



1. Introduction

The ComGage test step function “Device Control by RS232 / TCP/IP” allows to send commands to a device (e.g. measuring instrument, motor, controller, PLC, ...) which is connected via a COM port or TCP/IP port.

Additionally, a response string can be received from the device and written into a TXT file.

The commands have to be listed in a command file (file extension *.prg).

Important notes :

The software license 72 is required to execute this test step function !

2. Command file

The name of the command file has to be : <Name of the test scheme>.prg

The command file has to be placed in the ComGage data directory for test orders.

File structure

a) Comment lines :

The first character of a comment line is a semicolon :

Example :

```
;=====
;Command list for test scheme >>xxx<<
;
; Version :   V1.00
; Date    :   16.12.2007
;=====
```

b) Lines with configuration settings (e.g. COM port, Baud rate, ...) :

Configuration settings are identified by square brackets.

Syntax : [<Parameter> <Value>]

• for COM ports :

<Parameter> : C	➔	COM port
B	➔	Baud rate
P	➔	Parity
D	➔	Number of data bits
S	➔	Number of stop bits
T	➔	Timeout time
E	➔	(= time, by which the device must have sent a response)
	➔	ComGage result register, optional
	➔	(outputs the result of a command execution in a register
	➔	(see 2. d)) instead of message box display)

<Value> :	for C	➔	1...8
	for B	➔	1200, 2400, ..., 115200
	for P	➔	No, Even, Odd, Space, Mark
	for D	➔	7 or 8
	for S	➔	1 or 2
	for T	➔	Time in msec
	for E	➔	0 (= off) / 1 (= register 1) / ... / 2000 (= register 2000)



Example for COM port :

```
; COM port = COM2
[C2]
; Baud rate = 9600
[B9600]
; Parity = no
[PN]
; Number of data bits = 8
[D8]
; Number of stop bits = 1
[S1]
; Timeout time = 10000 msec
[T10000]
```

• for TCP/IP ports :

<Parameter> : I	➔	IP address + port (If IP address is set, the COM port is not used.)
T	➔	Timeout time (= time, by which the device must have sent a response)
E	➔	ComGage result register, optional (outputs the result of a command execution in a register (see 2. d)) instead of message box display)

<Value> :	for I	➔	<IP address>:<Port>, e.g. 127.0.0.1:27015
	for T	➔	Time in msec
	for E	➔	0 (= off) / 1 (= register 1) / ... / 2000 (= register 2000)

Example for TCP/IP port :

; IP address = 127.0.0.1:27015	➔	Because of this, the COM port settings are ignored.
[I127.0.0.1:27015]		
; Timeout time = 10000 msec		
[T10000]		

c) Command lines :

Every command line contains the command line number, the command itself and the expected command response from the device.

Syntax : < Line number>:<Command>><Response>

<Line number> : A number between 0 ... 32767
The software automatically sorts the commands by numbers.
(The first command does not have to have the number 0.)

<Command> : The command string can consist of a mixture of ASCII characters and ASCII codes.
The ASCII characters are enclosed in “ ”. The ASCII code is entered as a number. Furthermore, additional placeholders for the output of register values are available.
(Example “Placeholder for register 5” : {R5})
All elements are separated by commas.

Examples :

- For the output of **Move75<cr>Speed72<cr><lf>**, the command string looks like this : “Move75”,13,“Speed72“,13,10
- For the output of register value of R6 as a parameter of the move command, the command string looks like this : “Move”,{R6},13, ...

<Response> : The response string is structured exactly like a command string.
If no response string is given in a command line, no response from the device is expected.



Example : (*Note : A command response is not expected in line 1.*)

```
; Zero adjustment of the system
1:"CLR",13>
;
; Move relative 1000 steps and expect response "OK".
2:"MOVR 1000",13,10>"OK",13,10
```

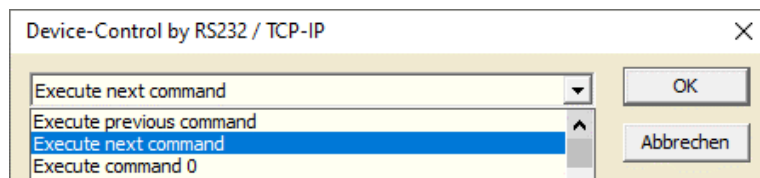
d) Result register :

If a result register was activated (see 2. b)), the result of the function execution will be output into this register. The register can have the following result values :

- 0 = No error
- 1 = Selected command is not available
- 2 = No response was received
- 3 = Error message received
- 10 = License is missing !
- 11 = File with command list not found
- 12 = File access error
- 13 = Syntax error in command line
- 14 = Syntax error in line with configuration settings
- 15 = Error opening the COM port
- 16 = IP address / port not configured correctly
- 17 = TCP connection could not be established
- 18 = Error in IP address or host name

3. Configuration

The test step function is created within a test step. Clicking the Setup button opens the following dialogue :



The dialogue provides the following configuration settings :

Execute next command

When the function is called, the command with the next higher line number is executed.

If no command from the list was executed before, the command with the lowest line number is executed.

If the command with the highest line number was executed before, the command with the lowest line number is executed next.

Execute previous command

When the function is called, the command with the next higher line number is executed.

If no command or the command with the lowest line number was executed before, the command with the highest line number is executed next.

Execute command 0 ... 32767

When the function is called, the command with the selected line number is executed.

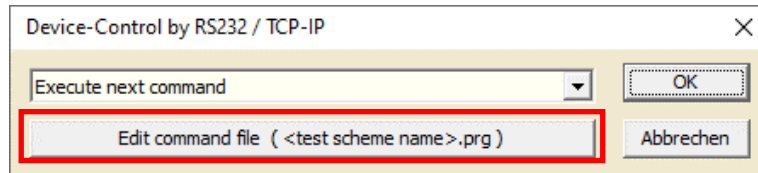
Store received data string in TXT-file

If a new data string has been received, this string is stored in a new TXT file.

The TXT files are created in the ComGage data directory for test orders.

The file name is composed like this : <Test order name>_<Date><Time>.txt

The button “**Edit command file (<test scheme name>.prg)**” opens the command file related to the current test scheme in an editor :



4. Examples

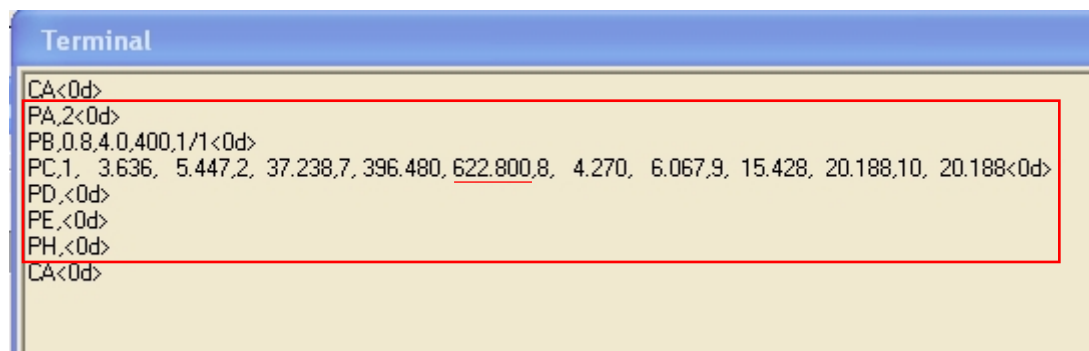
- Example 1 : controlling a Faulhaber motor

```
; switch on the motor
1:"EN",13>"OK",13,10
; switch off the motor
2:"DI",13>"OK",13,10
; move the motor with speed 500
3:"V500",13>"OK",13,10
; stop the motor
4:"V0",13>"OK",13,10
; move the motor to position 8000 with speed 500
5:"LA8000",13,"SP500",13,"M",13>"OK",13,10
```

- Example 2 : Receive measuring values from a Zeiss measuring instrument

```
; The following command starts the measurement :
1:"CA",13>
```

Upon this, the following string (shown here in a terminal) is written into a text file :



The required value (here underlined) can now be read from the TXT file by the WGL018 as measuring value of a characteristic.

The “PC” at the beginning of the line can be used to identify the correct line in the file.

The commas can be used as separators and column 9 can be defined as the column in which the measuring value can be found.

It is even possible to assign the values from different columns to different characteristics in the test scheme.

A detailed description of the WGL018 can be found in the related documentation.