated.

Third line:

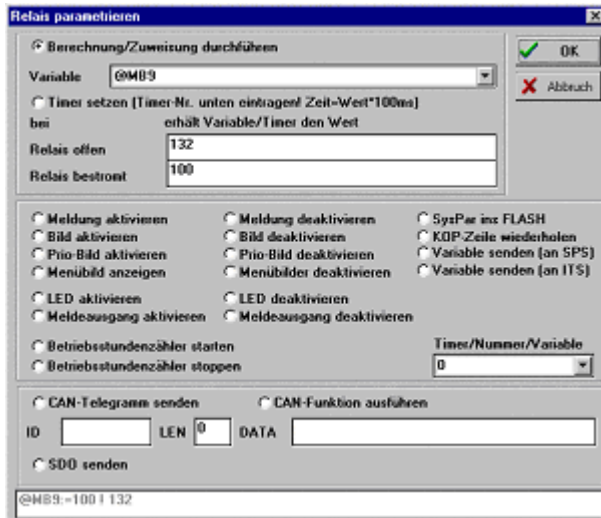
The MD0 flag double word is moved 1 bit to the right and the message number is counted up in MB9. The "repeat LD line" relay finally instructs the LD to continue with line 2.

After exactly 32 cycles, MB9 exceeds 131. This way, the request of the first closing switch is no longer fulfilled and LD continues with line 4.

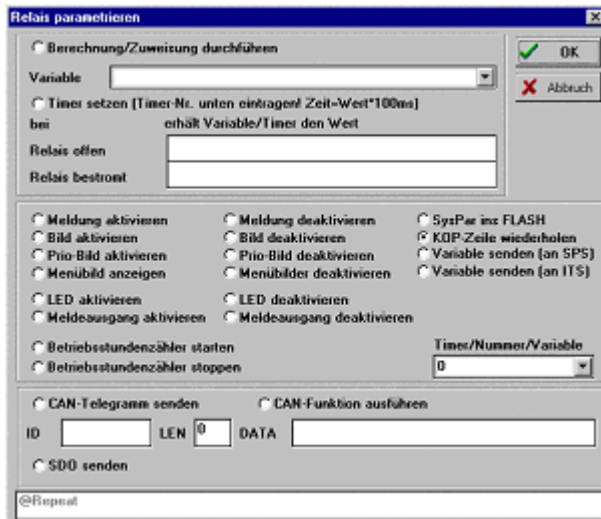
The parameterisation of special relays is described for reasons of completeness.

Line 1, column 8: Assignment of the start number

# Operating panel manual



Line 3, column 11: Repeat function



## 6.7 Scale a variable?

This problem also refers to CAN analogue inputs. Basically: You have to carry out the calculation step by step and use a flag word as buffer. An example: The "TEMPDIG" variable represents an unscaled digital measuring value from 0 to 1023. The respective temperature range lies between -30° and +50°, a decimal is to be displayed (e.g. 12.7)

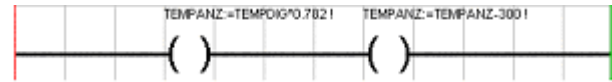
Solution: First, create a "TEMPANZ" variable (or any other variable) and develop the equation.

The TEMPANZ variable receives the tenfold value (because it has a decimal). The range lies between -30° (= -300) and 50° (=+500), i.e. has a range of 80° (=800).

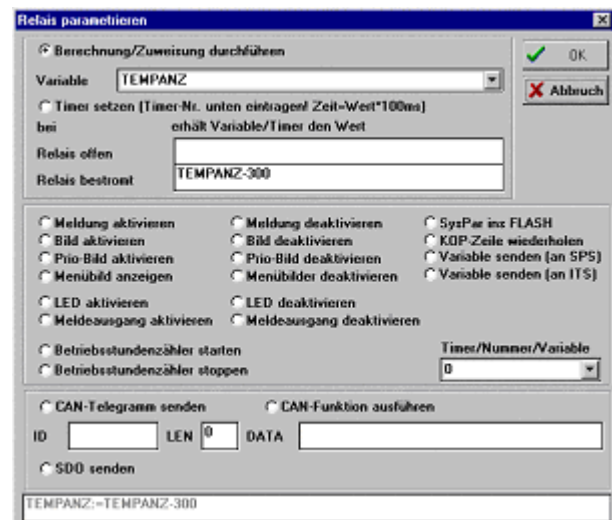
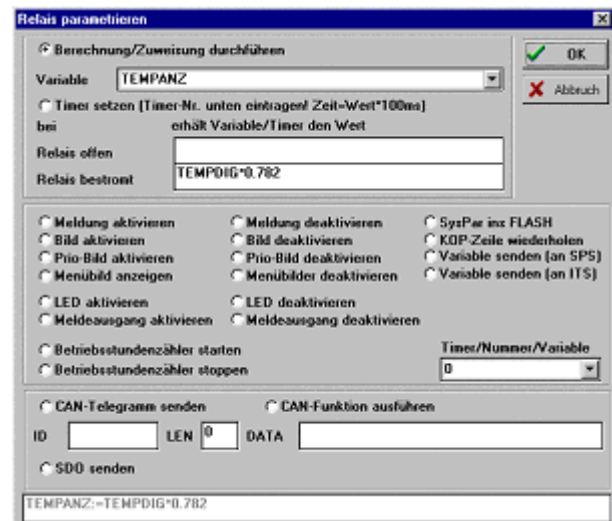
In our case, we use the following equation:  
 $TEMPANZ = TEMPDIG * 800 / 1023 - 300$

Simplified:  
 $TEMPANZ = 0.782 * TEMPDIG - 300$

Now we transfer this calculation into the ladder diagram:



The masks for the relay settings:

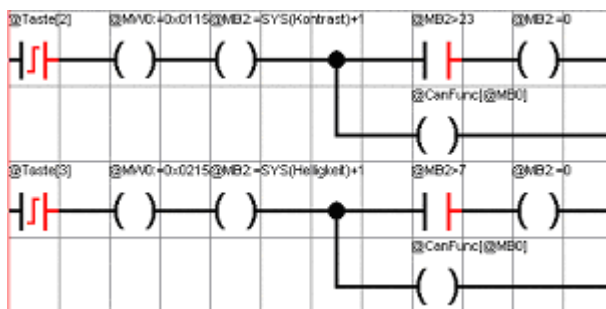


# Operating panel manual

## 6.8 Set contrast and brightness via function keys?

Via the system variables, contrast and brightness can be read but not modified. The modifications must be carried out via the "execute CAN function" relay function while building up the "CAN telegram" in the flag memory. The function can then be triggered via a key.

In the following example, we will use F2 to gradually set the brightness and F3 respectively for the contrast (incremental):



Description: two lines always belong together, the blocks basically function the same way.

Line 1/3:

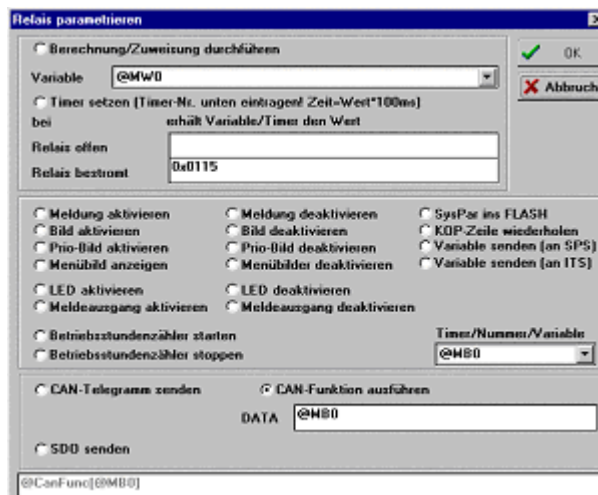
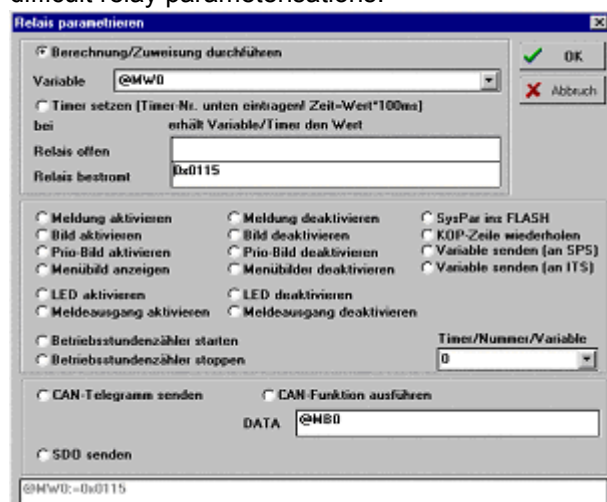
If the respective key is touched, MB0 and MB1 are first pre-assigned as follows: MB0 with 0x15 and MB1 with 0x01/0x02. This procedure is executed with the first relay via flag word access.

The values are contained in the "Param" telegram description with TT=0x15 (=MB0).

The second relay gets the current setting, adds 1 and saves it into MB2. Here, the setting is performed in the CAN telegram. The closing switch with cancel then checks the maximum value and, if required, resets the value to 0.

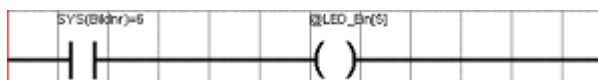
Line 2/4:

Only the CAN function is executed here. The data are used starting with @MB0, @MB1 etc. The two difficult relay parameterisations:



## 6.9 Determine the page number of the currently displayed page?

This task can be easily realised by a system variable. You can request these variables in the closing switches or take up their value within a relay into a variable (if required). In our example, we request whether page number 6 is displayed and illuminate LED 5 if this is the case (implicit: LED is not illuminated if page 6 is not displayed):



To get an impression of the closing switch, refer to the following mask:



Why do we not use the "page active" request? Because several pages can be active at the same time. On the other hand, an active page does not necessarily have to be displayed.

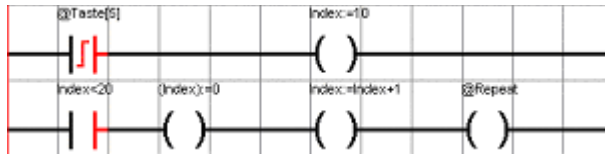
# Operating panel manual

## 6.10 Initialise several variables?

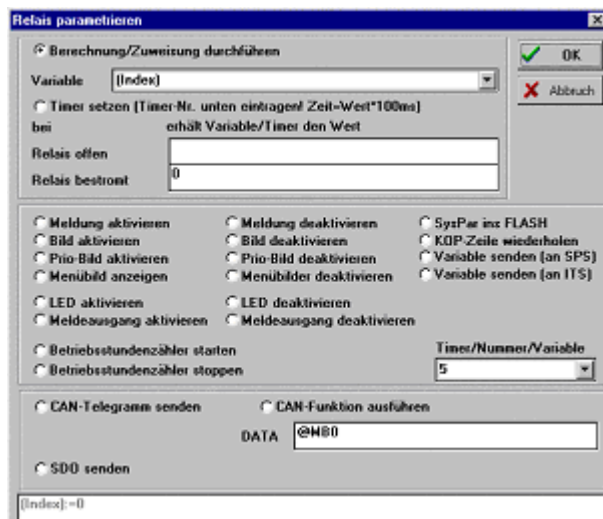
If the variable handles follow one after another, several variables can be initialised by simply using the repeat function. Save the first handle in an auxiliary variable and make it count up as index for the variables to be deleted.

Please also refer to the example "... administering variable bits for message call-ups?"

In this example, the variables with handles between 10 and 19 are to be deleted by touching a key:



In the first line, we set the index to the handle of the first variable. In the second line, the variable is set to 0 in the first relay, whose handle stands in the variable index:



Observe the indirect addressing (index)!

In the following relay, the index variable is counted up and in the last relay the line is repeated until the final index is reached.

# Operating panel manual

---

## 7 Compiling LD

After having edited a ladder diagram, you should first save the diagram before converting it into a device-compatible format via "Ladder diagram/compile". If you don't do this, the device also won't do anything ... but seriously, the LD editor displays a respective request prior to closing the file.

### 7.1 Enter LD program into the project

If you call up the LD editor from the ITE, the ITE automatically enters the LD program into the project settings.

During compilation, a file is created with the name of the project and the file ending ".HEX" for the ITS device series and the file ending ".ATX" for the AT device series. These file names are entered under "Device/Parametrize..." in the "Ladder diagram (LD)" field of the "Firmware/char set" index card.

### 7.2 LD compilation errors

During the compilation process, errors may occur which must be rectified. To provide you support for troubleshooting, the LD compiler lists all errors found in a window. You can position this window next to the LD editor and correct the errors step by step.

In addition to this, the error types which are supplemented by the line and column of the respective error give hints to the error reasons. By double clicking the error message, the LD editor automatically jumps to the indicated line.

The following list contains all possible LD errors, categorised according to error numbers:

#### 7.2.1 Error during connection check

##### 7.2.1.1 Error type 1 in line ... column ...:

No connection to...

**Description:**

The wiring element at the indicated position is not connected at one side.

**Rectification:**

Search the respective element and connect the open end.

#### 7.2.2 Error during program generation

##### 7.2.2.1 Error type 10 in line ... column ...:

Handle of variable ... is not numeric (variable table)

**Description:**

A wrong variable number (handle) has been en-

tered in the variable table. Blanks or letters may be contained in the number.

**Rectification:**

Edit the entry of the indicated variable in the variable table and ensure to correctly enter the number into the handle.

##### 7.2.2.2 Error type 11 in line ... column ...:

Variable ... not found

**Description:**

The variable is not contained in the variable table.

**Rectification:**

Check the correct spelling of the name of the indicated variable. Observe capitalisation/use of small letters!

##### 7.2.2.3 Error type 12 in line ... column ...:

Incorrect marker byte number: ....

**Description:**

Non-numeric characters have been found in the marker byte number.

**Rectification:**

Check for the correct spelling at the indicated position of the marker byte number and remove all non-numeric characters.

##### 7.2.2.4 Error type 13 in line ... column ...:

Marker byte number too high (max. 127): ...

**Description:**

128 marker bytes are available (0-127). You have indicated a number > 127.

**Rectification:**

Enter a permissible marker byte.

##### 7.2.2.5 Error type 14 in line ... column ...:

Incorrect marker word number: ....

**Description:**

Non-numeric characters have been found in the marker word number.

**Rectification:**

Check for the correct spelling at the indicated position of the marker word number and remove all non-numeric characters.

# Operating panel manual

---

## 7.2.2.6 Error type 15 in line ... column ...:

Marker word number too high (max. 126): ...

### Description:

128 marker bytes are available (0-127). Consequently, the marker word numbers range from 0 to 126.

### Rectification:

Enter a permissible marker word.

## 7.2.2.7 Error type 16 in line ... column ...:

Incorrect marker doubleword number: ....

### Description:

Non-numeric characters have been found in the marker doubleword number.

### Rectification:

Check for the correct spelling at the indicated position of the marker doubleword number and remove all non-numeric characters.

## 7.2.2.8 Error type 17 in line ... column ...:

Marker doubleword number too high (max. 124): ...

### Description:

128 marker bytes are available (0-127). Consequently, the marker doubleword numbers range between 0 and 124.

### Rectification:

Enter a permissible marker doubleword

## 7.2.2.9 Error type 18 in line ... column ...:

Invalid marker: ....

### Description:

A marker must be indicated in the @Mx.y. format  
Values for x: 0 - 127; values for y: 0 - 7

### Rectification:

Indicate a correct marker.

## 7.2.2.10 Error type 19 in line ... column ...:

Invalid digital input: ...

### Description:

The entry of the CAN-module digital input is faulty. The LD control program has direct access to the digital inputs/outputs of CAN modules; however only to the I/Os of the modules with the address 1-8. Here the inputs 0-47 and the outputs 0-23 can be addressed

You have probably used a wrong module number

and/or a wrong input number.

The input format is: @Dlx.y with: x module number 1-8, y input number 0-55

### Rectification:

Correct the input entry.

If you want to address a module outside the address range 1-8, you have to work via variables. This is, however, not possible in the time-processed program.

## 7.2.2.11 Error type 20 in line ... column ...:

Incorrect analogue input: ...

### Description:

The entry of the CAN-module analogue input is faulty.

The LD control program has direct access to the analogue inputs/outputs of CAN modules, however only to the I/Os of the modules with address 1-8. Here, the inputs 0-3 and the outputs 0-3 can be addressed.

You have probably used a wrong module number and/or wrong input number.

The input format is: @Alx.y with: x module number 1-8, y input number 0-3

### Rectification:

Correct the input entry.

If you want to address a module outside the address range 1-8, you have to work via variables. This is, however, not possible in the time processed program.

## 7.2.2.12 Error type 21 in line ... column ...:

Incorrect LD variable: ...

### Description:

The indicated variable cannot be identified. Valid variable types are: @Mx.y, @MBx, @MWx, @MDx, @Dlx.y, @DOx.y, @Alx.y, @AOx.y

### Rectification:

Use a permissible variable.

## 7.2.2.13 Error type 22 in line ... column ...:

Invalid digital output: ...

### Description:

The entry of the CAN-module digital output is faulty.

The LD control program has direct access to the digital inputs/outputs of CAN modules, however only to the I/Os of the modules with address 1-8. Here, the inputs 0-47 and the outputs 0-23 can be addressed.

You have probably used a wrong module number

# Operating panel manual

---

and/or a wrong output number.

The input format is: @DOx.y with: x module number 1-8, y output number 0-23

**Rectification:**

Correct the output entry.

If you want to address a module outside the address range 1-8, you have to work via variables. This is, however, not possible in the time processed program.

**7.2.2.14 Error type 23 in line ... column ...:**

Invalid analog output: ...

**Description:**

The entry of the CAN-module analog output is faulty.

The LD control program has direct access to the analogue inputs/outputs of CAN modules, however only to the I/Os of the modules with address 1-8. Here, the inputs 0-3 and the outputs 0-3 can be addressed.

You have probably used a wrong module number and/or wrong output number.

The input format is: @AOx.y with: x module number 1-8, y output number 0-3

**Rectification:**

Correct the output entry.

If you want to address a module outside the address range 1-8, you have to work via variables. This is, however, not possible in the time processed program.

**7.2.2.15 Error type 24 in line ... column ...:**

Indirect addressing invalid:

**Description:**

The indirect addressing of a marker byte was not allowed or faulty at the indicated position.

**Rectification:**

Enter the indirect addressing in the form @MB[23]. Put the number of the marker byte used as index into square parenthesis.

**7.2.2.16 Error type 50 in line ... column ...:**

Message ... is not in project

**Description:**

You have called up or requested a message which is not available in the project.

**Rectification:**

Exit the LD editor and create the message in the ITE editor before restarting the LD editor.

**7.2.2.17 Error type 51 in line ... column ...:**

Page ... has not been created

**Description:**

You have called up or requested a page which is not available in the project.

**Rectification:**

Exit the LD editor and create the page in the ITE editor before restarting the LD editor.

**7.2.2.18 Error type 52 in line ... column ...:**

CAN identifier isn't numeric.

**Description:**

Characters have been found in the CAN identifier of a relay with CAN transmission which cannot be converted into a numeric value.

**Rectification:**

Correct the entry of the CAN identifier. This can be

1. a fixed number
2. a variable
3. @SER (serial transmission)

**7.2.2.19 Error type 53 in line ... column ...:**

CAN identifier not entered.

**Description:**

The "ID" field of a relay with CAN transmission has not been filled in.

**Rectification:**

Complete the entry of the CAN identifier.

**7.2.2.20 Error type 54 in line ... column ...:**

CAN data length not entered.

**Description:**

The "LEN" field of a relay with CAN transmission has not been filled in.

**Rectification:**

Complete the entry of the CAN data length.

**7.2.2.21 Error type 55 in line ... column ...:**

CAN data length isn't numeric.

**Description:**

In the CAN data length of a relay with CAN transmission, characters have been found which cannot be converted into a numeric value.

# Operating panel manual

---

**Rectification:**

Correct the entry of the CAN data length which must be decimal (0-8). If the RTR bit is to be set, 16 must be added.

**7.2.2.22 Error type 56 in line ... column ...:**

CAN-data must be entered hexadecimal.

**Description:**

In the CAN data of a relay with CAN-transmission, characters have been found which cannot be converted into a hexadecimal value. Permissible characters are 0..9 and A..F (capitalised!) or a variable.

**Rectification:**

Correct the entry of the CAN data. Write all values directly one after another without any hyphens.

**Example:** 131520 for D0=0x13, D1=0x15, D2=0x20

**7.2.2.23 Error type 57 in line ... column ...:**

LED No.: ... invalid

**Description:**

The entered LED number is not valid.

**Rectification:**

Correct the LED number.

**7.2.2.24 Error type 58 in line ... column ...:**

Output No.: ... invalid

**Description:**

The entered output number is invalid.

**Rectification:**

Correct your entry.

**7.2.2.25 Error type 59 in line ... column ...:**

Timer No.: ... invalid

**Description:**

The entered timer number is not correct.

**Rectification:**

Use 0 to 9 as timer number.

**7.2.2.26 Error type 60 in line ... column ...:**

System variable ... is read only.

**Description:**

The system variable you want to modify in the LD program must not be modified by LD.

**Rectification:**

Check the documentation on system variables whether this setting can be modified in any another way.

**7.2.2.27 Error type 61 in line ... column ...:**

Count must be above 0.

**Description:**

This error only occurs with the deactivation of menu pages.

**Rectification:**

Enter a value > 0.

**7.2.2.28 Error type 62 in line ... column ...:**

SDO: ... parameter error

**Description:**

A parameter of a SDO transmission/receive function has not or incorrectly been set.

**Rectification:**

Check and correct the settings of the SDO in accordance with the data listed in the Communication manual.

**7.2.2.29 Error type 63 in line ... column ...:**

Missing touch coordinate.

Page No. not numeric

Bitmap not found in page: ...

**Description:**

The touchscreen request has not been parameterised correctly.

**Rectification:**

Check the settings of the touchscreen request.

**7.2.2.30 Error type 100 in line ... column ...:**

Unknown compare type

**Description:**

Unknown compare type of the switch.

**Rectification:**

Adjust the compare type correctly.

**7.2.2.31 Error type 101 in line ... column ...:**

Unknown switch type

**Description:**

This error only occurs with switches which have not been parameterised yet.

**Rectification:**

Parameterise the switch.

# Operating panel manual

---

## 7.2.2.32 Error type 102 in line ... column ...:

Unknown switch condition

### **Description:**

This error only occurs if a switch has not been parameterised.

### **Rectification:**

Parameterise the switch.

## 7.2.2.33 Error type 103 in line ... column ...:

Not a numerical value: ...

### **Description:**

A numerical value must be entered, but wrong characters (letters or similar) have been found.

### **Rectification:**

Remove the impermissible characters.

## 7.2.2.34 Error type 104 in line ... column ...:

Value out of allowed range

### **Description:**

This error occurs if, e.g., you want to assign a 2 to an output (bit value)...

### **Rectification:**

Enter a permissible value.

## 7.2.2.35 Error type 105 in line ... column ...:

Unknown relay function

### **Description:**

This error only occurs if a relay has not been parameterised.

### **Rectification:**

Parameterise the indicated relay.

## 7.2.2.36 Error type 200 in line ... column ...:

Internal overflow.

### **Description:**

This error only occurs if your LD program exceeds the device capacities.

### **Rectification:**

Reduce your program scope or contact our support for optimisations.

## 7.2.2.37 Error type 201 in line ... column ...:

Internal error.

### **Description:**

This error should not occur.

### **Rectification:**

In this case, you can only contact our support. Save your entire ITE project into a new directory and send us all files of this directory (e-mail, disk, CD ...)

## 7.2.3 Error during the C-code compilation (only AT devices)

AT devices check the ladder diagram for syntactical and logic faults during the compilation process. Within this context, C-files are automatically created which are then translated by the C-compiler into the machine code of the processor. At this stage, the C-compiler must not generate any error messages. If error messages still occur, please contact our support. Save your entire ITE project to a new directory and send us all files of this directory (e-mail, disk, CD ...)

# Operating panel manual

## 8 Numberings

Key and LED numbers are frequently used in the LD editor. This also refers to input and output numbers of CAN modules. The following sections provide an overview of the element numbering.

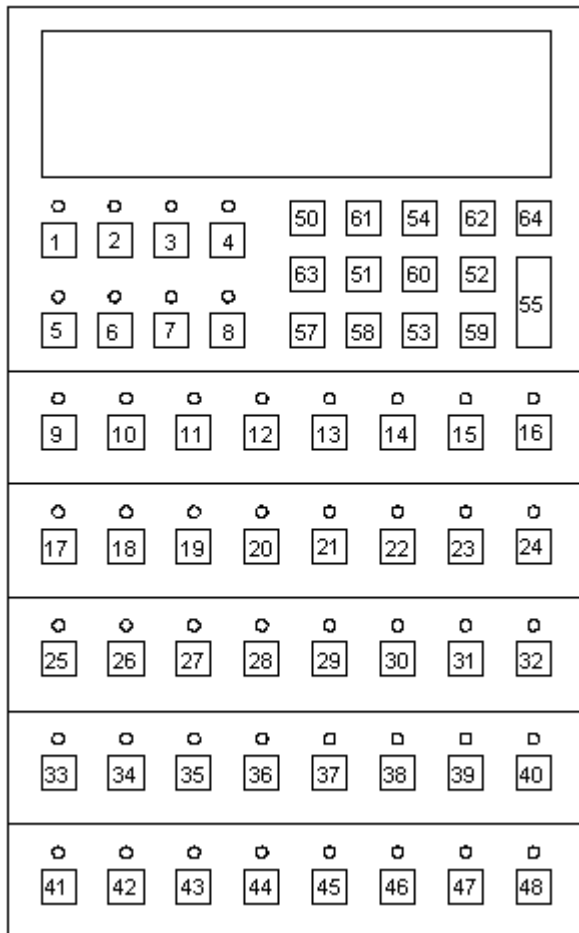
### 8.1 Keys and LEDs

Basically, LED and key numbers are identical and differ only in series.

Here, you always find the maximum series extension; keys which are not available in the device cannot be addressed. Irrespective of the extension, the numbers are always identical, particularly the numbers of the decimal block.

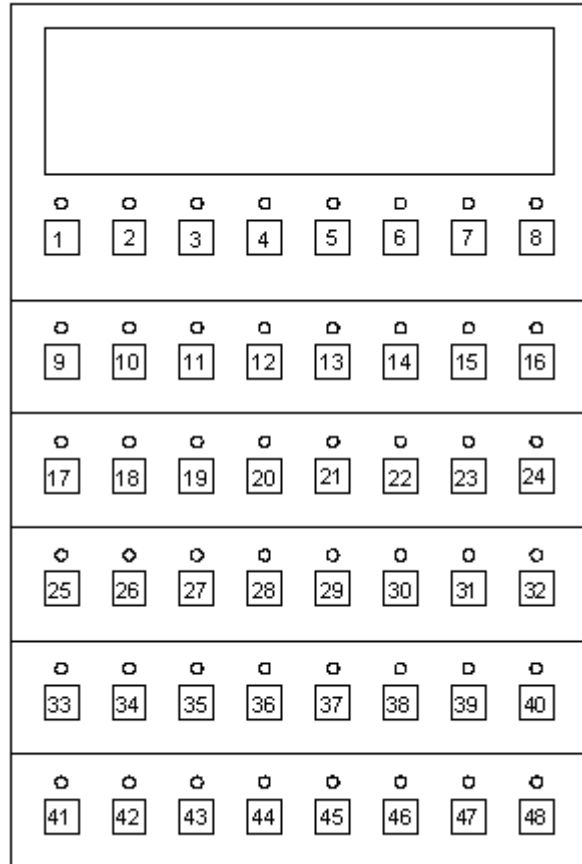
#### 8.1.1 ITS/AT 61 and 67 series

Keys and LEDs are numbered as follows:



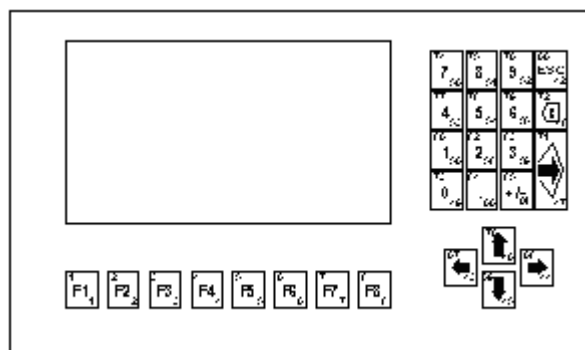
#### 8.1.2 ITS/AT 62/68/72/78 series

Keys and LEDs are numbered as follows:



#### 8.1.3 ITS/AT 71 and 77 series

Keys and LEDs are numbered as follows:



The LD key number can be found in the right bottom corner of the key.

# Operating panel manual

## 8.2 Inputs and outputs of CAN modules

The configurations of CAN modules are manifold. Therefore, we will provide you with a complete list of modules with all possible configurations.

Non-assigned inputs/outputs do not provide any defined results. As the space in drawings is limited, supplement the designations between "DI" or "DO" and the input/output number by the module number followed by a dot and prefix "@". DI4 then becomes @DI1.4 with module 1.

The number of the I/O points is limited to 48 per module. As each basic module can be equipped with 3 expansion modules, combinations which exceed 48 I/O may be possible. These combinations are not specified here.

### 8.2.1 Digital I/O modules of the GCM 200

We start with the GCM 200 module, which has 8 outputs in the basic version.

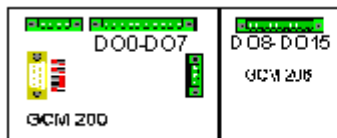
#### 8.2.1.1 GCM 200 basic version

8 outputs with the following numbering:



#### 8.2.1.2 GCM 200 + GCM 206

16 outputs with the following numbering:



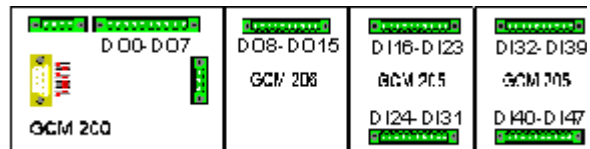
#### 8.2.1.3 GCM 200 + GCM 206 + GCM 205

16 outputs and 16 inputs as follows:



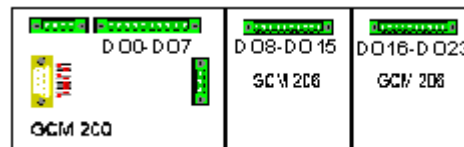
#### 8.2.1.4 GCM 200 + GCM 206 + 2x GCM 205

16 outputs and 32 inputs:



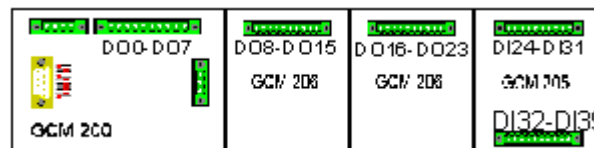
#### 8.2.1.5 GCM 200 + 2x GCM 206

24 outputs:



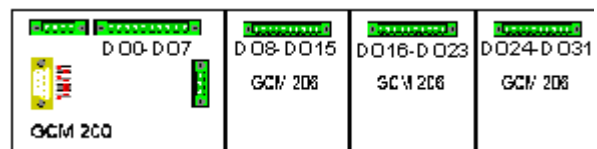
#### 8.2.1.6 GCM 200 + 2x GCM 206 + GCM 205

24 outputs and 16 inputs:



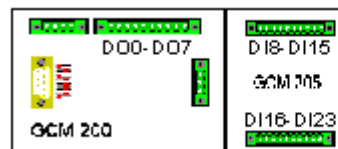
#### 8.2.1.7 GCM 200 + 3x GCM 206

32 outputs:



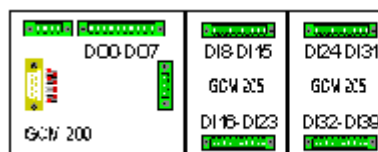
#### 8.2.1.8 GCM 200 + GCM 205

8 outputs and 16 inputs:



#### 8.2.1.9 GCM 200 + 2x GCM 205

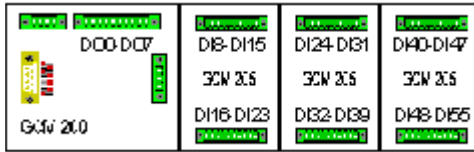
8 outputs and 32 inputs:



# Operating panel manual

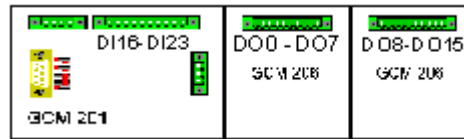
## 8.2.1.10 GCM 200 + 3x GCM 205

8 outputs and 48 inputs:



## 8.2.2.5 GCM 201 + 2x GCM 206

8 outputs and 16 inputs:



## 8.2.2 Digital I/O modules of the GCM 201

We continue with the GCM 201 module, which has 8 inputs in the basic version.

### 8.2.2.1 GCM 201 basic version

8 inputs with the following numbering:



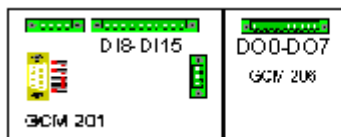
### 8.2.2.6 GCM 201 + 2x GCM 206 + GCM 205

16 outputs and 24 inputs:



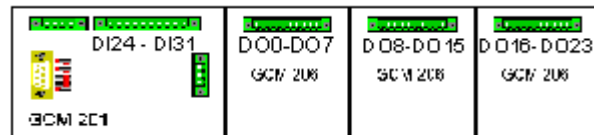
### 8.2.2.2 GCM 201 + GCM 206

8 inputs and 8 outputs:



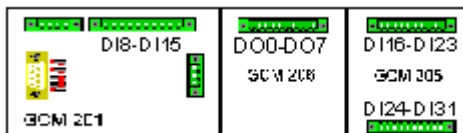
### 8.2.2.7 GCM 201 + 3x GCM 206

24 outputs and 8 inputs:



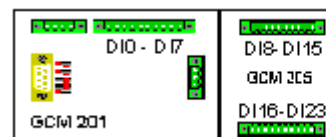
### 8.2.2.3 GCM 201 + GCM 206 + GCM 205

8 outputs and 24 inputs as follows:



### 8.2.2.8 GCM 201 + GCM 205

24 inputs:



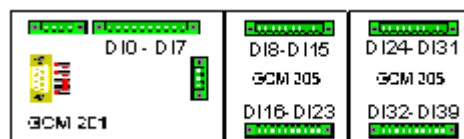
### 8.2.2.4 GCM 201 + GCM 206 + 2x GCM 205

8 outputs and 40 inputs:



### 8.2.2.9 GCM 201 + 2x GCM 205

40 inputs:



## 8.2.3 Digital I/O-modules of GCM 202

Finally, we would like to present modules on the basis of GCM 202, which have 8 inputs and 4 outputs in the basic version.

# Operating panel manual

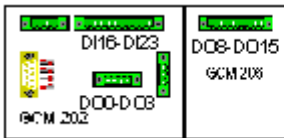
## 8.2.3.1 GCM 202 basic version

4 outputs and 8 inputs, which are numbered as follows:



## 8.2.3.2 GCM 202 + GCM 206

12 outputs and 8 inputs:



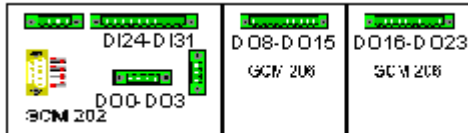
## 8.2.3.3 GCM 202 + GCM 206 + GCM 205

12 outputs and 24 inputs as follows:



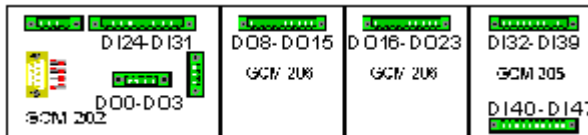
## 8.2.3.4 GCM 202 + 2x GCM 206

20 outputs and 8 inputs:



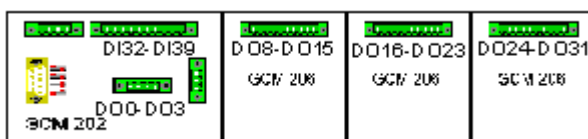
## 8.2.3.5 GCM 202 + 2x GCM 206 + GCM 205

20 outputs and 24 inputs:



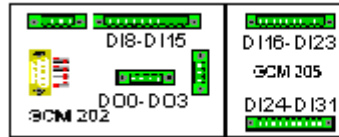
## 8.2.3.6 GCM 202 + 3x GCM 206

28 outputs and 8 inputs:



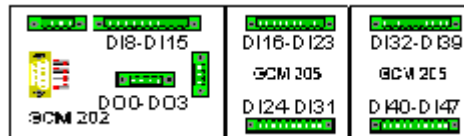
## 8.2.3.7 GCM 202 + GCM 205

4 outputs and 24 inputs:



## 8.2.3.8 GCM 202 + 2x GCM 205

4 outputs and 40 inputs:



## 8.2.4 Analogue input modules

Basically, these modules are equipped with 4 inputs. The input type has no impact on the numbering.

The inputs 1-4 are addressable as @Alx.0 to Alx.3 (x = module number 1-8).

## 8.2.5 Analogue output modules

Basically, these modules are equipped with 4 outputs. The output type has no impact on the numbering.

The outputs 1-4 are addressable as @Alx.0 to Alx.3 (x = module number 1-8).