

1. Introduction

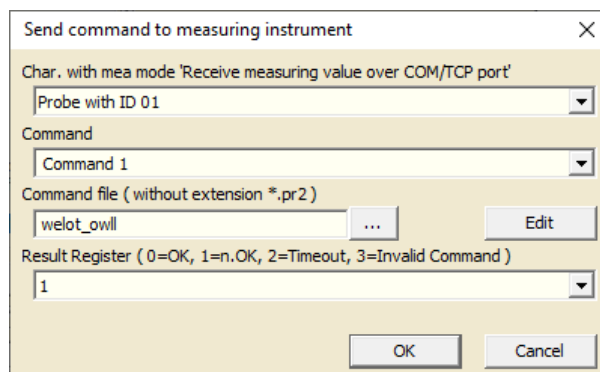
The test step function *Send command to measuring instrument* can be used to send a command to a measuring instrument that is used by a characteristic with the ComGage Special measuring mode *Receive measuring value over COM/TCP port* (wgl029).

Important notes :



- This test step function can only be used together with the chargeable Special measuring mode *Receive measuring value over COM/TCP port*, because the command is sent via the connection of this Special measuring mode to the measuring instrument.
- In order to be able to configure this test step function, characteristics which use the Special measuring mode have to be created first.
- The commands have to be entered in a command file (file extension *.pr2). This command file has to be placed in the ComGage data directory for test schemes.
- This function depends on the use of the wgl029 *Receive measuring value over COM/TCP port* as measuring mode for the selected characteristic. If changes are made to the characteristic, this function has to be adjusted accordingly.
- This test step function can send commands only via a COM port.

2. Configuration

The function is created in in a test step. By pressing the *Setup* button the following dialogue can be opened :



The dialogue offers the following configuration options :

- **Char. with mea mode 'Receive measuring value over COM/TCP port'**
Here the characteristic with measurement mode “Receive measuring value over COM/TCP port” has to be selected.
- **Command**
Here the number of the required command in the command file has to be selected (line number → see 3. c) ..
- **Command file (without extension *.pr2)**
File name of the command file (without extension). The command file has to be placed in the ComGage data directory for test schemes. With the button  a command file can be selected. The button  opens the command file in an editor.
- **Result Register**
The result of the function execution is written into the selected result register.
Possible values are :
 - (0) Command successfully executed
 - (1) Execution of the command was not successful
 - (2) Timeout during execution of the command
 - (3) Command format invalid



3. Structure of the command file :

a) Comment line :

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 :

Configuration settings are marked by square brackets.

Syntax : [<parameter><value>]

<parameter> : T ➔ timeout time (currently, only this parameter is available)
 = time until the instrument must have sent an answer

<value> : ➔ time in msec

Example :

```
; Timeout time = 5000 msec
[T5000]
```

c) Command line :

Every command line contains the number of the command line, the command and the expected answer from the instrument.

Syntax : <line number>:<command>><answer>

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

<command> : The command string can consist of ASCII characters and
 ASCII codes.
 The ASCII characters are enclosed in quotation marks (" ").
 The ASCII code is entered as numbers.
 Additionally, placeholders can be used for the insertion of
 register values.
 (Example „placeholder for register 5“ : {R5})
 All elements are separated by commas.

Example :

- For the output of **Move75<cr>Speed72<cr><lf>**
the command string would be : "Move75",13,"Speed72",13,10
- For the output of the register value of R6 as parameter of the Move
command, the command string would be : "Move",{R6},13, ...

<answer> : The answer string has the same structure as the command string.
 If no answer string is entered, no answer from the measuring instrument is
 expected.

Example :

```
;
; Command for disabling RS485 lock on sensor 1
1:":01W010;0;E9C3",13,10>"01a;89EE",13,10
```