Reliance – External Communicator

Description of the functions and parameters

Version 1.7.1

***GEOVAP, spol. s r.o.**,Čechovo nábřeží 1790, 530 03 Pardubice*
tel: +420 466 024 617, fax:+420 466 210 314, e-mail: reliance@geovap.cz, http://www.geovap.cz

Den zápisu do OR: 26. 2. 1991. Spisová značka: C.234 vedená u rejstříkového soudu v Hradci Králové.
IČO 15049248, DIČ CZ15049248

1. Introduction

External Communicator is intended for communication to devices of a simple communication protocol via the serial communication port. The program is controlled through its COM interface's procedures and functions called from scripts defined in a Reliance visualization project.

2. Instructions for use

2.1. Installation

The program is installed along with the Reliance SCADA/HMI system. Before first use, you should run the program manually from the Reliance root directory to register it with Windows.

2.2. Configuration

The configuration is stored in the initialization file *„Reli_Comm.ini".* You can change the settings there. If the file does not exist yet, it is created while shutting down the program for the first time. The parameters are separated into the following sections.

```
[MAIN]
ShowMessages=1
ClosePortAfterGetData=1
[DATA]
SaveDataToTxtFile=1
DataFileName=c:\Data\TestData.txt
```

The „MAIN" section

ShowMessages –determines whether a warning dialog box should appear if there is a problem. For example, opening the port failed because the communication port does not exist.

ClosePortAfterGetData – determines whether the communication port should be closed automatically after the data has been sent. In this case, the port is closed after the predefined timeout after sending the data.

Language – the language code – 0..Czech, 1.. English

The „DATA" section

SaveDataToTxtFile – determines whether the driver should save the received data to a text file

DataFileName – full name of the text file

2.3. Use with Reliance SCADA/HMI System

The program requires a license stored in the HW key shipped with Reliance. Without the license, the program runs as Demo version. There is no functionality restriction in the Demo version. Only the warning dialog appears at startup and every 30 minutes.

The program can be controlled from scripts defined in a Reliance visualization project. There is example of starting the program from a piece of VBScript code.

Example:

```
Dim Comm
Set Comm = CreateObject("Reli_Comm.Reli_Comm")      'allocates the object
Comm.OpenPort 1,38400,2,8,1                         'opens the communication port
Comm.SendData Data,14,1000                          'sends the data
```

The program is freed from memory automatically when the Reliance's runtime software terminates. If there is a need to free the program from memory earlier, use the following code in a script.

Example:

```
Set Comm = Nothing          'freed program from memory (program shutdown)
```

3. Description of functions

3.1. OpenPort Function

The function opens the communication port and sets the parameters. The function returns the result FALSE if it failed or TRUE if it succeeded. If the function ***OpenSocket*** have been called before then the procedure ***CloseSocket*** should be called to open the communication port successfully.

Parameters:

1. Serial communication port number [integer]
2. Baud [integer]
3. Parity [integer]
   - 0 – none
   - 1 – odd
   - 2 – even
   - 3 – log 1
   - 4 – log 0
4. Data bits [word]
5. Stop bits [word]

Example:

```
PortIsOpen = Comm.OpenPort (1,38400,2,8,1)
'opens the serial comm. port COM1, baud 38400, parity Even, 8 data bits, 1 stop bit
```

3.2. OpenSocket Function

The function opens the TCP socket with the preset parameters. The function returns the result FALSE if it failed or TRUE if it succeeded. If the function ***OpenPort*** have been called before then the procedure ***ClosePort*** should be called to open the socket successfully.

Parameters:

1. Port number [integer]
2. IP address [string]
3. Socket type 0 – TCP [integer]
   1 – UDP (not implemented)

Example:

SocketIsOpen = Comm.OpenSocket(4000,"127.0.0.1",0)
'opens the TCP socket port 4000, IP address 127.0.0.1

3.3. SendData Function

The function clears the input buffer and sends the data. If the option of saving data to text file is activated then the driver saves the received data to the text file “DATA.TXT” after the preset timeout. The data is separated by a space character. On the next call to the function the file is rewritten.

Parameters:

1. Variable that contains data to send. Maximum size is 4096 bytes (chars).
   [array of byte] or [string]
2. Number of bytes to be sent. If the variable is string then the parameter is ignored. The number equals to string length in this case.
   [integer]
3. Timeout for saving the received data to text file in milliseconds.
   [integer]

Example:

```
Dim OutBuffer                  ' declaration of the variable OutBuffer
OutBuffer = Array(0,1,2,3,4)   ' fills the variable with data
fcomm.SendData OutBuffer,5,200 ' sends 5 bytes, timeout 200 ms
```

3.4. GetData Function

The function returns received data and received data count. The function should be called after the time sufficient to receive all the desired data.

Parameters:

1. Variable that will contain received data. Maximum size is 4096 bytes (chars).
   [array of byte] or [string]

Example:

```
Dim InBuffer                   ' declaration of the variable InBuffer
ReDim InBuffer(50)             ' sets the variable length to 50 bytes
Dim Count                      ' declaration of the variable Count
Count = fcomm.GetData(InBuffer) ' removes received data from input buffer of the driver into the
                               ' variable
```

3.5. ClearInBuf Procedure

Procedure clears the input buffer of the communication driver. The buffer is automatically cleared on every call to the ***SendData*** function.

Parameters:
none

Example:
fcomm.ClearInBuf ***'clears the input buffer***

3.6. ClosePort

The procedure closes the serial communication port. The port can be used by any other application. The function OpenSocket can be called too since this procedure have been called.

Parameters:
none

Example:
fcomm.ClosePort ***'closes the serial communication port***

3.7. CloseSocket

The procedure closes the socket. The socket can be used by any other application. The function OpenPort can be called too since this procedure have been called.

Parameters:
none

Example:
fcomm.CloseSocket ***'closes the serial communication port***

3.8. Hide Procedure

The procedure hides the communication driver into the system tray. It is possible to invoke the program's user interface again by clicking the left mouse button on the icon in the tray.

Parameters:
none

Example:
fcomm.Hide ***'hides the communication driver into the tray***

3.9. SetLanguage Procedure

The procedure changes the language of the program.

Parameters:
1. Language number. [integer]
   0 – Czech
   1 – English

Example:
```
fcomm.SetLanguage 1                    ' sets the english language
```