

# OMRON

## Training manual

**NT4S-SF121B-E**

**NT4S-SF122B-E**

**NT4S-SF123B-E**

**NT15S-SF121B-E**

**NT18S-SF121B-E**

Version 1.0 created 03/09/99

This information has been reviewed for accuracy, but is not warranted to be error-free. Succeeding products and publications are subject to change without notification

# Table of Contents

1	General Information .....	5
1.1	Structure of this document.....	5
1.2	Conventions.....	5
1.3	Options .....	5
2	Creating a new project database .....	6
3	Creating a new mask .....	7
4	Inserting text elements into a mask .....	8
5	Inserting variable elements in a mask.....	9
5.1	Creating a numeric variable.....	10
5.2	Creating a <i>Selection text</i> variable.....	12
5.3	Creating a <i>Text list</i> .....	12
5.4	Creating an <i>Alphanumeric</i> variable .....	13
6	Creating an operator guidance.....	14
6.1	Operator guidance using the <i>Mask parameters</i> .....	14
6.2	Operator guidance using the <i>Soft keys</i> .....	15
6.3	Operator guidance using the <i>Global keys</i> .....	15
6.4	Operator guidance using the system variable <i>NewMask</i> .....	16
7	Creating a message system.....	17
7.1	Entering the message texts .....	17
7.2	Assigning message number and PLC address .....	18
7.3	Displaying messages in a mask .....	18
7.4	PLC address of the parallel message system .....	19
7.5	PLC address of the serial message system .....	19
8	Functions and assignment of the Poll area.....	20
9	General settings .....	21
9.1	Symbolic Addresses .....	21
9.2	Input variables .....	21
10	Parameterizing the protocol and interface X2 .....	22
11	Compiling and downloading the application .....	23
12	Recipes .....	24
12.1	Structure of a recipe .....	24
12.2	Creating a recipe .....	25
12.3	Outputting a recipe .....	26
12.4	System variables overview for recipes .....	26
12.5	Creating a recipe management .....	27

---

13 Selection image.....	28
13.1 Creating an Image list.....	28
13.2 Inserting an image from an existing file .....	28
13.3 Inserting a new image using another program (OLE).....	29
13.4 The Write-over Mode .....	29
13.5 The Display Mode.....	29
13.6 Creating a <i>Selection Image</i> variable.....	30
14 Bar chart.....	31
15 Tables .....	33
16 Appendix .....	34
16.1 Scaling of variables .....	34
16.2 System variables for recipe management .....	35
16.3 System variables for recipe data exchange.....	36
16.4 System variables for data set backup to a PC.....	37
16.5 System variables for data set printout .....	37
16.6 Types of variables .....	38

# 1 General Information

## 1.1 Structure of this document

Chapters 1 to 11 show the programmer how to create an entire project using the basic elements of this programming tool including the compiler and download function. Beginning with chapter 12 the programmer will become familiar with advanced programming features like dynamic images, bar charts, recipes and tables. For initial use it is recommended to work with the basic functions.

## 1.2 Conventions

- The names of tabs are represented in Capital characters
- Buttons are represented as **<characters>** enclosed between angle brackets
- (Menu) items are represented in ***bold italic black*** text format
- Inputs by the user are represented as [normal text] enclosed between brackets
- A sequence of actions and inputs is identified by the arrow character " | " and the font type Courier New
- The programming software for Omron operator terminals is herein referred to as **NTFK-ST**

## 1.3 Options

NTFK-ST provides a number of options which make it user-friendly and easy-to-use. We recommend the following settings:

- | Start NTFK-ST
- | Open the menu **Tools** from the menu bar and select the item **Options**
- | Select the **PROJECT MANAGEMENT** tab
- | Activate
  - the check box next to the item **New opens the appropriate editor**
  - the radio button **One** next to the menu item **Backup Databases**
  - the check box **Compress Databases**

All other options can be adapted to your individual requirements.

Within a NTFK-ST application (.tsw database) it is possible to enable the option **automatic download**. Selecting the **Enable Automatic Download** check box will cause the operating terminal to automatically detect and activate a download, without having to switch the user mode switch on the terminal. The download procedure can also be activated by a system variable. This variable can be edited by the operator. To prevent the terminal from being erased accidentally it is recommended to remove the system variable **IntEraseEprom** from your application.

- | Open or create a project database
- | Select the **PROJECT MANAGEMENT INFORMATION** tab
- | Select the menu item **System parameters**
- | Select the submenu **General parameters**
- | Activate the checkbox



To be able to use this function, a project with this setting must have been loaded into the terminal using the conventional procedure.

- | confirm your input with the **<OK>** button


## 2 Creating a new project database

Every NTFK-ST project consists of three tabs:

- **PROJECT MANAGEMENT INFORMATION**
- **LANGUAGE**
- **CONTROLLER**

| start NTFK-ST  
| open the **File** menu and select **New File**  
| enter a new name for a project, e.g [First\_Project]  
| select the type of terminal  
| select the size of application memory within your terminal (option)

or

| press the button   
| enter a new name for a project  
| select the type of terminal  
| select the size of the application memory within your terminal (option)

NTFK-ST automatically generates a new project with one tab in it  
**PROJECT MANAGEMENT INFORMATION**

| select the menu item **Language** from the tab  
**PROJECT MANAGEMENT INFORMATION**  
| push the button **<New>**  
| enter a new name for a language, for instance [E n g l i s h ]  
| activate the check box next to the new menu item

NTFK-ST has automatically generated the second tab  
**LANGUAGE**

| select the menu item **Controller** from the tab  
**PROJECT MANAGEMENT INFORMATION**  
| push the button **<New>**  
| select a protocol from the dialog, e.g. [Omron Host-Link]  
| activate the check box next to the new menu item

NTFK-ST has automatically generated the third tab  
**CONTROLLERS**

You have just created your first project database

NOTES :

---

---

---

---

### 3 Creating a new mask

The NTFK-ST project can include up to 9999 different masks (depending on the available application memory)

- | select the **LANGUAGE** tab
- | select the menu item **Masks**
- | press the button **<New>**
- | enter a name for the mask (only internal documentation)
- | confirm your input with the **<OK>** button

NTFK-ST automatically opens the mask editor. Several buttons in the extended toolbar are now activated. It contains toolbar icons for the following functions (from left to right).



Insert New Line (into lists and tables)



Delete Line (from lists and tables)



Zoom in (view)



Zoom out (view)



Message Text (Insertion of)



Selective Cursor (Selection of elements)



Static Text (Insertion of)



Variable (Insertion of)



Background Image ((Insertion of)



Recipe Field (Insertion of)



Table Field (Insertion of)



Message Field (Insertion of)

These buttons provide you with functions needed to create your first new mask.

#### NOTES :

---

---

---

---

## 4 Inserting text elements into a mask

Text can be inserted into a mask in two different ways. You can activate the button



### Static Text

- | position the cursor within the mask editor
- | start entering your text

Every text element is limited to the length of one row. To create multiple-line texts, multiple "one-row" texts must be created.

The button



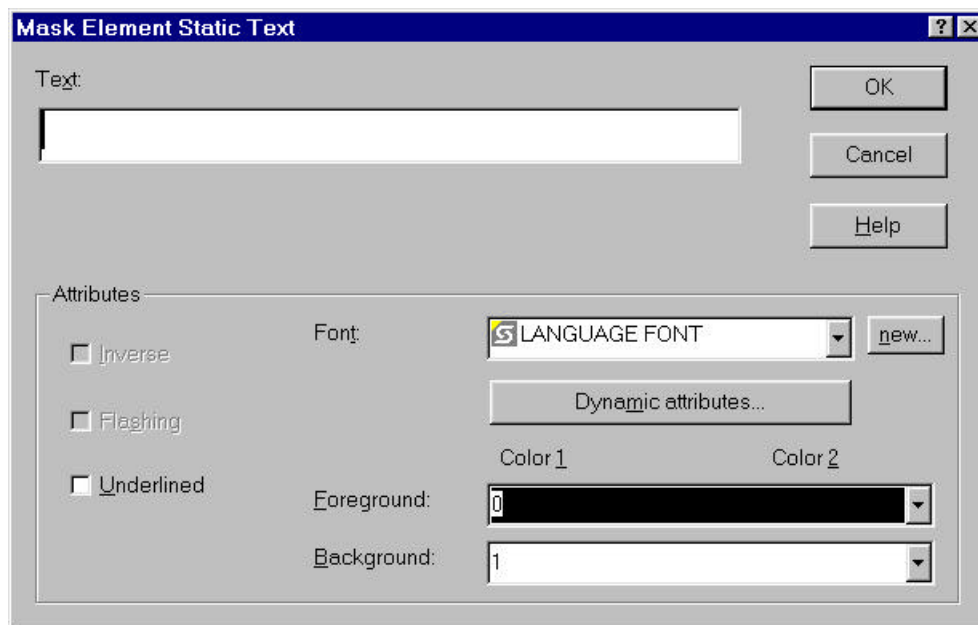
### Tool Semi-Graphic Character

allows you to select a semi-graphic character which will be inserted at the current text cursor position within a static text. Subsequently, the window will not disappear automatically, but will remain open to permit selection of further semi-graphic characters. To insert the previously selected semi-graphics character repeatedly, use the F5-key.

- | position the cursor within the mask editor
- | activate the button **<Tool Semi-Graphic Character>**
- | double-click the required character within the dialog

The second method is to use a cursor function.

Press the left mouse button, drag open a field with the desired length of the text and release the mouse button. The dialog below appears on the screen:



Enter your text and define the available attributes.  
This dialog always appears when you double-click on a text element.

- | confirm your input with the **<OK>** button

### NOTES :

---



---



---



---



## 5 Inserting variable elements in a mask

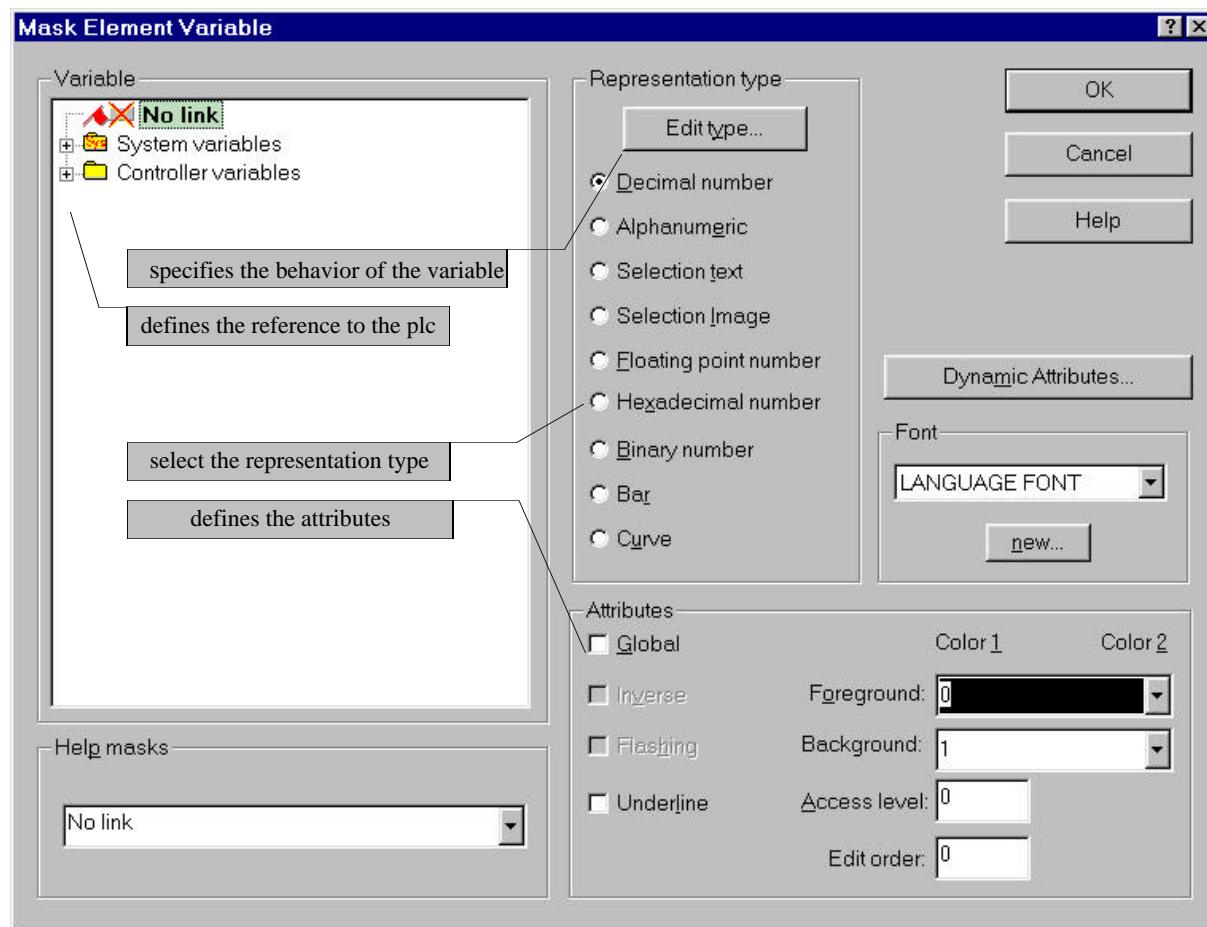
To create a new variable within a mask press the button



<New Variable>

- | position the cursor within the mask editor
- | press the left mouse button
- | drag open a field with the desired length of the variable
- | release the mouse button

The dialog below appears on the screen:



- | select the Representation type of the variable
- | press the <Edit Type> button and determine the behavior of the variable
- | define the attributes of this variable
- | define the reference to the PLC. To do so, highlight the **Controller variables** and press the right mouse button. In the dialog **New PLC variable** you can enter the variable name and address.
- | confirm your input with the <OK> button

**NOTES :**

---



---



---



---

### 5.1 Creating a numeric variable

I select the representation type

A horizontal button with a radio button icon and the text "Decimal number".

I press the button **<Edit type>**

I define the field type

A dialog box titled "Field type" containing three radio buttons: "Input" (selected), "Output", and "Cyclical" (checked).

In the Field Type field, define whether the decimal number is to be an input or an output variable and whether the value is to be read from the PLC at cyclic intervals.

I define the format of the variable

A dialog box titled "Format" with a label "[Unit in characters]". It contains two input fields: "Field length" with the value "5" and "Fractional digit" with the value "0". There are two checkboxes: "Only positive" (checked) and "Display leading zeros" (unchecked).

The representation format basically includes four specifications:

1. The field length, that is the total length including the sign and point.
2. The fractional digits (number of decimal places after the point), that is the digits to be displayed after the point (e.g. currency values have two digits after the point).
3. The option of limiting decimal values to positive values.
4. In the case of small numbers (requiring only a small number of digits), the option to fill the digits not used with zeros (leading zeros).

I define the limits of the variable

A dialog box titled "Limits" with two input fields: "Lower limit" with the value "-21474836" and "Upper limit" with the value "21474836".

The upper limit of a variable determines the maximum value that can be entered for an input variable. Entering a value that exceeds the upper limit or lower limit can cause an error message to be generated.

**NOTES :**

---



---

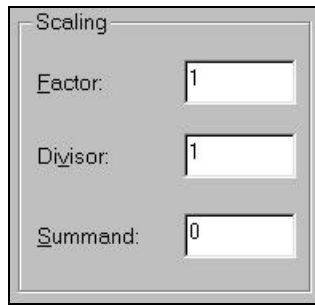


---



---

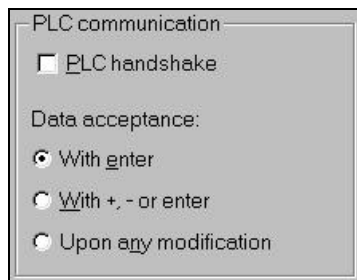
I define the scaling of the variable



The image shows a dialog box titled "Scaling". It contains three input fields: "Factor" with the value "1", "Divisor" with the value "1", and "Summand" with the value "0".

An output value is multiplied by the factor, divided by the divisor and added to the summand before being displayed on the operating terminal. This allows the values transmitted from the PLC to be scaled. (see also Appendix)

I define the "PLC communication" dialog for the variable

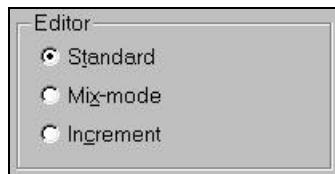


The image shows a dialog box titled "PLC communication". It has a checkbox for "PLC handshake" which is unchecked. Below it, under "Data acceptance:", there are three radio button options: "With enter" (selected), "With +, - or enter", and "Upon any modification".

To be able to transmit the values of the variable to the PLC, specify whether a value is to be transmitted only after the Enter-key is pressed, after the PLUS-key, MINUS-key or Enter-key is pressed or automatically upon every modification. The PLC handshake procedure can be used for this process, if desired.

**When the increment editor is selected, the variable can not be scaled.**

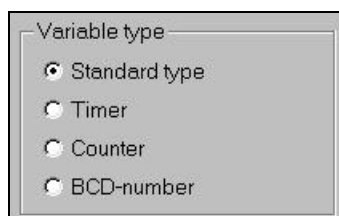
I define the type of editor of the variable



The image shows a dialog box titled "Editor". It contains three radio button options: "Standard" (selected), "Mix-mode", and "Increment".

There are two ways of entering values for an input variable: using the operating terminal's numerical keys (which corresponds to the standard editor) or using the operating terminal's PLUS-key and MINUS-key (which corresponds to the incremental editor). The mix mode editor supports simultaneous use of both editor types.

I define the variable type



The image shows a dialog box titled "Variable type". It contains four radio button options: "Standard type" (selected), "Timer", "Counter", and "BCD-number".

The **Variable type** area allows you to use the variable as a standard type, timer, counter or BCD number.

**NOTES :**

Four horizontal lines for taking notes.

### 5.2 Creating a Selection text variable

| select the representation type

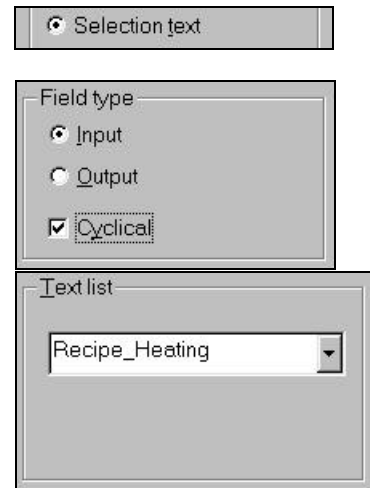
| press the button **<Edit type>**

| define the field type

| assign a textlist

If a text list does not exist yet, you can insert a name and create the text list in the next step. These settings determine the PLC address, the behavior and the attributes of the variable. The assignment between text and PLC value is programmed in a text list.

| confirm your inputs with the **<OK>** button



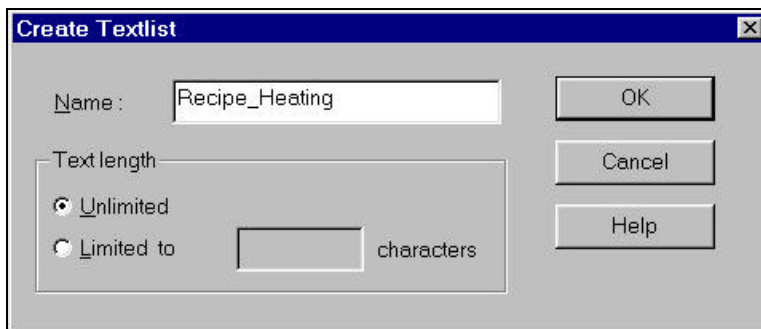
### 5.3 Creating a Text list

| select the **LANGUAGE** tab

| select the menu item **Text list**

| press the button **<New>**

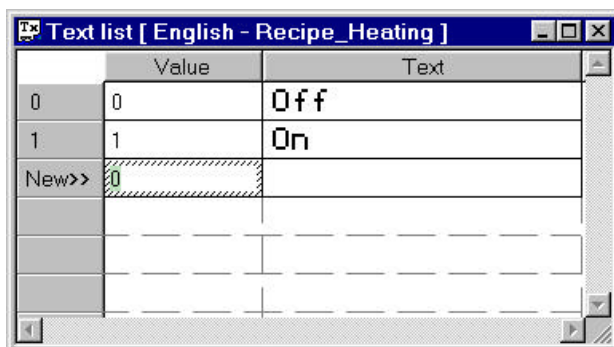
| the dialog shown below is displayed:



| enter a name for the new **Text list**, e.g. [Recipe\_Heating]

| confirm your input with the **<OK>** button

| the next dialog allows you to assign a text to a value



| confirm your input with the **<Cross>** button

**NOTES :**

---



---



---

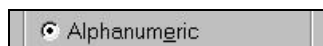


---

## 5.4 Creating an *Alphanumeric* variable

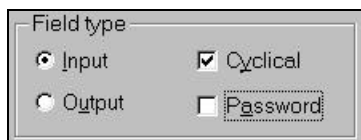
Alphanumerical variables permit the display of letter/number combinations. The max. length of the variable is limited to the size of the display of the terminal. Every character of this variable corresponds to a byte address within the controller. The character position and PLC address is assigned in a linear ascending order.

| select the representation type



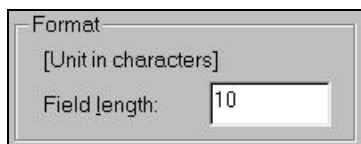
| press the button **<Edit type>**

| define the field type



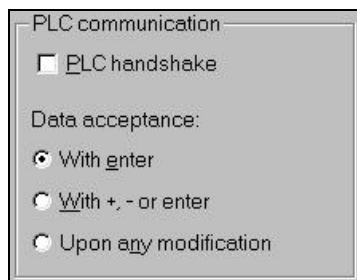
By selecting the setting Password, Xs are displayed instead of the entered password. If a user-defined font is used, the character with the corresponding index is used.

| define the variable format



For the variable format, specify the field length. Every character represents a byte address within the controller. The assignment between character position and PLC address is done in a linear ascending order.

| define the type of PLC communication



To be able to transmit the values of the variable to the PLC, specify whether a value is to be transmitted only after the Enter-key is pressed, after the PLUS-key, MINUS-key or Enter-key is pressed or automatically upon every modification.

| confirm your inputs with the **<OK>** button

| define the PLC address of this variable

| confirm your inputs with the **<OK>** button again

### NOTES :

---



---



---



---

## 6 Creating an operator guidance

To create an operator guidance, you can use function keys, soft keys as well as the system variable NewMask. By means of the hexcode 8000<sub>H</sub> plus the mask number (mask number 10 corresponds to the Hexcode 800A<sub>H</sub>) within the serial message channel (Poll area) a mask can be called by the PLC.

### 6.1 Operator guidance using the *Mask parameters*

- | select the Language tab
- | select the menu item **Masks**
- | select the mask to be parameterized
- | press the button **<Parameters>**
- | the dialog shown below is displayed:

Parameters Mask: English - Error messages

Number: 8      Access level: 0      OK      Cancel

Mask selection:

Home	Main-Mask
Left	No link
Right	Parametrization
Top	No link
Bottom	Process variables
Help mask	Default

Background color:      Select..

Options:

- Variables management topdown
- Automatic data release
- Reset password

- | enter your links to create an operator guidance
- | confirm your input with the **<OK>** button

#### NOTES :

---

---

---

---

### 6.2 Operator guidance using the *Soft keys*

- | select the **LANGUAGE** tab
- | select the menu item **Masks**
- | press the button **<Function keys>**
- | the dialog shown below is displayed:

	Mask	Press variable	Action	Release variable	Action
F1			1		0
F2	main mask		1		0
F3			1		0
F4	parameterization		1		0
F5			1		0
F6	process variables		1		0
F7			1		0
F8			1		0
F9			1		0
F10			1		0
F11			1		0
F12			1		0

- | select the column **Mask**
- | press the right mouse button and **Edit** this item
- | select a mask from the displayed list
- | confirm your input with the button

### 6.3 Operator guidance using the *Global keys*

- | select the **LANGUAGE** tab
- | select the menu item **Global keys**
- | press the button **<Edit>**
- | a similar dialog as shown above is displayed

	Mask	Press variable	Action	Release variable	Action	Output
Up			1		0	
Down			1		0	
Left			1		0	
Right			1		0	
Home			1		0	
F1	main mask		1		0	
F2			1		0	
F3			1		0	
F4			1		0	
F5			1		0	

In addition to the soft key dialog above you can also assign a function code to the cursor keys. A distinction is made between the validity of global keys and soft keys. A soft key is a mask specific definition. A global key assignment is valid in every mask. If both definitions are assigned to one function key, the mask specific assignment takes precedence over the global definition.

**NOTES :**

---



---



---

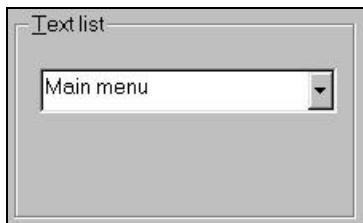


---

### 6.4 Operator guidance using the system variable *NewMask*

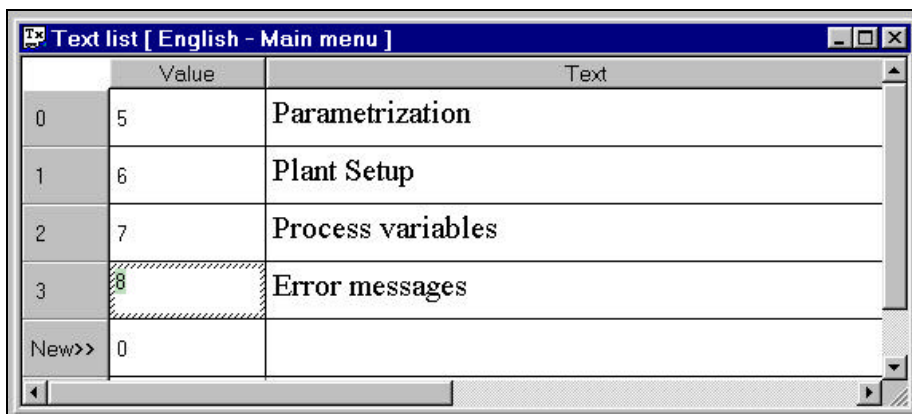
- | select the **LANGUAGE** tab
- | select the menu item **Masks**
- | open a mask with the **<Edit>** button to enter the variable
- | the mask editor is displayed
- | create a new variable (see chapter above)

- | select the representation type  Selection text
- | assign a text list



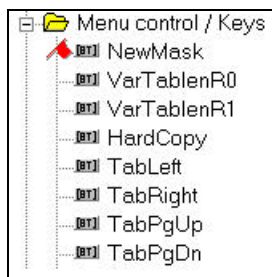
If a text list already exists, select it from this dialog. You can also enter a name in this dialog and create the text list after completing the variable dialog.

- | select the **LANGUAGE** tab
- | select the menu item **Text list**
- | to edit the assigned text list, press the **<Edit>** button
- | the text list editor is displayed
- | enter the mask number into the column **Value**
- | enter the text which should be displayed on the terminal



If a text list is assigned to a variable which is referenced to the system variable **NewMask**, the values are evaluated as a mask number. If the selected mask number exists, the assigned mask will be displayed.

- | assign the system variable **NewMask**



- | confirm your input with the **<OK>** button

**NOTES :**

---



---



---



---



## 7 Creating a message system

The programming system provides you with two different message systems:

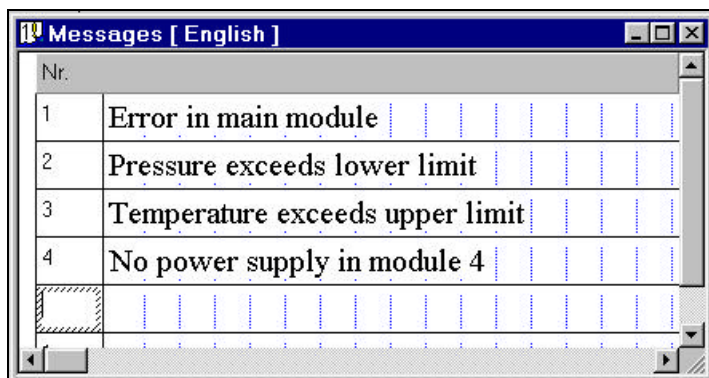
- A parallel message system which is designed as a status message system. Every message is assigned to a bit information within a linearly addressed PLC memory. As long as the bit is set to 1, the message will be displayed by the terminal.
- A serial message system which is designed as a static message system. Every message is stored in an internal RAM-memory. The messages can be deleted by the operator or by the PLC.

Both message systems display the same error messages but with different behavior.

The lower the message number the higher the priority of the message. Messages can be sorted by time and priority. The sort criteria can be altered by the operator. Therefore, the system provides several system variables for both message systems.

### 7.1 Entering the message texts

- | select the **LANGUAGE** tab
- | select the menu item **Messages**
- | press the **<Edit>** button to enter messages
- | the message editor is displayed
- | enter the message number in the column **No.**
- | enter the message text within the text field



An interface between both message system is provided. Every message can be assigned two attributes:

- | select the text field of the message editor
- | press the right mouse button
- | a dialog with the following attributes is displayed

<input type="checkbox"/>	Enter into message memory when mask is called
<input checked="" type="checkbox"/>	Store state message

#### **Enter into message memory when mask is called**

If, when a mask is called (change of mask from the controller), the message with the same message number is to be entered into the message memory, select the corresponding check box at the bottom of the window separately for each message.

#### **Store status message**

is the interface between serial and parallel message system. Every time a parallel message is activated it is also evaluated by the serial message system. So the operator has both, a current status display and a error history about the last processes that have occurred.

- | confirm your inputs by pressing the button

#### NOTES :

---



---



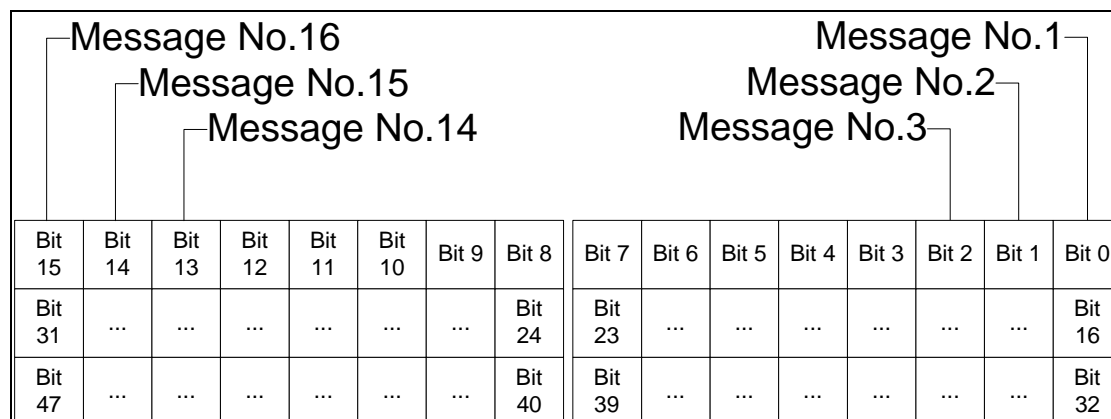
---



---

### 7.2 Assigning message number and PLC address

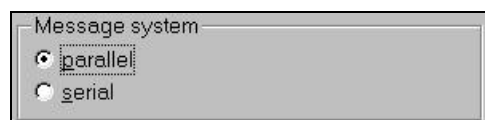
Parallel message system means that, for example, 16 bits are provided for 16 messages. These 16 bits are read in and evaluated simultaneously (in parallel). If a bit is set (logical 1), a corresponding message is activated. These 16 bits require a size of 2 bytes in the parallel message system. With the address of the variable for status messages (parallel messages), the starting address of a memory area is specified.



### 7.3 Displaying messages in a mask

Parallel messages can be output by means of a message area within a mask. A new mask has to be created for this purpose.

- | select the **LANGUAGE** tab
- | select the menu item **Mask**
- | press the **<New>** button to create a new mask
- | enter a name for this mask, for instance [error messages]
- | the mask editor is displayed
- | press the button
- | drag open a field within the mask using the cursor
- | the dialog **Message field parameter** is displayed
- | select the **parallel message system**



- | define the height of the message field and the max. number of lines per message



- | confirm your input with the **<OK>** button
- | create the operator guidance within the mask parameters dialog or soft keys dialog.

**NOTES :**

---



---



---



---

## 7.4 PLC address of the parallel message system

The parameters have to be entered within the system parameters

- | select the **PROJECT MANAGEMENT INFORMATION** tab
- | select the menu item **System parameters**
- | press the **<Edit>** button to parameterize the message system
- | the parameter dialog is displayed

The parameter **Size**

determines the number of messages that can be addressed. A size of 8 bytes allows the assignment of 64 messages.

The parameter **Poll time**

determines the polling time within which the terminal is to read out the status message data area from the controller. Values from 0 to 25.5 seconds can be specified for the polling time.

For the parameter **Variable for status messages**

enter a variable for the starting address of the data area where the messages are stored in the PLC in a bit-coded format.

## 7.5 PLC address of the serial message system

A 2 byte compartment is used in the cyclic poll area for the transmission of serial messages. The byte order depends on the selected data type of the poll area (see poll area). The PLC stores a 16 bit message number in this transmission compartment. The operating terminal polls the entire poll area of the PLC at cyclic intervals and transmits the serial message in the process. Upon detecting a message (message number >0), this message is stored in the internal message memory of the operating terminal and the compartment in the PLC is reset to zero. The value 0 in the compartment indicates to the PLC that the message has been fetched by the terminal. The polling time of the cyclic data area can be selected as required.

- | select the **PROJECT MANAGEMENT INFORMATION** tab
- | select the menu item **System parameters**
- | select the submenu **Poll area**
- | press the **<Edit>** button to parameterize the **Poll area**
- | the parameter dialog is displayed

The same procedure is used to address external masks and message masks. Whenever the number transmitted corresponds to a mask number, this mask is displayed. If a mask and a message text exist for this number, the mask (message mask, full-page fault message text) and the associated message text is entered into the message memory.

Make sure that the message number is always entered in the serial data compartment as a 16 bit command. As the result of asynchronous processing of some data transfer protocols, evaluation of the message number may lead to problems if the message number has been entered with single-byte commands.

It is recommended that the **Poll time** does not exceed a lower limit of 0.7 seconds. **A poll time below this value can cause a communication jam. In this case, process variables may not be updated as fast as possible.**

### NOTES :

---



---



---



---

## 8 Functions and assignment of the Poll area

The poll area is a data area that enables a data transfer between the operating terminal and the PLC. The data transfer is executed periodically (data polling). During this process, the data that start at the address for the poll area and that correspond to the length of the poll area are transferred.

The poll area can be byte-structured or word-structured.

W+0	Not assigned							DDR	LF	DP	RA	EDR	Not assigned							
W+1	EM	Serial message channel High-Byte										Serial message channel Low-Byte								
W+2	LED 1 ON OFF	LED 1 flashing	LED 2 ON OFF	LED 2 flashing	LED 3 ON OFF	LED 3 flashing	LED 4 ON OFF	LED 4 flashing	LED 5 ON OFF	LED 5 flashing	LED 6 ON OFF	LED 6 flashing	LED 7 ON OFF	LED 7 flashing	LED 8 ON OFF	LED 8 flashing				
W+3	LED 9 ON OFF	LED 9 flashing	LED 10 ON OFF	LED 10 flashing	LED 11 ON OFF	LED 11 flashing	LED 12 ON OFF	LED 12 flashing	LED 13 ON OFF	LED 13 flashing	LED 14 ON OFF	LED 14 flashing	LED 15 ON OFF	LED 15 flashing	LED 16 ON OFF	LED 16 flashing				
W+4	LED 17 ein aus	LED 17 flashing	LED 18 ein aus	LED 18 flashing	LED 19 ein aus	LED 19 flashing	LED 20 ein aus	LED 20 flashing	LED 21 ON OFF	LED 21 flashing	LED 22 ON OFF	LED 22 flashing	LED 23 ON OFF	LED 23 flashing	LED 24 ON OFF	LED 24 flashing				
W+5	LED 25 ON OFF	LED 25 flashing	LED 26 ON OFF	LED 26 flashing	LED 27 ON OFF	LED 27 flashing	LED 28 ON OFF	LED 28 flashing	LED 29 ON OFF	LED 29 flashing	LED 30 ON OFF	LED 30 flashing	LED 31 ON OFF	LED 31 flashing	LED 32 ON OFF	LED 32 flashing				
W+6	LED 33 ON OFF	LED 33 flashing	LED 34 ON OFF	LED 34 flashing	LED 35 ON OFF	LED 35 flashing	LED 36 ON OFF	LED 36 flashing	LED 37 ON OFF	LED 37 flashing	LED 38 ON OFF	LED 38 flashing	LED 39 ON OFF	LED 39 flashing	LED 40 ON OFF	LED 40 flashing				
W+7	LED 41 ON OFF	LED 41 flashing	LED 42 ON OFF	LED 42 flashing	LED 43 ON OFF	LED 43 flashing	LED 44 ON OFF	LED 44 flashing	LED 45 ON OFF	LED 45 flashing	LED 46 ON OFF	LED 46 flashing	LED 47 ON OFF	LED 47 flashing	LED 48 ON OFF	LED 48 flashing				

### EDR

Once the controller has found that the editing request bit in the <Read Co-ordination Byte> is set to logical 1, it can release editing on the operating terminal by setting the bit for external data release in the <Write Co-ordination Byte> to logical 1.

### RA

Once the controller has read the refresh request bit in the <Write Co-ordination Byte> and found that it is set to logical 1, it can read in the modified variable value. Subsequently, it can confirm the execution of the request to the operating terminal by setting the refresh acknowledgement bit (Write Co-ordination Byte) to logical 1.

### DP

By constantly querying the current mask number, the controller is able to recognize whenever a password-protected mask has been displayed on the operating terminal. To ensure reactivation of password protection, the <Delete Password> bit in the <write co-ordination byte> must be set to logical 1 after exiting of the protected mask.

### LF

In some communication protocols, it is not possible to check the operability of the interface on the controller end. This is why the function of the <Liveness Flag> was introduced. This is a simple functionality that has proven to be very effective in practice.

Whenever the PLC wishes to know whether a connection still exists, it writes a logical 1, and later a logical 0, into bit 3 of the <Write Co-ordination Byte>.

The operating terminal constantly monitors the liveness flag in the <Write Co-ordination Byte> and compares it against the status of the liveness flag in the <Read Co-ordination Byte>. As soon as a discrepancy occurs, the operating terminal copies bit 3 from the <Write Co-ordination Byte> to the <read co-ordination byte>.

The controller now has the task of also checking, within a timeout, whether both statuses are equal.

#### Important:

The data transfer and polling times must be taken into account when defining the timeout.

### DDR

The PLC can control the download of a data set by means of the Data Set Download Release bit in the Write Coordination Byte.

#### NOTES :

---



---



---



---

## 9 General settings

- | select the **PROJECT MANAGEMENT INFORMATION** tab
- | select the menu item **System parameters**
- | select the submenu item **General parameters**
- | press the **<Edit>** button to parameterize the settings

### 9.1 Symbolic Addresses

The **image of the mask number** corresponds to the number of the currently displayed mask. The operating terminal writes this number to the defined variable on every change of a mask.

For the image of the mask number, a variable must be defined with a word address in the variable list. This is because 16 bit variables are transmitted.

In the standard mode, the **image of the mode selector switch** is transferred to the controller after initialization, if a variable has been defined for this purpose.

The designation **Read Coordination Byte** means that the controller reads this byte.

The Read Coordination Byte is only written by the operating terminal. This byte is used for the handshake and for data coordination with the controller. For this purpose, the operating terminal reports its current status to the controller using the symbolic name entered here. The individual bits are independent of one another

The structure is illustrated below:

When tabular data are edited on the terminal, the current position of the cursor in the table is transmitted to the PLC as the **table index**. The variable is written to the controller at cyclic intervals.

One bit in a string of bytes is available for each key of the operating terminal to indicate the status of the key. If this bit is set to logic 1, this indicates that the associated key is pressed. Once the key is released, the bit is reset to logic 0. Before the **keyboard image** can be read out, the request code 7FFCh must be written to the cyclic poll area. This causes the operating terminal to write the current keyboard image to the defined data field of the controller.

Since the number of keys varies with the operating terminal, each terminal has its own keyboard image.

### 9.2 Input variables

If you want the inverse representation to be used for input variables while you are editing these variables, select the check box in the Input Variables area.

If you want the cursor to be automatically advanced to the next input variable after you press <Enter>, thereby allowing you to input data in a mask more quickly, select the <On Pressing Enter, Select Next Input Variable> check box.

#### NOTES :

---



---



---



---

## 10 Parameterizing the protocol and interface X2

- | select the **CONTROLLER** tab
- | select the menu item **Communication parameters**
- | press the **<Edit>** button to parameterize the dialog



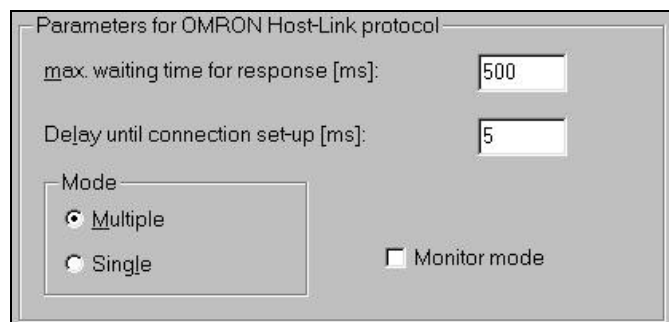
Parameters for interface X2

Baud rate: 9600      Data bits: 7

Parity: even      Stop bits: 2

Handshake: no handshake

The default settings should only be modified when the Omron controller settings do not correspond to these values.



Parameters for OMRON HostLink protocol

max. waiting time for response [ms]: 500

Delay until connection set-up [ms]: 5

Mode

Multiple

Single

Monitor mode

### **Max.waiting time for response[ms]**

specifies a time limit within which the controller has to send a response telegram to the terminal after receiving a request message. When the response time of the controller exceeds this value, a time out error is displayed on the terminal.

### **Delay until connection set-up[ms]**

after a communication error the terminal tries to set-up a new connection. In this case, the terminal tries to establish a connection every 5 seconds.

### **Multiple mode**

can be used for all Omron controllers and supports the communication between one terminal and several controllers.

### **Single mode**

is only supported by a few Omron controllers. This mode only allows the communication between one terminal and one controller.

### **Monitor mode**

when this check box is activated the terminal has read and write access to the controller variables. Without the monitor mode the terminal is an output device without write access to the controller.

### NOTES :

---



---



---



---

## 11 Compiling and downloading the application

- | select the **PROJECT MANAGEMENT INFORMATION** tab
- | select the menu item **Project**
- | press the **<New>** button to create a new project
- | enter a name for the new project
- | the project dialog is displayed



Within this dialog all created languages and controllers are listed in the window. Every database can consist of several languages and controllers.



- | select a language within the project management
- | copy one language and one controller into the project window with the button



In the project window the **Startup Language** is automatically set to the first language in the list.



Now this project can be compiled and transmitted to the terminal.

- | connect the download cable to the PC and the terminal
- | set the terminal to the download mode
- | press the button  to start the compiler
- | then press the button  to start the transmission

### NOTES :

---



---



---



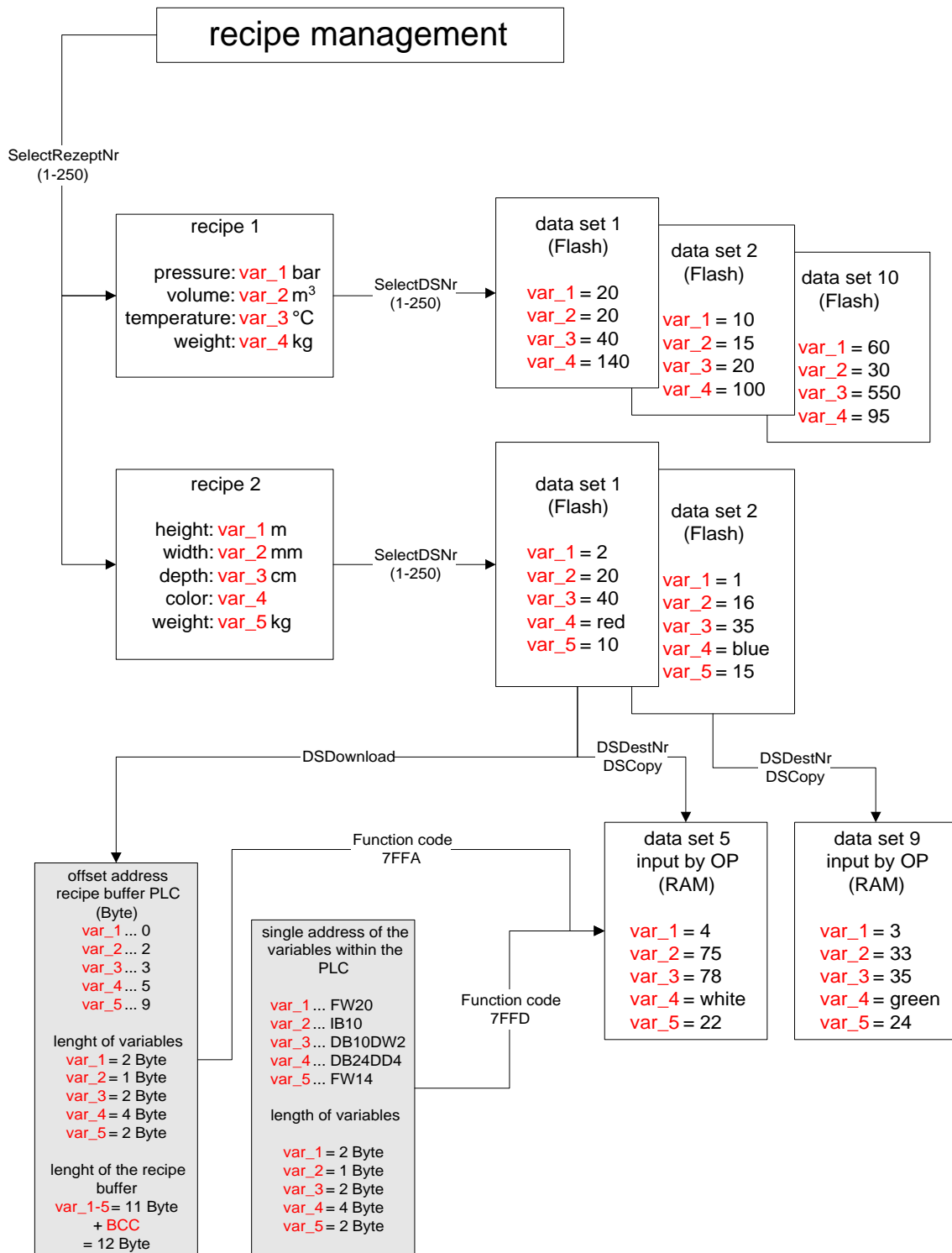
---

## 12 Recipes

Recipes provide the programmer with the possibility to store data within the terminal - not within the PLC. The operator can create a new data set and delete or modify an existing data set. The data exchange between terminal and PLC can be activated by the operator as well as by the PLC. Data can be sent from the terminal to the PLC and vice versa.

A backup system allows the operator to store the data set on a connected computer. The data set can also be printed.

### 12.1 Structure of a recipe



**NOTES :**

---



---



---



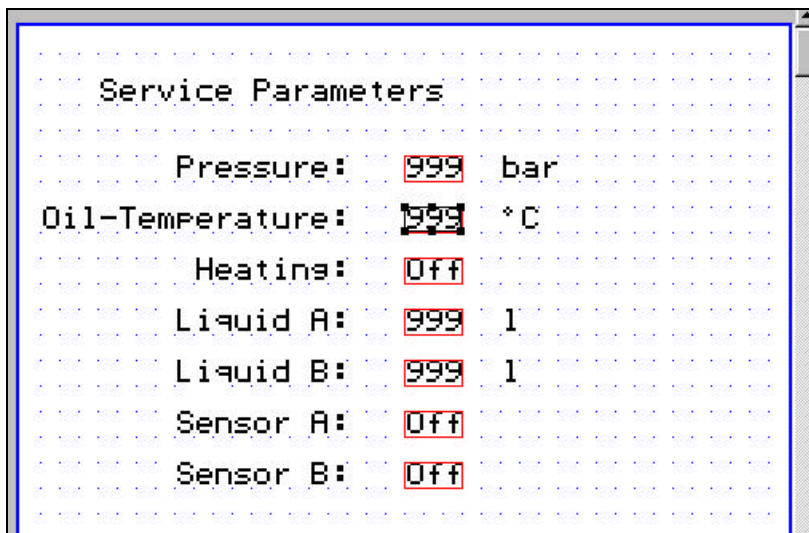
---



## 12.2 Creating a recipe

- | select the **LANGUAGE** tab
- | select the menu item **Recipes**
- | press the **<New>** button to create a new recipe
- | enter a name for this new recipe
- | the recipe editor is displayed

This window consists of two sections. The left section displays a view similar to that of a mask. This section allows you to enter the static texts (e.g. item names and units) and variables specific to a particular recipe.



The values for the variables, which are stored in a data set, can be entered in the right section of this window. Therefore, select a variable within the recipe (left section of the window) and enter the data set values in the right section. For every variable you select on the left side the assigned values are display in the right section. To enter a new data set, select the row **New** and enter a name and a value for this data set. The number of data set has to be the same for all variables within one recipe.

	Data set name	Data set no.	Variable value
0	DATASET 1	1	50
1	DATASET 1	2	60
New>>		0	

The creation of the data set is controller-specific. Therefore, you will need to select the name of a controller from a list at the top of the view. Here, you will also need to assign an offset to every variable of a data set.



The offset of a variable determines the position and length of variables within the recipe buffer. This buffer is the destination address range for the download of a recipe to the PLC. The offset of the variables does not have to be addressed in a linear order. Unused ranges between the offsets will, however, be overwritten with random values from the RAM-memory of the terminal when executing a download. The programmer has to take care to ensure that no values of the PLC are damaged by a recipe download.

**NOTES :**

---



---




---



---

### 12.3 Outputting a recipe


- | select the **LANGUAGE** tab
- | select the menu item **Mask**
- | press the **<New>** button to create a new recipe
- | enter a name for this new recipe, for instance [Recipe 1]
- | the mask editor is displayed

- | press the recipe button 
- | drag open a field using the mouse cursor
- | release the mouse button
- | the dialog **Recipe Field Parameters** is displayed



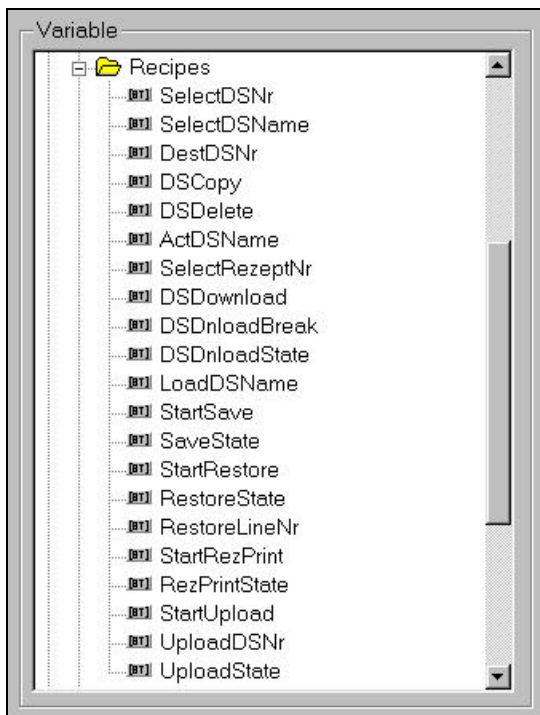
From the dialog box select the name of the recipe which should be output by default when displaying the mask. The **Recipe height** determines the number of rows in the display which are reserved for the output of the recipe. A recipe with more rows than the value entered here can be scrolled up and down using a cursor function.

- | confirm your inputs with the **<OK>** button
- | link the mask to another (operator guidance)

- | save your settings with the  button

### 12.4 System variables overview for recipes

A large number of system variables are available to work with recipes. These variables allow the programmer to implement an easy-to-use operator interface.



SelectDSNr	Active Data Set Number
SelectDSName	Active Data Set Name, Variant 1
DestDSNr	Destination Data Set Number
DSCopy	Copies a Data Set
DSDDelete	Deletes a Data Set
ActDSName	Active Data Set Name, Variant 2
SelectRezeptNr	Active Recipe Number
DSDownload	Transmits a Data Set to Controller
DSDnloadBreak	Stops Data Set Transmission
DSDnloadState	Monitors Data Set Transmission to Controller
LoadDSName	Last Data Set Transferred
StartSave	Transmits Data Set to PC
SaveState	Monitors Data Set Transmission to PC
StartRestore	Transmits a Data Set from PC to Terminal
RestoreState	Monitors Data Set Transmission to Terminal
RestoreLineNr	Line Number of Data Set File, to Terminal
StartRezPrint	Prints a Data Set
RezPrintState	Monitors Data Set Printout
StartUpload	Reads a Data Set from Controller
UploadDSNr	Data Set Number for Data Set Read from Controller
UploadState	Monitors Data Set Read Operation

**NOTES :**

---



---



---



---

### 12.5 Creating a recipe management

To edit, alter, copy and delete a data set, the operator has to be provided with the corresponding system variables.

In this exercise you will include system variables that allow you to edit, copy and delete data sets. Further functions for working with recipes can be inserted as required.

As shown in the diagram **Structure of a recipe** (Chapter 12.1), the programmed data sets are stored in the flash application memory. This data set can not be altered by the operator.

To edit data, the flash data set first has to be copied to the RAM. To do so, the operator has to carry out the following steps:

1. Select a recipe

System variable: **SelectRezeptNr**

This variable contains the active recipe. The variable can be modified in the recipe mask or any other mask.

2. Insert the source which has to be copied

System variable: **SelectDSNr**

This variable contains the number of the active data set.

3. Insert the destination data set number for the copy process

System variable: **DestDSNr**

When copying data sets, this variable contains the number of the destination data set.

4. Start the copy process

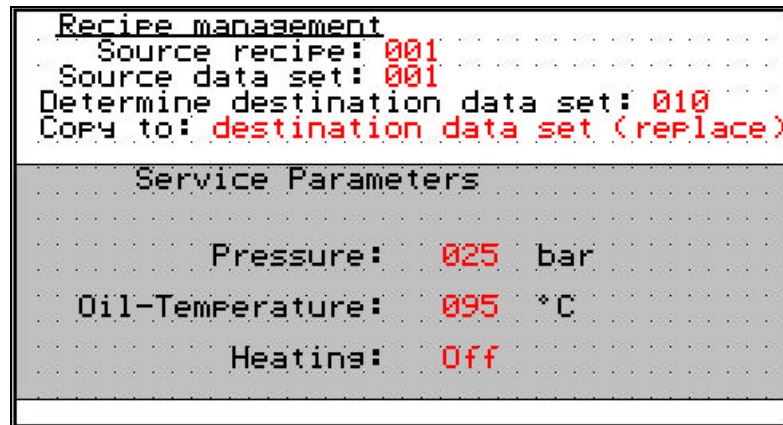
System variable: **DSCopy**

This variable can be used to copy the active data set to the destination specified in DestDSNr.

Data type:	Numeric
Editor:	Selection text, decimal number, softkey, function key
Output:	---
Possible values:	(0) Normal position
	(1) Copy to destination in system variable DestDSNr
	(2) Copying and automatic search for a free data set
	(3) Copy to destination in system variable DestDSNr. Overwrites an existing data set.

To ensure an easy-to-use operator guidance, these variables should be provided within the recipe mask.

The procedure to create a system variable is nearly the same as that to create a PLC variable. The only difference is the address. Instead of creating a PLC address, the variable has to be assigned to a system function. See an example of a recipe management mask below:



Additional service and backup functions can be provided in so-called service masks. Please configure your recipe management and services as required.

NOTES :

---



---



---



---

## 13 Selection image

Numeric values can be displayed in the familiar manner and as graphic objects.

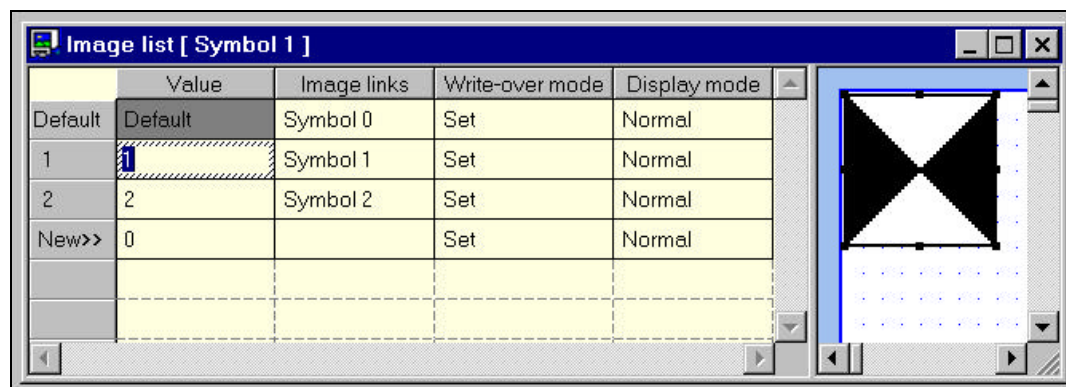
The variable Selection Image allows you to display images from an image list instead of displaying values. The images must be predefined in an image list. The variable will then be linked to this image list. To do so, select an image list from the Image List area.

### 13.1 Creating an Image list

An image list is a table containing the assignments of images to numerical values. It is used wherever an image is to be displayed in a mask instead of a numerical value from the PLC. The window for image lists consists of two sections. The left section of the window displays a table, the right section displays a blank mask. The mask located on the right allows you to specify the size of the image and to control the representation in the process.

The table on the left consists of four columns and at least two rows.

- | select the **PROJECT MANAGEMENT INFORMATION** tab
- | select the menu item **Image lists**
- | press the **<New>** button to create a list
- | enter a name for this new image list, for instance [Symbol 1]
- | the list editor is displayed



The first row of the column **Value** contains the word Default. "Default" indicates that the image in this line is displayed as long as no valid value is read from the controller. Otherwise, enter the numerical value in the column **Value** that the controller must transmit to the terminal for the corresponding image to be displayed.

### 13.2 Inserting an image from an existing file

Any graphics program can be used to create the images to be used in the programming software. The required procedure is explained below:

- | select the default row on the column **Image links**
- | press the right mouse button
- | select the item **Create new Image**
- | enter a name, e.g. [Symbol 1] and complete the entry with **<OK>**
- | from the Insert Object dialog box, select the **<Create from File>** button
- | click **<Browse>** to select the corresponding image file
- | confirm your selection with **<Insert>**
- | confirm again with **<OK>**

The image is displayed, in mask size, on the right side of the window

You can use the sizing handles to resize the image. The new numeric values for height and width are automatically entered in the table on the left. You can also enter values in the table and view the resulting image displayed on the right (Example: width = 48 pixels and height = 48 pixels).

The image is finished and you can now insert another image into the table.

#### NOTES :

---



---



---



---

### 13.3 Inserting a new image using another program (OLE)

Images required in the programming software can be created directly from within the programming software's operator guidance by using a suitable graphics program.

The graphic program to be used for this purpose must be an OLE enabled server.

The required procedure is explained below:

```
| select the default row on the column Image links
| press the right mouse button
| select the item Create new Image
| enter a name, e.g. [Symbol 1] and complete the entry with <OK>
| click the program entry that you want to use to create an object (image)
| confirm your selection with <OK>
| create the object using the selected program
| return to the programming software (using the menu item which exits the
  program)
```

The image is displayed, in mask size, on the right side of the window

You can use the sizing handles to resize the image. The new numeric values for height and width are automatically entered in the table on the left. You can also enter values in the table and view the resulting image displayed on the right (Example: width = 48 pixels and height = 48 pixels)

### 13.4 The Write-over Mode

This column allows you to select the mode in which this image is to overwrite other images in the mask.

Mode in which a pixel of the new image overwrites that of the old image.

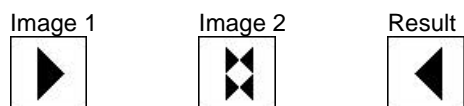
#### SET Mode

Image 1 is overwritten by image 2.



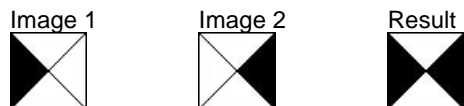
#### XOR Mode

Mode in which either the pixels of the top image or the pixels of the bottom image are visible in the overlapping area.



#### OR Mode

Mode in which either a pixel of the top image or a pixel of the bottom image are visible in the overlapping area depending on which pixel is <activated>. The representation represents the <as well as> principle.



### 13.5 The Display Mode

This column allows you to determine if this image is to be displayed normal, inverse, flashing or inverse and flashing.

NOTES :

---



---



---

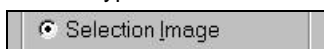


---

### 13.6 Creating a Selection Image variable

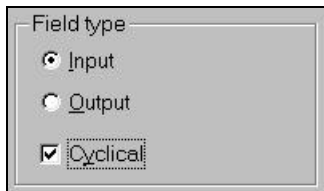
The procedure to create a variable **Selection image** is nearly the same as that to create a variable **Selection text**. The only difference is the assignment of an image list instead of a text list.

| select the representation type



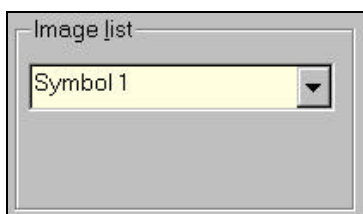
| press the button **<Edit type>**

| define the field type



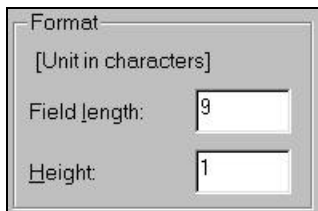
The field type defines whether the selection image variable is to be an input or output variable and if the value is to be read from or written to the PLC at cyclic intervals.

| assign an image list

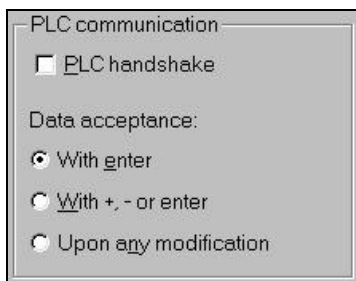


Select an image list from the selection field. If an image list does not exist yet, you can insert a name and create it after entering the name in a separate procedure.

| define the variable format



In the Format area, enter the field length and the field height. The unit **<Characters>** is used. A graphic can only be a multiple of the size of a character. The character set of the terminal NT18S-SF121B-E has a size of 6x8 pixels per character (width x height).



To be able to transmit the values of the variable to the PLC, specify whether a value is to be transmitted only after the Enter-key is pressed, after the PLUS-key, MINUS-key or Enter-key is pressed or automatically upon every modification.

| confirm your inputs with the **<OK>** button

**NOTES :**

---



---



---



---

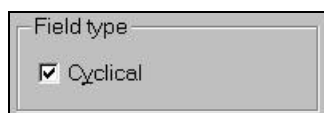
## 14 Bar chart

The variable type Bar is a pure output variable used to display the values of variables as bars.  
The value for the bar variable can be refreshed by the PLC at cyclic intervals.

- | select the **LANGUAGE** tab
- | select menu item **masks**
- | create a new or open an existing mask
- | create a new variable and insert a name, e.g. [bar]
- | select the representation type

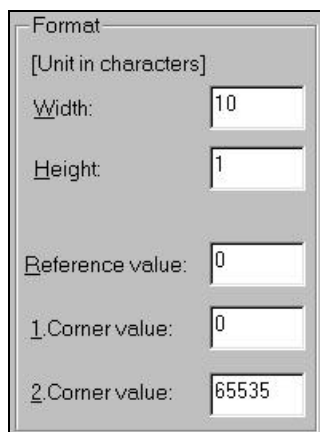


- | press the button **<Edit type>**
- | define the field type



The field type defines whether the selection image variable is to be an input or output variable and if the value is to be read from or written to the PLC at cyclic intervals.

- | define the variable format



The Format area allows you to specify values for the width and height to determine the area which the bar will take up in a mask. The value for the width and the height is specified using the unit <Pixel>. The size of a bar depends on the pixel size and the distance from pixel to pixel and can produce different results depending on the type of terminal.

In the Reference Value field, specify the value the bar is to be filled up to the current value. The starting point of the bar is always the reference value. If the defined reference value is half way between the minimum and maximum value, the bar will begin to expand from the center. If the specified reference value is placed at the first corner value, the bar will begin to expand from the bottom or the left.

Using the first and the second corner value (upper and lower limits), you are able to determine the range of values for the bar.

With the first corner value determine:  
the value of the bar at the left or bottom end

With the second corner value determine:  
the value of the bar at the right or top end

### NOTES :

---



---



---



---



Define the representation of the bar chart

Representation

Bar:

Background:

Corner values:

fallen below lower limit:

exceeded upper limit:

Expansion:

horizontal  vertical

To be able to display a bar on the terminal, you will need to specify a representation (fill pattern) in the Representation area, one fill pattern for the representation of the bar and another for the background of the bar area. In addition, you can use two further fill patterns to illustrate that a value is outside of the upper or lower limit of the range of values for a bar.

Apart from any images you create, seven standard patterns are available for bars: Blank, filled, horizontal, vertical, backslash, slash and hatched. The comment (internal) placed next to the corresponding names in the list are only used to identify each standard pattern. The bar can be displayed vertically or horizontally.

While defining the representation of the bar, the preview box allows you to view your settings.

Assign a PLC variable to the bar chart  
confirm your inputs with <OK>

**Examples:**

Reference value:

1. Corner value:

2. Corner value:



Reference value:

1. Corner value:

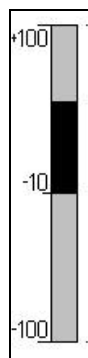
2. Corner value:



Reference value:

1. Corner value:

2. Corner value:



Reference value:

1. Corner value:

2. Corner value:



A bar chart can be displayed as any type of graphic, e.g. as a tank. For this purpose, you need two images with the same size where each of these images represents the status of one of the two limits (filled and empty). The output of the bar chart is automatically calculated by the terminal.

**NOTES :**

---



---



---




---



## 15 Tables

The table field is the area within a mask which is used to display values in a table.

- | select the **LANGUAGE** tab
- | select menu item **masks**
- | create a new mask or open an existing mask

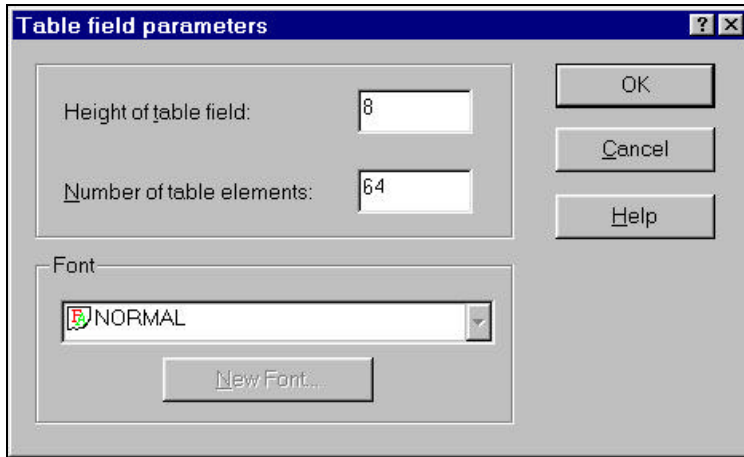
| click the button 

| drag open a field within the mask with the mouse cursor

This area is displayed as a rectangle. The letter **T** is displayed at the left edge of the rectangle of the table field.

The height of this area can be modified using the sizing handle on the corners. The width cannot be modified.

- | parameterize the displayed dialog



Specify the number of lines for the table field and the number of elements to be displayed in this table field. The Font area displays the font used to display all of the elements of the table on the operating terminal.

Example:

256 elements are to be displayed in a table with four columns. The operating terminal's display provides 16 lines, 12 of which are to be used to output elements. In this case, create a table field with a height of 16. Enter 64 for the number of table elements (16 lines x 4 elements).

- | confirm your parameterization with the **<OK>** button

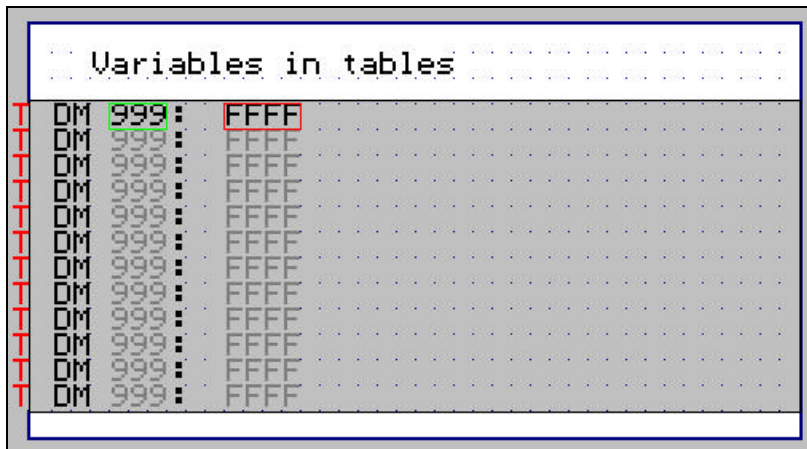
| click the table button 

| drag open a field within the table field with the mouse cursor

| insert a type of variable you want to display within the table

| insert the system variable **VarTablenR0** or **VarTablenR0**

| confirm your inputs with **<OK>**



NOTES :

---



---



---



---

## 16 Appendix

### 16.1 Scaling of variables

The

- Factor
- Divisor
- Summand

can be calculated with the following equation

$$\frac{y - y_1}{x - x_1} = \frac{y_2 - y_1}{x_2 - x_1}$$

where:

X is the value on the terminal  
 X<sub>1</sub> is the lower limit on the terminal  
 X<sub>2</sub> is the upper limit on the terminal

Y is the value in the controller  
 Y<sub>1</sub> is the lower limit in the PLC  
 Y<sub>2</sub> is the upper limit in the PLC

Example:

range of input values 0 = X<sub>1</sub> / 100 = X<sub>2</sub>  
 range of value within PLC -4096 = Y<sub>1</sub> / +4096 = Y<sub>2</sub>

Step 1 - Enter the variable values:

$$\frac{y - (-4096)}{x - 0} = \frac{4096 - (-4096)}{100 - 0}$$

$$\frac{y + 4096}{x} = \frac{4096 + 4096}{100}$$

Step 2 - Solve the fraction

$$100y + 409600 = 8192x$$

Step 3 - Solve the equation for x:

$$x = \frac{100}{8192}y + \frac{409600}{8192}$$

$$x = \frac{100}{8192}y + 50$$

Factor

Summand

Divisor

**NOTES :**

---



---



---



---

## 16.2 System variables for recipe management

### SelectDSNr

This variable contains the number of the active data set. To edit the variable, the associated selection field editor must be used.

Data type: ..... Numeric  
 Editor: ..... Selection field  
 Output: ..... Decimal number  
 Possible values: ..... 0...250

### SelectDSName

This variable contains the name of the active data set. To edit the variable, the associated selection field editor must be used.

Data type: ..... Alphanumeric  
 Editor: ..... Selection field  
 Output: ..... Alphanumeric  
 Possible values: ..... 15 characters

### DestDSNr

When copying data sets, this variable contains the number of the destination data set.

Data type: ..... Numeric  
 Editor: ..... Decimal number  
 Output: ..... ---  
 Possible values: ..... 1..250

### DSCopy

This variable can be used to copy the active data set to the destination specified in DestDSNr.

Data type: ..... Numeric  
 Editor: ..... Selection text, decimal number, softkey, function key  
 Output: ..... ---  
 Possible values: ..... (0) Normal position  
                   (1) Copy to destination in system variable DestDSNr  
                   (2) Copying and automatic search for a free data set  
                   (3) Copy to destination in system variable DestDSNr.  
                       Overwrites an existing data set.

### DSDelete

this variable can be used to delete the active data set. The first data set of the same recipe will become the new active data set.

Data type: ..... Numeric  
 Editor: ..... Selection text, decimal number, softkey, function key  
 Output: ..... ---  
 Possible values: ..... (0) Normal position  
                   (1) Deletes the data set

### ActDSName

This variable contains the number of the current data set. It can be read, RAM-data sets can additionally be written.

Data type: ..... Alphanumeric  
 Editor: ..... Alphanumeric  
 Output: ..... Alphanumeric  
 Possible values: ..... 15 characters

### SelectRezeptNr

This variable contains the active recipe. The variable can be modified in the recipe mask or any other mask.

Data type: ..... Numeric  
 Editor: ..... Numeric, Selection text  
 Output: ..... Numeric, Selection text  
 Possible values: ..... 1...255

### NOTES :

---



---



---



---

### 16.3 System variables for recipe data exchange

#### DSDownload

This variable can be used to write the active data set to the controller.

Data type: ..... Numeric

Editor: ..... Selection text, decimal number, softkey, function key

Output: ..... ---

Possible values:..... (0) Normal position  
(1) Writes the data set

#### DSDnloadBreak

This variable can be used to terminate a data transfer to the controller currently in progress.

Data type: ..... Numeric

Editor: ..... Selection text, decimal number, softkey, function key

Output: ..... ---

Possible values:..... (0) Normal position  
(1) Terminates data set transfer

#### DSDnloadState

This variable can be used to monitor the data transfer to the controller.

Data type: ..... Numeric

Editor: ..... ---

Output:..... Selection text

Possible values:..... (0) Initial state

(1) Data transfer was requested, but not yet released by the controller  
(2) Data transfer active

#### LoadDSName

This variable contains the name of the data set which has most recently been sent to the controller. If the data set has already been deleted, question marks are displayed.

Data type: ..... Alphanumeric

Editor: ..... ---

Output:..... Alphanumeric

Possible values:..... 15 characters

#### StartUpload

This variable can be used to read, for the active recipe, a data set from the controller and to save it in the terminal.

Data type: ..... Numeric

Editor: ..... Selection text

Output: ..... ---

Possible values:..... (0) Initial state

(1) Variables are read individually from their specified addresses  
(2) Variables are read as a block from the buffer specified for the recipe.  
(3) Variables are read individually from their specified addresses.  
A free data set is automatically being searched for.  
(4) Variables are read as a block from the buffer specified for the recipe.  
A free data set is automatically being searched for.

#### UploadDSNr

The variable specifies the data set number to which the uploaded data set is to be written.

Data type: ..... Numeric

Editor: ..... Numeric

Output: ..... ---

Possible values:..... 1...255

#### UploadState

When uploading data sets, this variable indicates the upload status.

Data type: ..... Numeric

Editor: ..... ---

Output:..... Selection text

Possible values:..... (0) Initial state

(1) Upload of data set in progress

#### NOTES :

---



---



---



---

## 16.4 System variables for data set backup to a PC

### StartSave

This variable can be used to transfer data sets to the PC.

Data type: ..... Numeric

Editor: ..... Selection text

Output: ..... ---

Possible values:..... (0) Initial state  
 (1) Transfer one data set to the PC  
 (2) Transfer all data sets of a recipe to the PC  
 (3) Transfer all data sets in the terminal to the PC

### SaveState

During a transmission to the PC, this variable indicates the current status of the transmission process.

Data type: ..... Numeric

Editor: ..... ---

Output: ..... Selection text

Possible values:..... (0) Initial state  
 (1) One data set is transferred to the PC  
 (2) All data sets of a recipe are transferred to the PC  
 (3) All data sets in the terminal are transferred to the PC

### StartRestore

This variable controls the download from the PC to the Terminal.

Data type: ..... Numeric

Editor: ..... Selection text

Output: ..... ---

Possible values:..... (0) Initial State  
 (1) The terminal switches to ready-to-receive state  
 (2) The terminal terminates a transmission currently in progress.

### RestoreState

This variable indicates the status of the transmission from the PC to the terminal.

Data type: ..... Numeric

Editor: ..... ---

Output: ..... Selection text

Possible values:..... (0) Initial state  
 (1) Data transmission in progress

### RestoreLineNr

This variable indicates the current line number of the data set file. It is used to display the progress of the receive process and in the case of an error, for fault localization.

Data type: ..... Numeric

Editor: ..... ---

Output: ..... Numeric

Possible values:..... 1..255

## 16.5 System variables for data set printout

### StartRezPrint

This variable can be used to print data sets.

Data type: ..... Numeric

Editor: ..... Selection text

Output: ..... ---

Possible values:..... (0) Initial state  
 (1) Starts data set printout  
 (2) Terminates print process

### RezPrintState

When printing data sets, this variable indicates the current printer status.

Data type: ..... Numeric

Editor: ..... ---

Output: ..... Selection text

Possible values:..... (0) Initial state  
 (1) Data set printout in progress

### NOTES :

---



---



---



---

## 16.6 Types of variables

### Decimal number

The representation format basically includes four specifications:

The field length, that is the total length including the sign and point.

The fractional digits (number of decimal places after the point), that is the digits to be displayed after the point (e.g. currency values have two digits after the point).

The option of limiting decimal values to positive values.

In the case of small numbers (requiring only a small number of digits), the option to fill the digits not used with zeros (leading zeros).

### Alphanumeric

Represents an alphanumeric string. Every character corresponds to a byte address within the PLC. The complete ASCII-character set can be displayed. The string is addressed in a linear order.

### Selection text

The selection text variable allows you to display texts from a text list. The texts must be predefined in a text list.

The variable is then linked to this text list. Variables of the type Selection Text can not be created globally.

The variable must be created and linked to a text list for each language separately. For this process, you can select a text list from a list in the Text List area. The insertion of images is limited to 2048.

### Selection Image

The variable Selection Image allows you to display images from an image list instead of displaying values. The images must be predefined in an image list. The variable will then be linked to this image list. To do so, select an image list from the Image list area.

### Floating point number

The representation format of a floating point number consists of the field length and the fractional digits (number of decimal places after the point).

### Hexadecimal number

The length of a variable in the hexadecimal format is limited to a maximum of 8 Characters (Digits).

The highest number is therefore FFFF FFFF. It is therefore not possible to define negative hexadecimal numbers.

### Bar

The variable type Bar is a pure output variable used to display the values of variables as bars. The Format area allows you to specify values for the width and height to determine the area which the bar will take up in a mask.

The value for the width and the height is specified using the unit <Pixel>. The size of a bar depends on the pixel size and the distance from pixel to pixel and can produce different results depending on the type of terminal.

### Curve

The variable type Graph (for curve representation) is a pure output variable. For the field type, simply specify if the values of the graph are to be refreshed at cyclic intervals. For the format, enter a height and width. The expansion might vary with the various terminal types due to the different displays being used. The documentation value is a character string used to fill the variable field in the mask. If the documentation value is shorter than the field, it is entered repeatedly.

A graph variable allows you to display a continuous stream of values graphically. The 1st quadrant of a Cartesian coordinate system is used for the graphical representation. For this process, the values to be represented must be transmitted to the terminal as consecutive bytes. The number of bytes determines the length of the graph. This means that the "number" of the byte corresponds to the x-axis position of the graph point and the value of the byte corresponds to the y-axis position of the graph point.

The graph can be displayed as a "flowing" graph. In this case, the range of values to be represented must be updated in the controller in accordance with the FIFO-principle between every poll process.

### NOTES :

---



---



---



---



