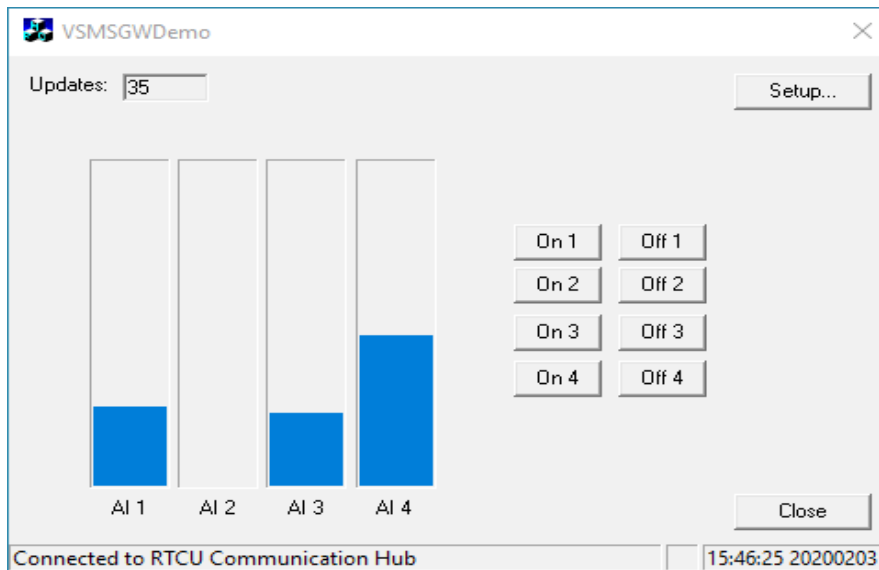


The Logic IO

RTC Communication Hub Demonstration Package

Version 3.01



The screenshot shows the RTCU IDE environment. The main window displays the source code for "test.vpl", which includes comments and code for handling RTCU communication. A "Debug Messages" window is open, showing a log of messages such as "Sending message=253, 0, 0, 411" and "Forcing a send". A "Gateway settings" dialog box is open in the foreground, showing fields for IP, Port, and Key. On the right side, there are two panels: "Devices" and "IO". The "Devices" panel shows a list of inputs and outputs, with "Onboard" selected. The "IO" panel shows analog input and output values, such as "a[1] 253" and "a[4] 475". At the bottom right, there is a "Project Devices" window showing the project configuration, including the device name "AX9 turbo Device 1" and the application "test v0.01". The status bar at the bottom indicates "Connect time: 00:00:52", "APP: test_0.01", and "SER: 296999123 FW: 5.06 TRIG: AX9 turbo (Emulated)".

Table of Content

Introduction.....	3
Contents of package.....	4
How to install and run the demo application.....	5
Short introduction to the Logic IO RTCU Communication Hub.....	8
Introduction to the Library.....	10
Functions in the VSMMSGW Library.....	11
Return Codes:.....	11
cbfuncText().....	11
cbfuncPDU().....	12
cbfuncPackage().....	13
vsmsgwInit().....	14
vsmsgwInitAdv().....	15
vsmsgwSetSMSTextCB().....	15
vsmsgwSetSMSPDUCB().....	16
vsmsgwSetDataPackageCB().....	16
vsmsgwIsConnected().....	16
vsmsgwGetMyNodeID().....	17
vsmsgwSendSMS().....	17
vsmsgwSendPDU().....	18
vsmsgwSendDataPackage().....	19
Description of the demo project.....	20

Introduction

This document describes the details of a practical RTCU Communication Hub application, demonstrating communication between a RTCU device (alternatively a RTCU Emulator) and a PC server-like application.

If you follow this document you will be able to install and test the demo application, and see for yourself, that the deployment of M2M technology is easy and straightforward when using Logic IO products!

This demo application transfers the value of the up to 4 analog inputs of the RTCU device to a PC application every 10 seconds (or if one of the analog inputs changes significantly). The PC application can switch one of 4 digital outputs either on or off, on the RTCU device.

All the communication is transferred with the help of the RTCU Communication Hub, which simplifies the implementation dramatically. Messages between the PC application and the RTCU device are transferred using VSMS (Virtual SMS) messages.

What's new in version 3.00:

- The VSMSGW library and the PC demo application are now built using Visual Studio 2019.
- Large Packet Support is implemented.

Contents of package

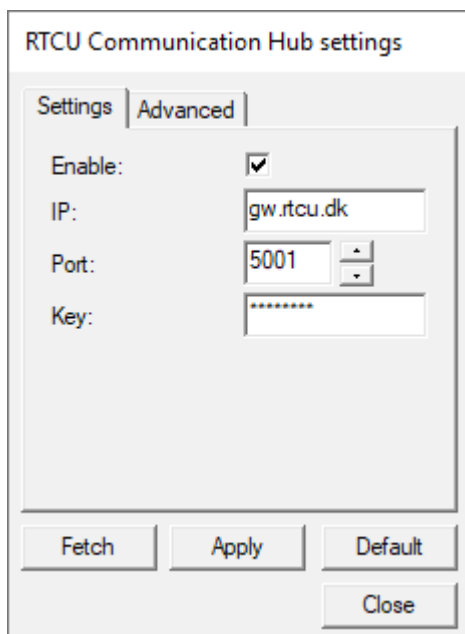
The package this document is part of, contains the following:

“\RTCU Communication Hub Demonstration Package.pdf”	This document
“\PC App\”	The complete PC application
“\RTCU App\”	The complete RTCU application
“\VSMMSGW\”	The DLL containing library functions for the RTCU Communication Hub communication.
“\Library\”	The .H, .LIB and .DLL file for the VSMMSGW library

How to install and run the demo application

Below, you will find a step-by-step instruction how to install and make the demo application run:

- 1) Unpack the contents of the “RTCU Communication Hub Demonstration Package.ZIP” file.
- 2) Start the RTCU IDE program, and connect to the RTCU device. Alternatively the RTCU Emulator can be used.
- 3) Configure the TCP/IP settings for the RTCU device with the correct settings for the actual GPRS enabled SIM card you have installed in the device. Alternatively an Ethernet or Wi-Fi connection may be configured.
- 4) Configure the Communication Hub settings for the RTCU device with the following parameters:



RTCU Communication Hub settings

Settings | Advanced

Enable:

IP: gw.rtcu.dk

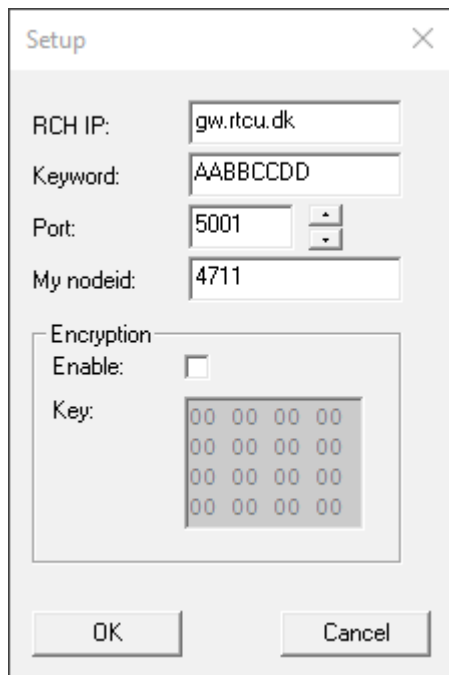
Port: 5001

Key:

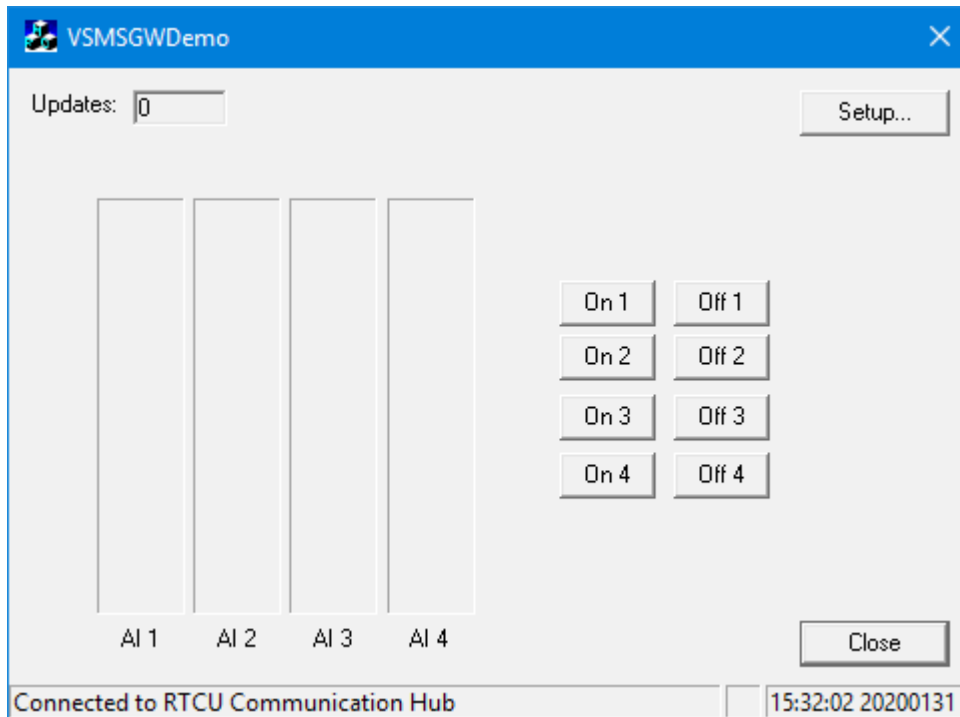
Fetch Apply Default Close

- 5) Using the above parameters the device will connect to the test RTCU Communication Hub that is running at Logic IO. If you decide to install the RTCU Communication Hub in your own environment it can be downloaded for free from www.logicio.com.
- 6) Load the “RTCU App” project in the RTCU IDE
- 7) Upload the project to the RTCU device

- 8) After some tenths of seconds, it should connect itself to the RTCU Communication Hub.
- 9) Then start the PC Application located in the "PC App\Release" directory.
- 10) The default settings in the "Setup" dialog should be sufficient for the program to connect to the test RTCU Communication Hub at Logic IO. Should you decide to install and run your own RTCU Communication Hub, you will have to change the parameters in the Setup dialog. Factory default for the parameters are:



11) After some seconds, the PC application should look something like this:



The field “Updates” shows how many messages that has been sent from the device, and the 4 bar graphs shows graphically the values of the 4 analog inputs. By pressing one of the 8 buttons (On 1...4 and Off 1...4) it is possible to control the 4 digital outputs on the RTCU device.

Short introduction to the Logic IO RTCU Communication Hub

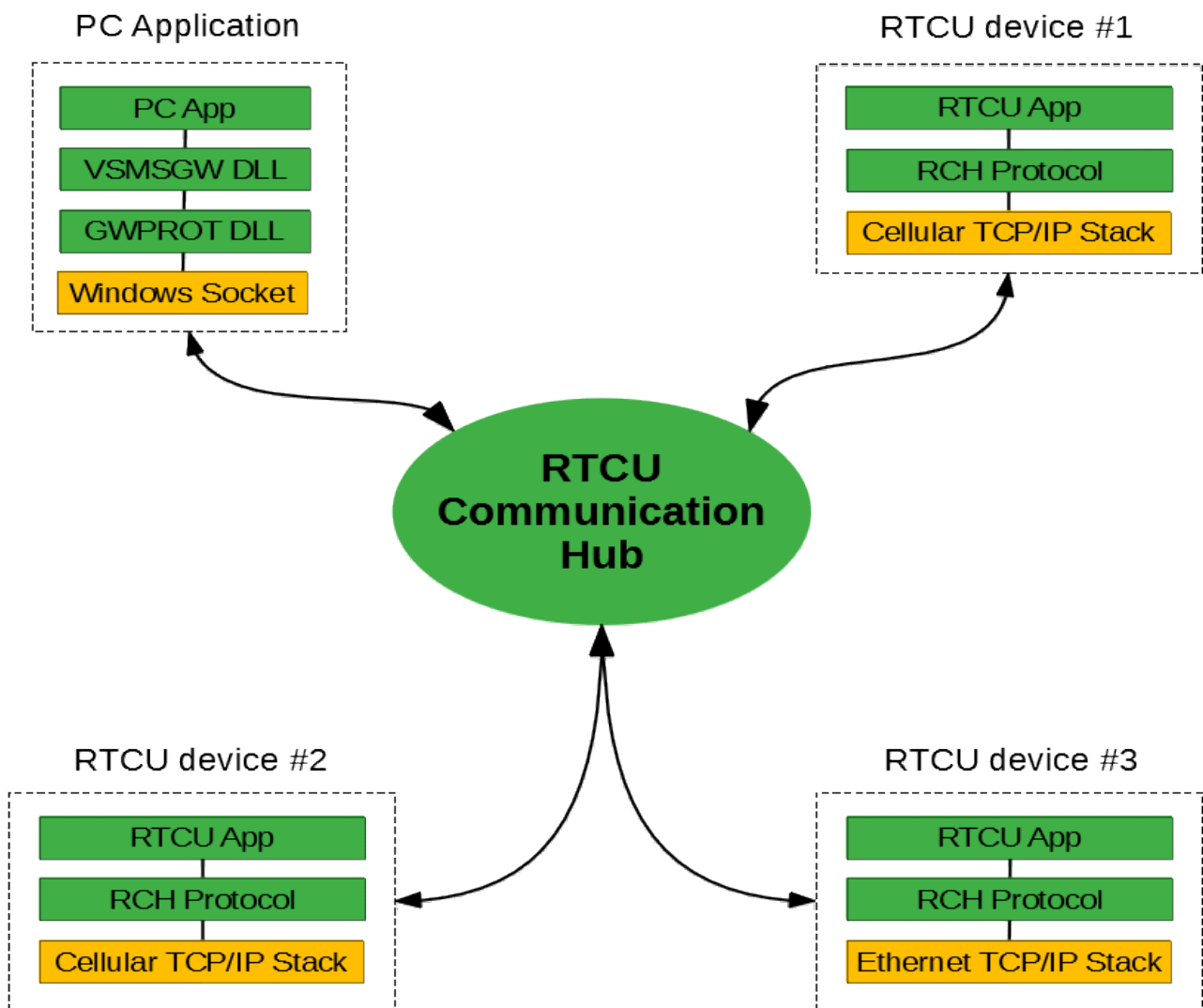
The Logic IO RTCU Communication Hub is a software solution that allows easy communication and access to remote devices connected using the revolutionary wireless technologies such as GPRS and Wi-Fi. The product is especially suited in cases where the GSM operator does not offer a fixed and/or global IP-address. With the Logic IO RTCU Communication Hub product the remote devices can easily be accessed without any need for expensive IP-tunneling solutions provided by the specific GSM-operators.

The RTCU Communication Hub, acts as a kind of “switch-board”, each “client” can connect to the switchboard, and then send messages to other “participants”.

When seen from the RTCU Communication Hub, all connecting parties, either RTCU devices or “PC” applications, are seen as “Clients”. The RTCU Communication Hub can see no difference between them; they are all “just clients of the switchboard”. It is a Client’s node id that makes it unique to the Communication Hub. A node id can be one of two things: if the Client is a RTCU device, the node id of the device is simply the serial number of the device. If the Client is a “PC” application, the node id is a number the application selects (it can be “0”, in that case the Communication Hub assigns it a unique node id).

The GWPROT DLL forms the low-level protocol towards the RTCU Communication Hub. It handles all issues regarding connect/disconnect socket connection, monitoring of line status, and it is responsible for bringing the connection up again, should it be down etc. The VSMSGW Library, uses the GWPROT DLL for the RTCU Communication Hub connection, and essentially just encapsulates some of the functions of the GWPROT DLL, and makes sending and receiving VSMS messages more easy and straightforward.

Schematic presentation of the architecture:



Introduction to the Library

The VSMSGW library is a small library that encapsulates some of the transactions that is possible to make against a RTCU device.

For a description of the complete RTCU Gateway protocol (RACP2), please consult the appropriate document from Logic IO.

The VSMSGW library is written in Microsoft Visual Studio 2019, and was created using the wizard in Visual Studio for creation of a DLL. The VSMSGW project is part of the workspace for the PC Application.

Functions in the VSMSGW Library

The Library is a collection of the following functions described in this section.

Return Codes:

Symbolic name	Value	Description
GWRC_OK	0	Operation successful
GWRC_ERROR	1	Unspecified error
GWRC_NOT_CON	2	Not connected
GWRC_TIMEOUT	3	A timeout occurred
GWRC_INV_LEN	5	Invalid length
GWRC_IS_OPEN	7	Is already open
GWRC_NOT_OPEN	8	Is not open
GWRC_INV_PARM	9	Invalid parameters
GWRC_DST_UNREACH	10	Destination node is unreachable

cbfuncText()

Synopsis

```
typedef int(_cdecl *cbfuncText)(int SenderNodeID, char str[161], void *arg);
```

Description

Definition of callback function for receiving Text SMS messages.

Input :

SenderNodeID	The nodeid of the sender (the serial number of the RTCU device that sent the message)
str	The text string sent by the RTCU device. Max size is 160 characters + a null-terminator.
arg	A user supplied 32 bit variable that was set when vsmgwInit() was called

Output:

rc	This is the return code that should be returned to the RTCU device, 0 if OK, else error
----	---

cbfuncPDU()

Synopsis

```
typedef int(_cdecl *cbfuncPDU)(int SenderNodeID, unsigned char data[140],
int length, void *arg);
```

Description

Definition of callback function for receiving PDU SMS messages.

Input :

SenderNodeID	The nodeid of the sender (the serial number of the RTCU device that sent the message)
data	The data sent by the RTCU device. Max size is 140 bytes.
length	The length of data sent by the RTCU device
arg	A user supplied 32 bit variable that was set when vsmgwlnit() was called

Output:

rc	This is the return code that should be returned to the RTCU device, 0 if OK, else error
----	---

cbfuncPackage()

Synopsis

```
typedef int(_cdecl * cbfuncPackage)(int SenderNodeID, unsigned char data[4064],
int length, void *arg);
```

Description

Definition of callback function for receiving RTCU Communication Hub data packages.

Input :

SenderNodeID	The nodeid of the sender (the serial number of the RTCU device that sent the message)
data	The data sent by the RTCU device. Max size is 4064 bytes.
length	The length of data sent by the RTCU
arg	A user supplied 32 bit variable that was set when vsmsgwSetDataPackageCB() was called

Output:

rc	This is the return code that should be returned to the RTCU device, 0 if OK, else error
----	---

vsmgwnit()

Synopsis

VSMGW_API int _cdecl

vsmgwnit(int MyNodeID, const char *GWIP, int GWPort, const char GWKey[9],
cbfuncText SMSText, cbfuncPDU SMSPDU, void *arg);

Description

Initialize the connection to the RTCU Communication Hub

Input :

MyNodeID	The nodeid for the PC application. If set to 0, it will be assigned by the RTCU Communication Hub
GWIP	The IP address (or symbolic name) of the RTCU Communication Hub
GWPort	The portnumber the RTCU Communication Hub listens on
GWKey	The key value (an 8 character password) used to access the RTCU Communication Hub. Must be null terminated,
SMSText	A callback function that will be called whenever a Text SMS is received
SMSPDU	A callback function that will be called whenever a PDU SMS is received

Returns 0 if OK, else error (Look at the GWRC_xx codes)

vsmgwInitAdv()

Synopsis

```
VSMSGW_API int _cdecl
vsmgwInitAdv(int MyNodeID, const char *GWIP, int GWPort, const char GWKey[9],
unsigned char CryptKey[16]);
```

Description

Initialize the connection to the RTCU Communication Hub

Input :

MyNodeID	The nodeid for the PC application. If set to 0, it will be assigned by the RTCU Communication Hub
GWIP	The IP address (or symbolic name) of the RTCU Communication Hub
GWPort	The portnumber the RTCU Communication Hub listens on
GWKey	The key value (an 8 character password) used to access the RTCU Communication Hub. Must be null terminated,
CryptKey	The encryption key used to access the RTCU Communication Hub.

Returns 0 if OK, else error (Look at the GWRC_xx codes)

vsmgwSetSMSTextCB()

Synopsis

```
VSMSGW_API void _cdecl
vsmgwSetSMSTextCB(cbfuncText SMSText, void *arg);
```

Description

Set callback function that will be called whenever a Text SMS is received.

Input :

SMSText	A callback function that will be called whenever a Text SMS is received
arg	A user supplied 32 bit variable that will be supplied to the callback function when it gets called

vsmsgwSetSMSPDUCB()

Synopsis

VSMSGW_API void _cdecl
vsmsgwSetSMSPDUCB(cbfuncPDU SMSPDU, void *arg);

Description

Set callback function that will be called whenever a PDU SMS is received.

Input :

SMSPDU	A callback function that will be called whenever a PDU SMS is received
arg	A user supplied 32 bit variable that will be supplied to the callback function when it gets called

vsmsgwSetDataPackageCB()

Synopsis

VSMSGW_API void _cdecl
vsmsgwSetDataPackageCB(cbfuncPackage PACKAGE, void *arg);

Description

Set callback function that will be called whenever a data package is received.

Input :

PACKAGE	A callback function that will be called whenever a data package is received
arg	A user supplied 32 bit variable that will be supplied to the callback function when it gets called

vsmsgwIsConnected()

Synopsis

VSMSGW_API bool _cdecl vsmsgwIsConnected(void);

Description

Returns true if connected to the RTCU Communication Hub

vsmsgwGetMyNodeID()

Synopsis

VSMSGW_API int _cdecl **vsmsgwGetMyNodeID**(unsigned long* MyNodeID);

Description

This function will return this nodes nodeid. This function is used especially when a dynamic node-id is requested (by setting MyNodeID to 0 in the vsmsgwInit() function)

vsmsgwSendSMS()

Synopsis

VSMSGW_API int _cdecl vsmsgwSendSMS(unsigned int HisNodeID, const char str[161], int *rc);

Description

Send a Text SMS message to the specified NodeID, the return code from the RTCU device, will be contained in rc

Input :

HisNodeID	The node number (serial number) of the receiving RTCU device
str	The string to send to the receiving RTCU device. Maximum size is 160 characters + NULL terminator.

Output :

rc	The return code from the receiving RTCU device, 0 if OK, else error
----	---

Returns 0 if OK, else error (Look at GWRC_xx codes)

vsmgwSendPDU()

Synopsis

```
VSMGW_API int _cdecl vsmgwSendPDU(unsigned int HisNodeID,
const unsigned char data[140], int length, int *rc);
```

Description

Send a PDU SMS message to the specified NodeID, the return code from the RTCU device, will be contained in rc

Input :

DstNodeID	The node number (serial number) of the receiving RTCU device
data	The data to send to the receiving RTCU device Maximum size is 140 bytes.
length	The length of data to send

Output :

rc	The return code from the receiving RTCU device, 0 if OK, else error
----	---

Returns 0 if OK, else error (Look at the GWRC_xx codes)

vsmsgwSendDataPackage()

Synopsis

VSMMSGW_API int _cdecl vsmsgwSendDataPackage(unsigned int HisNodeID, const unsigned char data[4096], int length, int *rc);

Description

Send a data package to the specified NodeID, the return code from the RTCU device, will be contained in rc.

Input :

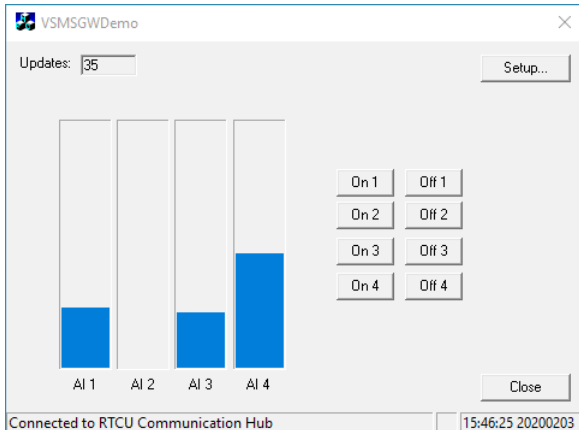
DstNodeID	The node number (serial number) of the receiving RTCU device
data	The data to send to the receiving RTCU device Maximum size is 4096 bytes on devices using large package support else 480 bytes.
length	The length of data to send

Output :

rc	The return code from the receiving RTCU device, 0 if OK, else error
----	---

Returns 0 if OK, else error (Look at the GWRC_xx codes)

Description of the demo project



The demo project consists of a PC application and a RTCU application (see Appendix A). The RTCU application sends periodically the values of its 4 analog inputs to the PC application, and the PC application can send commands to the RTCU device that will command one of the four outputs on the RTCU device to switch either on or off.

The PC application uses the VSMSGW Library for sending/receiving information from the RTCU device. When started, the application tries to connect to the RTCU Communication Hub, using information set in the “Setup” dialog. In the status bar of the program, you can see if the application is connected or not to the RTCU Communication Hub.