



24.08.2016

Programmer's Guide

PM String Change

PSi PSi Family PP 40x/PP 80x

Acknowledgement

EPSON is a trademark of the Seiko Epson Corporation.

IBM, ProPrinter are trademarks of the International Business Machines Corporation.

Microsoft, Windows, Windows NT, are trademarks of Microsoft Corporation

A Publication of Psi Matrix GmbH
Hommewiese 116c
D – 57258 Freudenberg
Federal Republic of Germany
November 2015
<http://www.psi-matrix.eu>

Great care is taken to ensure that the information in this handbook is accurate and complete. However, should any errors or omissions be discovered or should any user wish to make suggestions for improving this handbook, please feel encouraged to send us the relevant details.

The contents of this manual are subject to change without notice.

Copyright © 2016 by Psi Matrix GmbH.

All rights strictly reserved. Reproduction or issue to third parties in any form is not permitted without written authorization from the publisher.

Content

- 1 Description of PM’s Supporting StringChange-Functionality 4**
 - 1.1 Personality Modules with Function StringChange..... 4
 - 2.1 Installation of the PM’s..... 5
 - 3.1 Connectors, Keys, and LED’s 5
- 2 Application and Installation..... 6**
 - 2.1 Programming 6
- 3 Definition of StringChange Functions 7**
 - 3.1 StringChange – Functions Mode 0 to Mode 8 1
 - 3.2 Format of a StringChange – Definition 2
 - 3.3 Use of Joker and Taking- over Character 2
 - 3.4 Which Trigger becomes valid? 3
- 4 Descriptions and Examples of the various Modes..... 3**
 - 4.1 Mode 0..... 3
 - 4.2 Mode 1..... 4
 - 4.3 Mode 2..... 4
 - 4.4 Mode 3..... 5
 - 4.5 Mode 4..... 6
 - 4.6 Mode 5..... 6
 - 4.7 Mode 6..... 7
 - 4.8 Mode 7..... 8
 - 4.9 Mode 8..... 8
- 5 General Comments 10**
 - Working with a STGCHG.ini- File..... 10
 - 5.1 Content of the STGCHG_D.ini- File 10
 - 5.2 Technical Data..... 15

This appendix contains basic information about the StringChange function for all models in these types of printers supported:



1 Description of PM’s Supporting StringChange-Functionality

Personality Modules with StringChange- functionality are used where problems with the printer data stream are encountered, which can be solved by modifications and adaptations of the data stream. This kind of problems appear if existing old printer installations shall be replaced by new equipment’s and incompatibilities have to be fixed which cannot be done on system level. The StringChange- functionality of the corresponding Personality Modules for the printer families PP80x and PP40x offers the PSi VAR’s opportunities to develop independent solutions for his customers and to protect the investments.

In general, terms StringChange is the treatment of incoming character strings and output to the printer. StringChange can modify or convert control commands or delete unnecessary characters from the data stream. Of course, pre-conditions are definite code strings and functions. A good understanding of printer functions and their codes will ease the task.

1.1 Personality Modules with Function StringChange

Order number:

PP 40x:

8707-241-90110
8707-241-90111
8707-241-90114

PM PAR STRING CHANGE PP40x
PM ETH STRING CHANGE PP40x
PM ETH/USB PjL STGCHG PP40x

Order number:

PP 80x:

8707-340-90127
8707-340-90128
8707-340-90142
8707-340-90143
8707-340-90144
8707-340-90145

PM PAR STGCHG 80X MT50
PM ETH 10/100 STGCHG 80X MT50
PM ETH 10/100 MB/S PjL PP80x
PM ETH 10/100 MB/S PjL STGCHG PP80x
PM ETH 10/100 MB/S PjL IGP PP80x
PM ETH 10/100 MB/S PjL STGCHG IGP PP80x

2.1 Installation of the PM's

Insert the Personality Module into the corresponding shaft of the printer until the connector fully engages. Hand tighten the two locking screws. The installation of the Personality Module.

3.1 Connectors, Keys, and LED's

Green LED Push Button

Ethernet or Parallel



For downloading a StringChange – definition file depress and hold the push button while the printer is powered on. The green LED is on until the download process begins and off after termination. While download of larger files the LED is blinking. The description for the Ethernet interface is available in a separate documentation.



PP 80x



+

PP 404/5



PP 407/8



2 Application and Installation

It is recommended to analyse the required steps in the fore field. This can be done without having a Personality Module with the StringChange function. Only the StringChange tools installed on a PC and the data stream of the application are required. Preferably, the data streams of the application should be available as printable file or need to be re-edited from a hex- dump. After analysis of the required data conversion, the StringChange- tools allow to test the StringChange- definitions and the output of the conversion.

For those purpose two files, the StringChange- definition and the data stream of the application (output) are processed into a file containing a printable output of the conversion. This printable output file can be analysed using a so-called Hex- Editor. This file can be printed with a standard printer without StringChange to test the print result of the conversion.

2.1 Programming

2.1.1 Installation of StringChange Tools

Tools are available for English or German languages in their own directories on the CD. Copy the preferred directories (language) and the related files on your PC into a new directory.

Then copy the DLL- files from the subdirectory SYSTEM32 into the windows directory WINDOWS\SYSTEM32 and WINDOWS\SYSTEM

The directories contain standardized files, which shall ease building of definitions for the StringChange functions.

- **STRCHG_D.ini** - File for the definitions of StringChange- functions TESTSTG.bat BAT- file for testing of STRCHG_D.ini, generates a printable output- file STGCHG.out
- **mkSTGCHG.bat** - file generates a MOT- file for downloading the StringChange – definitions into a StringChange Module (PM)
- **STGCHG0000.mot** - Disables the StringChange Module on the Personality Module (StringChange without IGP)
- **Emu0000.mot** - Disables the StringChange Module on the Personality Module with IGP- functionality
- **Print_STGCHG_ver** - Prints the Versions- Output of the StringChange-functions

It is helpful and timesaving to use a standard HEX- editor to analyses the application data and the test results STGCHG.out before the STGCHG.mot will be installed in the Personality Module.

A free Hex- editor WinVi can be downloaded from www.winvi.de.

3 Definition of StringChange Functions

A StringChange function allows the definition of replacement functions by conversion of parts of the incoming data stream into the outgoing data stream. These are in common control commands to be encoded in other control commands or deleting unwanted codes. Variable data of the incoming data stream, e.g. parameter of a control command can be inserted into the new definition for the outgoing data stream.

The data stream of the application should be analysed carefully. Formulate unique equations, which confront the coding of the incoming data stream with the new function in the outgoing data stream.

Special attention requires the handling of variables. Variable ones can be character strings of fixed or variable length with clear end criterion or also parameter (values) in a control instruction.

Into the outgoing data stream, the variable data can be transferred to pre-defined places in the output string or also eliminated however; StringChange does not convert to other values. Therefore, a good knowledge of the instructions and control functions existing in the printer is of advantage, in order to define the replacement functions optimally.

Generate yourselves one the application appropriate binary file. This can come from a trace of the data stream of an application or selectively parts, which are edited from a HEX- DUMP into a new file. Designate this file for the project development with STRCHG.inp. Use the available file STRCHG.ini to define the StringChange functions.

In the file STRGCHG_E.ini are included some examples, which shall ease the entrance into the StringChange programming. Delete the examples Trg.../Out... and replace these by your own definitions. You can set the examples also to the end of the definitions in comment lines with a # - indication at the start of a line.

Proceed with the development of the StringChange definitions gradually and examine repeatedly occasionally the correct function by application of the test function TESTSTG.bat. TESTSTG.bat needs the files STRCHG.ini with the StringChange definitions and the file STRCHG.inp as application input file.

TESTSTG.bat generates a STGCHG.out file with the data, which resulted from the StringChange definitions of the STRCHG.ini after conversion of the data from the STRCHG.inp.

With a usual Hex editor, you can examine the desired result of the conversion in the file STGCHG.out. Depending on complexity, several runs are necessary, until the desired StringChange functions can be used for the application. The STGCHG.out file can be transferred also directly to the printer, in order to examine the correct printout of the application after conversion by StringChange. For that, no Personality Module (PM) with StringChange functionality is necessary it is less time- consuming to test in the first step only with the printout of the file STGCHG.out before the final new StringChange definitions are downloaded in the printer. This proceeding is interesting also; if a StringChange, definition shall be provided or changed, without having an appropriate PM with a StringChange module at hand.

3.1 StringChange – Functions Mode 0 to Mode 8

Nine different modes are available as:

Mode 0:

Character strings look up and replace. Variable in the trigger possible with one character per Joker character. No Transfer of the variables in the Outputs.

Mode 1:

Search for character strings with and without Joker and replace them with a bi-stable expression without taking-over of variable characters from the trigger. With the first occurrence of the trigger character string, the first defined output will sent and with the next occurrence the second one, then again the first output a.s.O.

Mode 2:

Search and replace character strings with variable characters of variable length without taking-over of the variable characters in the output string. The taking-up of the variable data takes place to the character after the Joker or if the first character of the trigger will found again.

Mode 3:

Search for a character string, which shall repeat the character placed in front of the trigger in the output in the location assigned by the taking-over character as often as the value of the variables marked by the Joker in the trigger.

Mode 4:

A character string, also with variable characters, will removed.

Mode 5:

Bi-stable name, defines name of the output. Functional extension for Mode 1 to define additional pairs of output strings.

Mode 6:

Counter, n times the outputs. Divides a large count value into smaller values in the output string.

Mode 7:

Search for a character string with variable data, variable length and Taking-over at a defined position in the output. The taking-over of the variables is optional.

Mode 8:

Search for a character string with one or more variable characters. The variable characters can be transferred to the output. A taking-over character represents one character location of the variable characters in the output.

3.2 Format of a StringChange – Definition

For each replacement function (trigger), the suitable mode must be selected. The General Definition has the following format:

The trigger define: Trg... and the output define: Out.....

- Trg <#> : <mode> , "<Definition>"
- Out <#> : "<Definition>"
- <#> : sequential number of the triggers and output definitions, which do not have to be Arranged however in ascending order.
- <mode> : number, which marks the mode: 0 to 8
- <Definition>: definition of the trigger string and/or output string.

Blank lines may not be entered between the definitions Trg... and Out.... There are different kinds of character input:

Normal ASCII characters and hexadecimal representations, e.g.. ^0A for a linefeed, can be used. The hexadecimal representation is always then necessarily, if the character is not contained in the standard ASCII character set, or it is special character, or it is a national substitution character, or a control character.

With excessive line lengths of the definitions, e.g. > 80 columns, the line is partitioned, so that it remains readable: „\" stands for line-wrap-around.

```
"<Definition>"\  
"<Definition>"\  
"<Definition>"\  
"<Definition>"
```

Maximally a definition (trigger or output string) may contain 255 characters. If more characters are used then FF- characters (hexadecimal) appear in the output string and the output string is faulty.

3.3 Use of Joker and Taking- over Character

- a) Variable characters and variable character strings are indicated in the trigger definition (Trg) with wildcard characters so-called Joker characters (?) and in the output (Out) with taking-over characters (!). For a Joker character, as many as desired characters can be taken up, limited by the size of the input - buffers, with exception in the mode 0, with which only one character per Joker is permitted. The taking-up of characters by a Joker is terminated through: a): the character after the Joker in the trigger
- b) b): Joker break- character (!=\$00)
- c) c): the first character of the trigger is again found. However, stand two Joker (??) after each other; the taking-over of characters is not stopped. Joker 1 and Joker 2 can be defined. Both Joker are equivalent in taking-up the data. The difference consists of the fact that only data of the Joker 1 can be transferred in the outputs.

For the taking-up with Joker, it is not necessary to use more than two Joker one behind the other in the trigger. However, it can as many as desired places be defined with Joker.

If a Joker is indicated as last indication in a trigger, then thereafter the trigger is terminated and only one character as variable is considered. (Example for this in the description of MODE 8)

3.4 Which Trigger becomes valid?

In General, that trigger becomes effectively, which took up most characters. If several triggers with the same data and data sets are active and have a match, then that is valid, which was set on first in the INIT file. If same triggers are defined, which differ only by fixed and variable values, and then the triggers with the fixed values are to be put on in the INIT file before the triggers with variable values.

4 Descriptions and Examples of the various Modes

The simple examples are to clarify the principle and the function of the various modes. The more complex examples are rather from practice.

4.1 Mode 0

A character string (trigger) is replaced by another character string (output). In the trigger Joker- character can be used for variables. However, the characters from the Joker positions cannot be transferred to the output string. A Joker- character stands for in each case for a single variable character.

Example 1:

A simple example is the replacement of a name

Trigger 1:0

```
Trg 1: 0,"PP405"  
Out 1: "PP806"
```

If the character sequence PP405 in the data stream is found, then this is replaced with the character sequence PP806.

The character sequence PP with three further variable characters is to be converted into the character sequence PP809. Exception is the character sequence PP405, which is defined as fixed character sequence with trigger 1 before trigger 2 and sends PP806 as character sequence. Because in the mode 0 a Joker defines only a single variable character, three Joker- characters are indicated for three variable characters in the trigger- definition for Trg 2.

Trigger 2:0

```
Trg 2: 0,"PP???"  
Out 2: "PP809"
```

Example 2:

A practical example:

A bar code - definition shall be replaced by an appropriate for a PSi- printer

Trigger 3, 4 und 5:0

```
Trg 3: 0,"^14^14^1b^21^02^11^00" # Barcode Header  
Out 3:  "1b^5b;103;;;;; z"  
Trg 4: 0,"^19^14^14^1b^28^1d^00" # Start Barcode  
Out 4:  "1b^5b?0h:"  
Trg 5: 0,"^19^14^14^1b^59^02" # Stop Barcode  
Out 5:  ";1b^5b?0l"
```

If the character sequence of the trigger (Trg 3/4/5) in the data stream is found, then this is replaced by the character sequence (Out 3/4/5). In this case geometry of the bar code must be determined and the appropriate values set into the bar code header of the output (Out). In this case it is code 39, bar code height 5/12 inch, width of the narrow lines 2/144 inch, width of the narrow gaps 2/144 inch, relationship to wide to narrow 3.0 : 1

4.2 Mode 1

For a trigger is defined (Bi stable) output (two output strings). With the first occurrence of the trigger Definition, the first output character string, with the next trigger the second character string (Out 1,1) and then again, the first character string is sent a.s.o.

As a real application a switching on / off function would come into consideration, e.g. "bold print" and "bold print off"

Example 1:

Trigger 1:1

Trg 1: 1,"PSI"

Out 1: "Printers for professionals ^OD^OA"

Out 1.1: "PP806 ^OD^OA"

^OD^OA represent the hexadecimal values for the control characters CR LF.

Input – Data of the application:

PSIPSPSPSPSPSPSI

After transformation of the input data, the following output string is sent:

Printers for professionals

PP806

Printers for professionals

PP806

Printers for professionals

PP806

In the trigger, definition Joker- character can be used.

Example 2:

Trigger 2:1

Trg 2: 1,"?PSI"

Out 2: " ! Printers for professionals ^OD^OA"

Out 2.1: "PP806 ^OD^OA"

^OD^OA represents the control commands CR LF.

Input – Data of the application:

1PSI2PSI3PSI4PSI5PSI6PSI

After transformation of the input data, the following output string is sent:

1 Printers for professionals

2 PP806

3 Printers for professionals

4 PP806

5 Printers for professionals

6 PP806

4.3 Mode 2

Mode 2 defines the replacement function for a string with variable data without their taking-over in the output string. The taking-up of the variable data takes place up to the character after the Joker or if the first character of the trigger is found again. The second condition applies only, if in the trigger a single Joker character (?) instead of two Joker characters (??) are used.

Example 1:

A trigger with variable characters (Joker) is defined. The output shall be a fixed character string without taking-over of the variable characters. Compared to mode 0 one or more characters can be taken up until reaching the character after the Joker.

In the INI- file, the following trigger is defined:

Trigger 1:2

Trg 1: 2, "P??!"

Out 1: "PP803"

The following character strings are detected in the input data as triggers:

PSI PPI PTI PLI P1I PXI Pxxxxl Pyyyyyyyyyy P123467890l

And they are placed in each case after transformation as fixed character string into the output:

PP803 PP803 PP803 PP803 PP803 PP803 PP803 PP803 PP803 PP803

Example 2:

A practical example:

A string begins always with one or more well-known characters, followed from a fixed number of variable characters (Joker) and always ending with one or several well-known characters. In this example, unwanted characters in the data stream must be eliminated.

The definition should be defined as complex as possible, so that different data of the application are not inadvertently included.

Trigger 1:2

Trg 1: 2, "^0A^00??????^20"

Out 1: "^0A^20"

All data between ^0A and ^20 are removed. Here ^00 was also included into the definition, so that the definition is more definite and different data are not inadvertently removed

4.4 Mode 3

Mode 3 repeats the character preceding the trigger with the value of the character(s) marked by the Joker "?" in the output string at the position marked by "!". The default value for the variable "?" is 1. The number of repetitions is limited to 100. The value for the variable is coded with ^30 to ^39 (in hexadecimals).

Example 1:

If the trigger in the input data stream is detected, then the character before the trigger is as often to be output, as in the position with the Joker (?) is defined.

Definition in the INI- file:

Trigger 1:3

Trg 1: 3, "PSI?test"

Out 1: "!!!"

Input – data of the application:

^PSI4test

Here the character ^ preceding the trigger in the input data stream is sent 4 times in the output.

Output after conversion: ^^^^

Example 2:

The ANSI instruction Repeat character is replaced by an appropriate number of characters in the output.

Trigger 1:3

Trg 1: 3, "^1B^5B?b"

Out 1: "!!!"

Preceding and trailing characters may be added to the taking-over character "!".

4.5 Mode 4

Mode 4 is used as filter. The trigger is removed from the output data stream.

Example 1:

The character string "OLD" is to be removed from the data stream. Definition in the INI- file
Definition in the INI-File

Trigger 1:4

Trg 1: 4,"^22OLD^22"

^22 is the hexadecimal representation of the character quotation mark Input data:

"OLD" shall not appear in the output data stream Output data:

"OLD" was removed.

Example 2:

A character string with variable characters is to be removed.

Trigger 2:4

Trg 2: 4,"alte Ausgabe ?? "

The last character in the trigger is a blank (space).

Input Data:

Test 1 alte Ausgabe 1 !

Test 2 Das ist eine alte Ausgabe von 1999

Test 3 diese alte Ausgabe xxxxxxxxxxxxxxxxxx ?

Output Data:

Test 1 !

Test 2 Das ist eine 1999

Test 3 diese ?

4.6 Mode 5

BI STABLE name, defines Name of the output.

Extension function to mode 1. Further pairs can be assigned as bi stable output strings (Out) with names in the mode 1. The name is 1 character long and read by the position of the Joker. In the output, definition (Out) the name (1 character) is indicated in the first position. The name is not put into the output. After the last output, string should stand a blank line. In the application data, a \$FF- character should be placed after the last output string. If no name is defined or the name is missing then the first pair of output strings without names, (1.1 and 1.2) is used.

Example:

Trigger 1:5

Trg 1: 5,"Toggle_"

Out 1.1: "TOGGLE ON^0A^0D"

Out 1.2: "TOGGLE OFF^0A^0D"

Out 1.3: "xTOGGLE X^0A^0D"

Out 1.4: "yTOGGLE Y^0A^0D"

Out 1.5: "zTOGGLE Z^0A^0D"

Trg 2: 5,"Bistab?Name"

Out 2: "!^0A^0D"

At least the taking over character "!" is required in the OUT string to activate the name.

Input Data:

a Toggle_

b Toggle_

a Toggle_

b Toggle_

BistabxName

c Toggle_

d Toggle_

c Toggle_

d Toggle_

BistabyName

e Toggle_

f Toggle_

e Toggle_

f Toggle_

Output Data:

a TOGGLE ON

b TOGGLE OFF

a TOGGLE ON

b TOGGLE OFF

c TOGGLE X

d TOGGLE Y

c TOGGLE X

d TOGGLE Y

e TOGGLE Y

f TOGGLE Z

e TOGGLE Y

f TOGGLE Z

4.7 Mode 6

Counter, n times the outputs divides a larger count value into smaller values in the expenditure stringer. Values more largely 500 are divided in the expenditure into $n * 500$ plus residual value. The application of this function is very special.

Example:

Trigger 1:6

Trg 1: 6, "Count ? Mode"

Out 1: "COUNT ! times^20"

Input Data:

Count 1 Mode

Count 20 Mode

Count 400 Mode

Count 1531 Mode

Output Data:

COUNT 1 times

COUNT 20 times

COUNT 400 times

COUNT 500 times COUNT 500 times COUNT 500 times COUNT 31 times

4.8 Mode 7

Mode 7 defines a replacement function with variable data and variable length. The variable in the trigger is defined with one "?" (Joker). The variable from the trigger is inserted into the position designated by the taking-over character "!" in the output (Out). Also an output string (Out) without taking-over of a variable can be defined.

Example:

"^2A?^2A^19" is a part of a bar code, the variable characters of the bar code are enclosed with the character asterisk hexadecimal ^2A. Here it concerns the code 39.

The output defines a barcode according to PSi syntax with taking-over of the variable data.

Barcode Header: ^1b^5b^3b^31^30^33^3b^34^3b^31^3b^31^3b^31^3b^20^7a
Start Barcode: ^1B^5B^3F0h
Variable Barcode Data: ^2A!^2A^2A
Stop Barcode: ^1B^5B^3F0l

"!" marks as taking-over character the position of the variables in the output string, ^2A must be defined before and after the variable data again, as they are part of the trigger. A further ^2A is necessary, because a ^2A marks the end of the variables for the StringChange function.

Stop Barcode: ^1B^5B^3F^30^6C as termination of the variable barcode data.

Vertical linefeeds: ^0A^0A^0A, so that the following data are printed into the correct line. Just in case of barcodes the active print position after the barcode is differently and depends on the emulation and becomes here balanced with linefeeds ^0A. Complete replacement function:

Trigger 1:7

Trg 1: 7,"^2A?^2A^19"
Out 1: "^1b^5b^3b^31^30^33^3b^34^3b^31^3b^31^3b^31^3b^20^7a"\
 "^1B^5B^3F0h^2A!^2A^2A^1B^5B^3F^30^6C^1B^5B^3F0l ^0A^0A^0A"

Example:

Example of filtering a data string with variable data. The variable data must be clearly enclosed with well-known characters.

Trigger 1:7

Trg 1:7 "^0D^0A?^20"
Out 1: "^0D^0A^20"

4.9 Mode 8

Mode 8 defines the taking-over of one or several variable characters with one character per taking-over character ("!").

In the INIT file, two triggers for MODE 8 are defined. Trigger 1 is to take-over the first four variable characters following the two characters <esc> (in the output (Out). However, the trigger is valid only if after the variable data an <esc> is found again.

Trigger 1:8

Trg 1: 8,"^1B(???^1B"
Out 1: "!!!!"

Trigger 2 is to take-over the first three variable characters after an < esc > to the output followed by the fixed characters Drucker. Here too the trigger is valid only if after the variable data an < esc > is found.

Trigger 2:8

Trg 2: 8,"^1B???????^1B"

Out 2: "!!! Drucker"

In the input file, the following data are contained:

```
<esc>(TESTTESTTESTTESTTESTTESTTEST<esc>
```

```
<esc>PSIPSI<esc>
```

<esc> stands for the character escape with the hexadecimal code 1B. The output (output file) is after transformation by the definitions in the INIT file:TEST

PSI Drucker

In the following example, a positioning instruction with a one-digit parameter is converted into a graphics function. The parameter of the positioning command has the unit 1/72 inch. In the command set of the PSI - printer one can use the graphics command of the IBM emulation ESC * m n1 n2 with m = 5, which prints graphics databased on a horizontal resolution of 1/72 inch. N1 and n2 define the number of the graphics bytes. For n1 the parameter is taken-over directly from the trigger, followed of a maximally possible number of NUL- codes ^00. The NUL- codes work like a horizontal positioning with the number, which is defined by the parameter n1. NUL- codes exceeding the number of graphics bytes are interpreted as NUL codes by the printer and are ignored. In the example the maximally possible number of 255 characters for an OUT- string is reached.

Example 1:

Trg 10: 8,"^1D?"

Out 10: "¹B²A⁰⁵!⁰⁰"\

[illegible]

Looked at it more closely one will state that positioning commands with a parameter > ^FA does not spend sufficient ^00 codes. This is due to the limited number of max. 255 characters of an output string. To bypass this limitation one can use here for example a mode 0 - function, aimed to convert the not covered parameters ^FB to ^FF.

Example 2:

Trg 10: 0,"^1D^FB"

Out 10: "¹B²A⁰⁵FA⁰⁰"\

[illegible]

Here the parameter of the horizontal parameter is not taken-over with a variable, but with a constant. Thereby a small positioning error is accepted, however, which is to be elected in practice. Appropriate definitions are to be repeated for the parameters ^FC to ^FF in the INIT- file. If one would do without definitions for the parameter range ^FB to ^FF, then the printing would be substantially disturbed with the occurrence one of the parameter values.

5 General Comments

Generally applies:

That trigger with most taken up data is valid. If several triggers with the same data and quantity of data are active and have match then that is valid, which is first in the INIT file. If same triggers with certain fixed characters and variable characters are defined, then the triggers with the fixed characters shall be arranged before the trigger with the variable characters (Joker) in the INIT file.

Working with a STGCHG.ini- File

5.1 Content of the STGCHG_D.ini- File

```
##*, Version-Output: <Fri Apr 29 2005>
##*, Version-Output: 14.10.2005
# Init-File: STGCHG.ini
# Output-File: STGCHG.mot
# Test one - Metal - EXAMPLE
# Mode 0: Standard, 1 Character/Joker, no taking over
# Mode 1: Bistable, toggle Output-String
# Mode 2: Joker, output only, no taking over
```

```

# Mode 3: Repeat, repeats the preceding character of the trigger x-times
# Mode 4: Filter, removes the trigger, no output
# Mode 5: Bistable name, defines the name of the output
# Mode 6: Counter, x- times the output
# Mode 7: Variable with variable taking over
# Mode 8: Variable with one character per taking over character
# HEX_ESC: Hex-prefix for ASCII-Init and versions- output
#     range: ^00..^FF, special characters only is recommended
# LINE_ESC: line extension for ASCII-Init and versions- output
#     range: ^00..^FF, special characters only is recommended
# UDE_CHAR: Hex-prefix for the data stream, used in the UDE-module
#     range: ^00..^FF, special characters only is recommended
# UDE_MODE: chosen UDE- mode, values: N/H/B/D (Non, Hex, Block, Duo)
# JOKER1: in place of the Joker in the trigger any character can be used
#     range: ^00..^FF
# JOKER2: in place of the Joker in the trigger any character can be used
#     range: ^00..^FF
#JOKERBREAK: this character in the data stream terminates a trigger while a joker is active
#     ^00=inactive, Range: ^00..^FF
#UEBERNAHME: this character in the output: the characters are taken over from the input data
#     stream beginning with the position of the Joker 1 in the trigger
#     range: ^00..^FF
HEX_ESC    ^^    # Binary-Init-File: ^0B[^5E], default ^5E
LINE_ESC   \     # Binary-Init-File: ^0C[^5C], default ^5C
UDE_CHAR    ^^    # Binary-Init-File: ^09[^5E], default ^5E
UDE_MODE    D     # Binary-Init-File: ^08[^44], default ^44
JOKER1      ?     # Binary-Init-File: ^04[^3F], default ^3F
JOKER2      ^00   # Binary-Init-File: ^05[^00], default ^00
JOKERBREAK  ^00   # Binary-Init-File: ^06[^00], default ^00
UEBERNAHME  !     # Binary-Init-File: ^07[^21], default ^21
NOT_UDE     "^^CR"
NOT_UDE     "^^EX"
NOT_UDE     "^^AB"
NOT_UDE     "^^34"
Trg  1: 0,"^1B[0;9;180;27;18;96;0;0;0'q" # change Barcode header
Out  1:  "^1B[;101;4;2;1;; z"
Trg  2: 0,"^1B% 0" # Barcode start
Out  2:  "^1B[^3F0h*"
Trg  3: 0,"^1B%@" # Barcode stop
Out  3:  "*"^1B[^3F0|"
Trg  4: 7,"^1B%???????^1B%@"^1B[;;;;;;;;;2'q" # delete second Barcode Line
Out  4:  " "
etc.....

```

The examples Trg/Out can be deleted or be shifted as comment lines to the end of the STGCHG.ini file. Before the StringChange definitions (Trg/Out)) no comment line (#....) may be entered. Blank lines may not be entered between the lines Trg and Out. The file is worked on only with a standard ASCII character editor.

5.1.1 Test StringChange- Definition

After the file, STGCHG.ini with the desired definitions is finished, and a file with the input data stream of the application is present (STGCHG.inp) the first test of a conversion to the output file STGCHG.out can take place:

Both files are in a directory with the file TESTSTG.bat.

Start TESTSTG.bat and generate the output file STGCHG.out. The contents of a TESTSTG.bat – file:

```
REM TEST THE STRINGCHANGE DEFINITIONS:
```

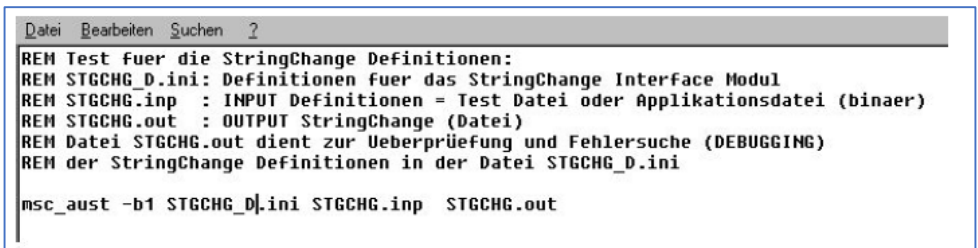
```
REM STGCHG_D.ini: Definitionen fuer das StringChange Interface Modul
```

```
REM STGCHG.inp : INPUT Definitionen = Test Datei oder Applikationsdatei (binaer)  
REM STGCHG.out : OUTPUT StringChange (Datei)  
REM Datei STGCHG.out dient zur Ueberprüfung und Fehlersuche (DEBUGGING)
```

```
REM COMPARE THE STGCHG.out WITH YOUR DEFINITIONS FOR DEBUGGING
```

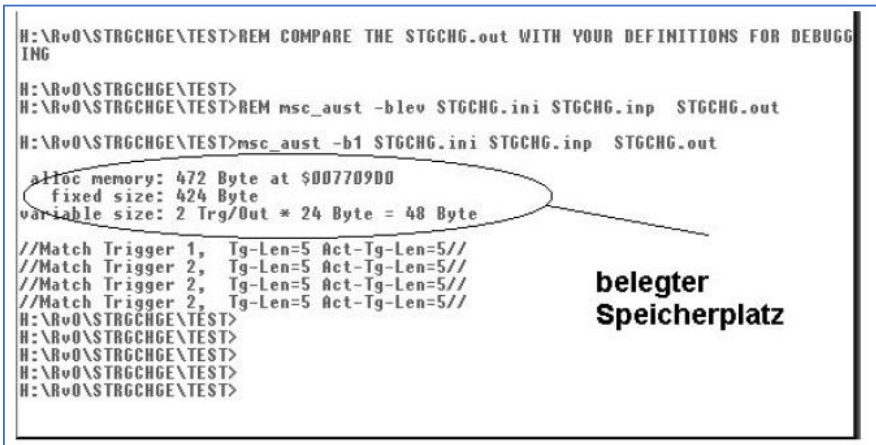
```
msc_aust -b1 STGCHG.ini STGCHG.inp STGCHG.out
```

The illustration shows a DOS window after a successful run of TESTSTG.bat with a small STGCHG_D.ini - file.



```
Datei Bearbeiten Suchen ?  
REM Test fuer die StringChange Definitionen:  
REM STGCHG_D.ini: Definitionen fuer das StringChange Interface Modul  
REM STGCHG.inp : INPUT Definitionen = Test Datei oder Applikationsdatei (binaer)  
REM STGCHG.out : OUTPUT StringChange (Datei)  
REM Datei STGCHG.out dient zur Ueberprüfung und Fehlersuche (DEBUGGING)  
REM der StringChange Definitionen in der Datei STGCHG_D.ini  
  
msc_aust -b1 STGCHG_D.ini STGCHG.inp STGCHG.out
```

The figure shows a DOS window after a successful run of TESTSTG.bat with a small STGCHG_D.ini - file.



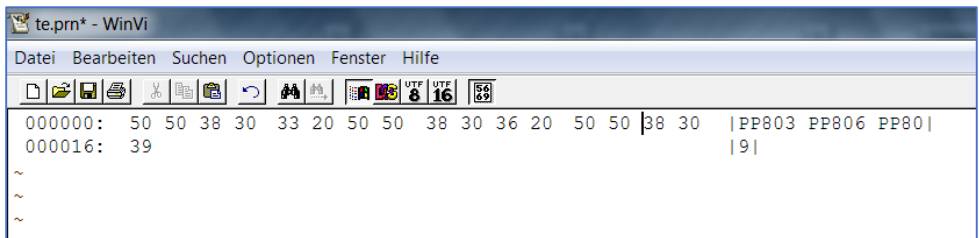
```
H:\Rv0\STRGCHGE\TEST>REM COMPARE THE STGCHG.out WITH YOUR DEFINITIONS FOR DEBUGG  
ING  
H:\Rv0\STRGCHGE\TEST>  
H:\Rv0\STRGCHGE\TEST>REM msc_aust -blev STGCHG.ini STGCHG.inp STGCHG.out  
H:\Rv0\STRGCHGE\TEST>msc_aust -b1 STGCHG.ini STGCHG.inp STGCHG.out  
  
alloc memory: 472 Byte at $00770900  
fixed size: 424 Byte  
variable size: 2 Trg/Out * 24 Byte = 48 Byte  
  
//Match Trigger 1, Tg-Len=5 Act-Tg-Len=5//  
//Match Trigger 2, Tg-Len=5 Act-Tg-Len=5//  
//Match Trigger 2, Tg-Len=5 Act-Tg-Len=5//  
//Match Trigger 2, Tg-Len=5 Act-Tg-Len=5//  
H:\Rv0\STRGCHGE\TEST>  
H:\Rv0\STRGCHGE\TEST>  
H:\Rv0\STRGCHGE\TEST>  
H:\Rv0\STRGCHGE\TEST>  
H:\Rv0\STRGCHGE\TEST>
```

belegter Speicherplatz

Alloc memory: = used space

With a HEX EDITOR now, the implementation of the definitions in the StringChange- STGCHG.out file to be checked.

This diagram shows the contents of a STGCHG.out - file using a hex editor.



Additionally the file STGCHG.out can be transferred to the printer, in order to examine the correct print. Before it must be paid attention to the correct menu setup of the printer regarding emulation etc.

If the result does not correspond to expectations, then the StringChange definitions in the file STGCHG.ini are to be examined and corrected.

5.1.2 Load StringChange- Definition into the Personality Module

After the StringChange definitions was checked for correct function, now the download can take place into the Personality Module. The StringChange functions are loaded with a file STGCHG.mot into the Personality Module. For the creation of this file start mkSTGCHG.bat. The file STGCHG.ini must be in the same directory. Dependent on the used Personality Module the correct entry of an address in the file mkSTGCHG.bat must be checked or adapted beforehand

For a Personality Module without IGP emulation the address 8050000, for a Personality Module with IGP emulation the address 81E0000 must be defined.

The contents of the mkSTGCHG.bat - file.

REM Version mkSTGCHG.bat 2005.08.11

REM Make of file STGCHG.mot for download into the StringChange- Interface Module.

REM Prior to download depress and hold the push button of the PM and power- on the printer.

REM StringChange only (without IGP) : the address must be set to 8050000

REM StringChange in combination with IGP: the address must be set to 81E0000

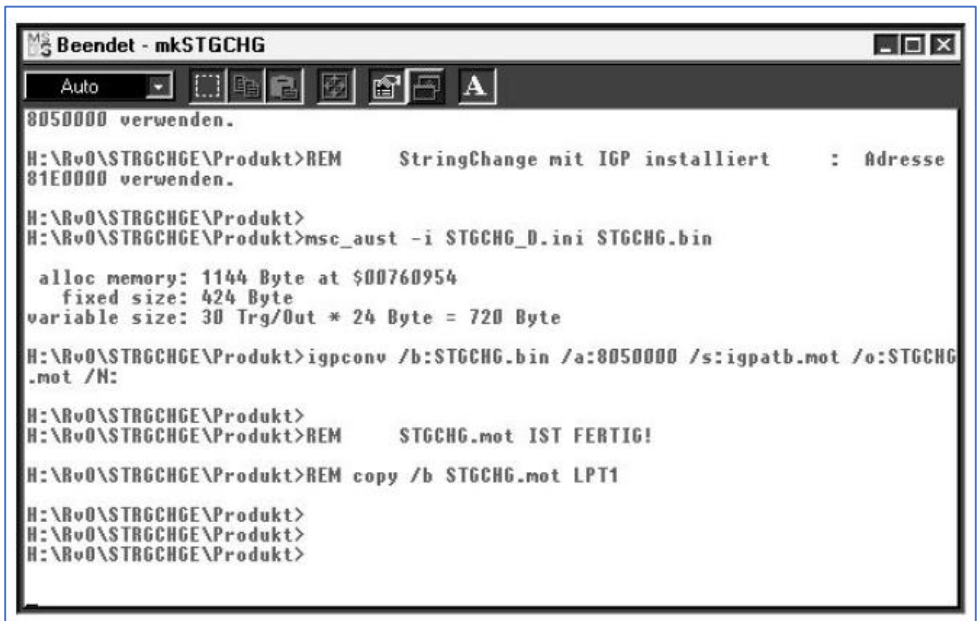
msc_aust -i STGCHG.ini STGCHG.bin

igpconv /b:STGCHG.bin /a:8050000 /s:igpatb.mot /o:STGCHG.mot /N:

REM STGCHG.mot IS READY

REM copy /b STGCHG.mot LPT1

The illustration shows a DOS window after successful conversion of a STGCHG.ini file into a STGCHG.mot file.



```
Beendet - mkSTGCHG
Auto
8050000 verwenden.
H:\Rv0\STRGCHGE\Produkt>REM      StringChange mit IGP installiert      : Adresse
81E0000 verwenden.
H:\Rv0\STRGCHGE\Produkt>
H:\Rv0\STRGCHGE\Produkt>msc_aust -i STGCHG_D.ini STGCHG.bin

  alloc memory: 1144 Byte at $00760954
    fixed size: 424 Byte
variable size: 30 Trg/Out * 24 Byte = 720 Byte

H:\Rv0\STRGCHGE\Produkt>igpconv /b:STGCHG.bin /a:8050000 /s:igpatb.mot /o:STGCHG
.mot /N:
H:\Rv0\STRGCHGE\Produkt>
H:\Rv0\STRGCHGE\Produkt>REM      STGCHG.mot IST FERTIG!
H:\Rv0\STRGCHGE\Produkt>REM copy /b STGCHG.mot LPT1
H:\Rv0\STRGCHGE\Produkt>
H:\Rv0\STRGCHGE\Produkt>
H:\Rv0\STRGCHGE\Produkt>
```

The file STGCHG.mot can be transferred now e.g. by means of COPY - instruction over the parallel interface to the printer. Switch for this the printer off and restart with pressed push button () the printer. The green LED next to the push button shines.

Subsequently, transfer file STGCHG.mot to the printer.

E.g. COPY /b STGCHG.mot LPT1 with connection to a parallel interface or with connection to an Ethernet interface by means of FTP.

Note:

After the download of a StringChange function, the Personality Module can be installed into another printer, without lost of the StringChange definitions.

5.1.3 Reset of the StringChange- Definition

If all StringChange definitions is to be reset, then depending on the model of the Personality Module one of the files Emu0000.mot (IGP emulation) or STGCHG0000.mot (StringChange only) is to be transferred to the printer. Prior to the transfer, the printer must be switched on with pressed push button ().

Old StringChange definitions becomes always invalid, if a new file STGCHG.mot is loaded

5.1.4 Checking the Installed StringChange- Functions by a Printout

In principle a printout of the installed StringChange definitions can be released with no function at the printers (e.g. depressing a key or through the menu). This measure protects against unwanted access to the installed StringChange definitions. So that however a control of the installed functions is possible, the tool package includes a particularly coded file `print_stgchg_ver.txt`, which when transferred to the printer causes a printout of the definitions.

5.1.5 Comments on Updating the Printer Firmware

StringChange functions can change the firmware - code, if coincidentally trigger conditions emerge and then are converted into other code. Then the firmware is usually not executable or incorrect. One can go around this problem, if the PM- basic board is detached from the IGP/ATB board inserted in the printer. The firmware - update is made then by a special parallel cable connected at the internal parallel interface of the PM- basic board.

Alternatively, one can switch-off the StringChange function before the firmware update by application of the file `STGCHG0000.mot`. However, after firmware update the StringChange functions must be installed again.

5.2 Technical Data

- The number of StringChange definitions is 32767 maximally
- The number of bytes per trigger (input) – string is 255 bytes maximally
- The number of bytes per Out (output) – string is 255 bytes maximally
- The number of characters of triggers/outputs of pairs is 65520 bytes maximally

*) All maximum values are limited by the actually available storage capacity practically one can count on the fact that the number of trigger/output strings of a very complex application does not lie over 1000 definitions.