Rhio232

Serial I/O Manager

User Manual

Version 1.0.1

2005-08-08

Guide for the Rhio232

Version 1.0.1

Firmware version 1.3.2

Printed in Korea

Trademark

HelloDevice™ is a trademark of Sena Technologies, Inc.

Windows® is a registered trademark of Microsoft Corporation.

Ethernet® is a registered trademark of XEROX Corporation.

Notice to Users

When a system failure may cause serious consequences, protecting life and property against such consequences with a backup system or safety device is essential. The user agrees that protection against consequences resulting from system failure is the user's responsibility.

This device is not approved for life-support or medical systems.

Changes or modifications to this device not explicitly approved by Sena Technologies will void the user's authority to operate this device.

Technical Support

Sena Technologies, Inc.

210 Yangjae-dong, Seocho-gu

Seoul 137-130, Korea

Tel: (+82-2) 573-5422

Fax: (+82-2) 573-7710

E-Mail: support@sena.com

Website: http://www.sena.com

Revision History

Revision	Date	Name	Description
V1.0.0	2005-05-21	D.H. Shin	Initial Release
V1.0.1	2005-08-08	D.H. Shin	"*Table 2-2. Terminal Block Pin Assignment of the Rhio232* " is added.

Table of Contents

1. Introduction

1.1 Overview

The Rhio232 is a Serial I/O Manager that enables Sena Device Servers to control and monitor I/O devices. It is designed to connect to a Sena Device Server through the RS232 interface.

The Rhio232 supports 10 Digital Relay Output ports and has basic logic function capability such as AND, OR, NOT and Delay/Pulse along with the status of the input ports. The Rhio232 supports 12 optically isolated digital inputs for monitoring of the digital sensors. The Rhio232's Analog ports support both level mode for data acquisition and switch mode for threshold detection. The data communication between host computers is done by event-driven method which is triggered when the status is changed.

Users may probe, configure, test the Rhio232 system using Windows application, Rhio Manager and create their own Windows based application program by using MFC DLL library.

The Rhio232, which can be used directly or with device servers, is designed to meet the requirements of various applications such as remote data acquisition system, distributed I/O system, industrial automation control/monitoring/metering.

Please note that this manual assumes user knowledge of Internetworking protocols and serial communications.

1.2 Package Check List

- Rhio232 external box
- CAT5 cable
- RJ45 to DB9 Female cable connector
- DIN rail mount kit
- Quick Start Guide
- CD-ROM including the Rhio Manager and Rhio232 DLL and User Guide

1.3 Product Specification

Serial Interface	Supports RS232 serial port, RJ45 connector Baud rate: 9,600/Flow control: None/Data: 8 bit/Stop: 1 bit
Digital Input	-Number of channels: 12 -Input type: Voltage -Input circuitry: Optically isolated photo-coupler -Input range: 0V ～ ±24V OFF 0V ～ ±1.2V, ON ±3.3V ～ ±24V -Sampling rate: 20ms -Isolation voltage: 5KV
Digital Output	- Number of channels: 10 - Output type: Relay - Rated load: 3A/240VAC - Insulation resistance: 1000㏁ Min (DC500V) - Isolation voltage (coil and contact): 4KV - Reaction within 10ms
Analog Input	- Number of channels: 4 - Effective resolution: 10-bit - Input type: Voltage, Direct Coupling - Input range: 0V ～ Aref (Analog reference voltage, 2~5V) - Sampling rate: 1000 samples/sec
Protocol	ARP, IP/ICMP, TCP telnet, DHCP client, PPPoE
Management	- Rhio Manager Windows Utility, Serial Console or Telnet
Software Support	- Windows MFC DLL library - I/O configuration, I/O status monitoring/control
Diagnostic LED	- Power, Link, Act - Digital Output, 1~10 - Digital Input, 1~12 - Analog Input, 1~4
Power	9V～48VDC, Max. 5W
Environmental	- Operating temperature: 5℃ to 50℃ - Storage temperature: -40℃ to 66℃ - 90% Non-condensing

Physical properties	137 x 111 x 58 (mm), 5.4 x 4.4 x 2.3 (in.) Weight: 730g
Certification	FCC (A), CE, MIC
Warranty	5-year limited warranty

2. Getting Started

This chapter describes how to set up and configure the Rhio232 in the first place.

- *2.1 Panel Layout* explains the panel layout and LED indicators.
- *2.2 Connecting the Hardware* describes how to set up DIN rail mount kit and how to connect the power and the serial device to the Rhio232.

Following items are required to get started.

- DIN rail mount kit (included in the package).
- CAT5 cable for configuration or connecting device server (included in the package).
- RJ45 to DB9 Female connector for configuration or connecting device server (included in the package).

2.1 Panel Layout

The Rhio232 has LED indicator lamps for status display. The lamps in the left hand side indicate the system power-on status, Serial Rx and Serial Tx for RS232 communication status. There are 10 lamps for displaying digital output status, 12 lamps for digital input status, and lamps for 4 analog port status. Table 2-1 shows the description of the indicator lamps of the Rhio232.

Table 2-1. LED indicator lamps

Lamps		**Function**
Serial	**Rx**	Green LED blinks whenever there is any incoming data stream through the serial port of the Rhio232
	Tx	Green LED blinks whenever there is any outgoing data stream through the serial port of the Rhio232
Status	**Power**	Turned on to RED if power is supplied
Digital Input	**DI 1 ~ DI 10**	Turned on to GREEN if input status
Digital Output	**DO 1 ~ DO 12**	Turned on to GREEN if output status
Analog Input	**AI 1 ~ AI 4**	In Level Input mode, it is turned on to GREEN if the value is larger than 512. In Switch Input mode, it is turned on to GREEN if it is larger than threshold value.

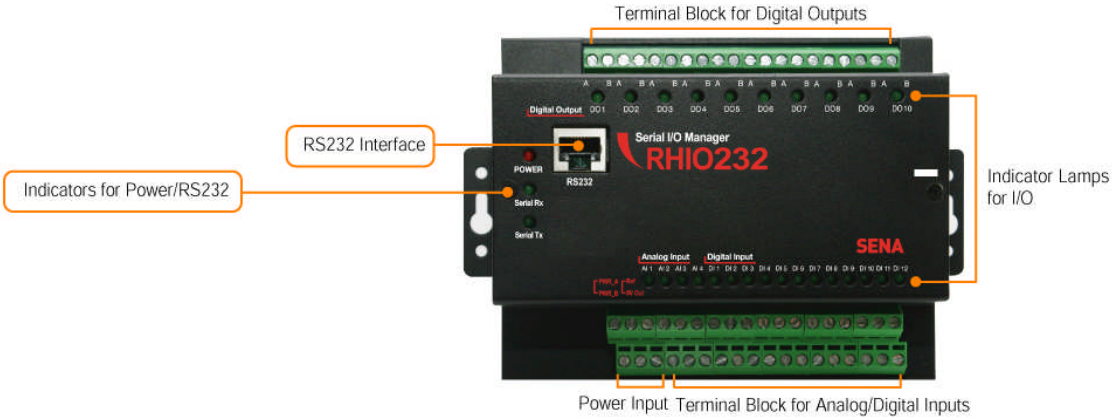


Figure 2-1. The panel layout of the Rhio232

Table 2-2. Terminal Block Pin Assignment of the Rhio232

Input Pin Assignment			AI1	AI2	AI3	AI4	DI1	DI2	DI3	DI4	DI5	DI6	DI7	DI8	DI9	DI10	DI11	DI12
	1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33	35
	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34	36

• Power Input (PWR_A , PWR_B): (1,2) – No polarity.

• Analog reference voltage (Ref) : (3)

- Rhio's analog reference voltage represents voltage from 0 V to a reference voltage in 1,024 steps. The reference voltage should not exceed 5 V. For details, please refer to the section *4.4 ADC Input Port Setting*.

• 5V Out : (4)

- 5v Out is an AVCC output pin. User can use output of this pin as a power input of user's device. One of pin no. 5,7,9 or 11 can be used as a power ground.

• Analog Input (AI1 ~ AI4) : (5,6),(7,8),(9,10),(11,12)

• Digital Input (DI1 ~ DI12) : (13,14),(15,16),(17,18),(19,20),(21,22),(23,24),
(25,26),(27,28),(29,30),(31,32),(33,34),(35,36)

Note : 1. Each Input is composed of one pair of upper and lower pins.

2. Except for the input pin no.3 and 4, users can connect the wire to the pins without considering the polarity.

Output Pin Assignment	DO1		DO2		DO3		DO4		DO5		DO6		DO7		DO8		DO9		DO10	
	A	B	A	B	A	B	A	B	A	B	A	B	A	B	A	B	A	B	A	B
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

	Each Output is composed of one pair of pins as follows. • Digital Output (DO1 ~ DO10) : (1,2), (3,4), (5,6), (7,8), (9,10), (11,12), (13,14), (15,16), (17,18), (19,20)

2.2 Connecting the Hardware

2.2.1 Setting up DIN Rail mount kit

Users may use DIN rail mounting kit included in the package to install the Rhio232 on to the DIN rail.

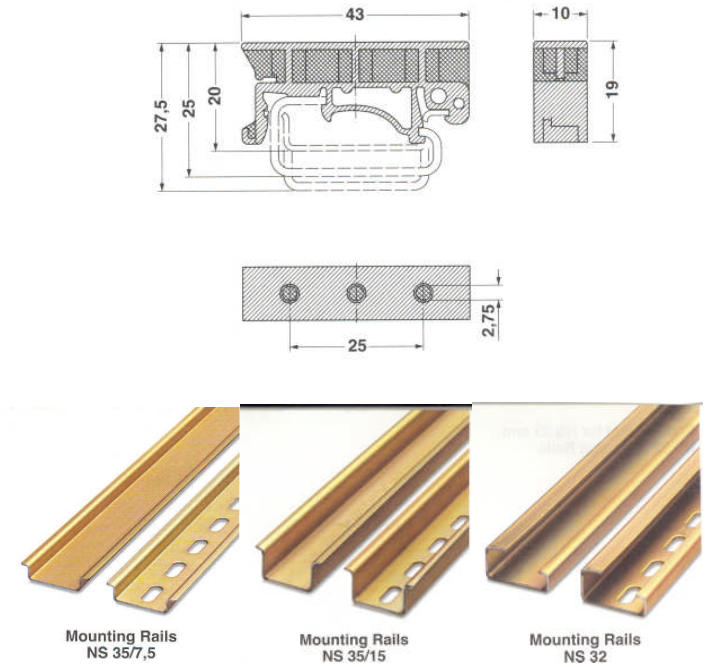


Figure 2-2. Dimension of DIN Rail mount kit and applicable DIN Rails

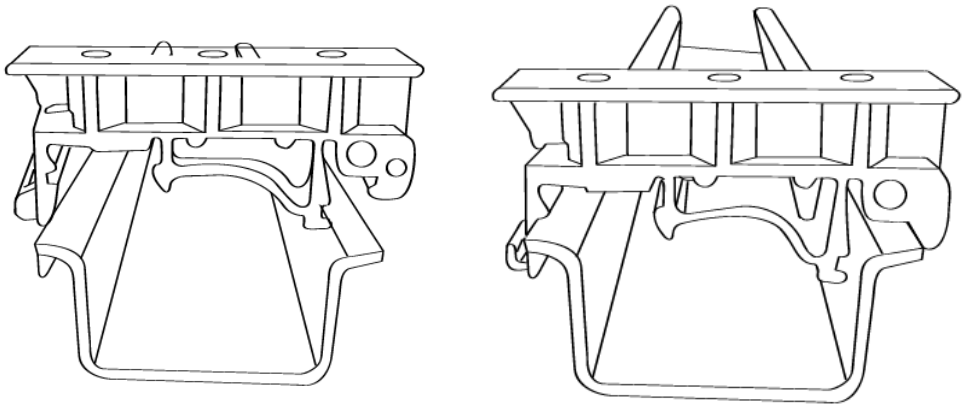


Figure 2-3. Installing DIN Rail mount kit into DIN Rail

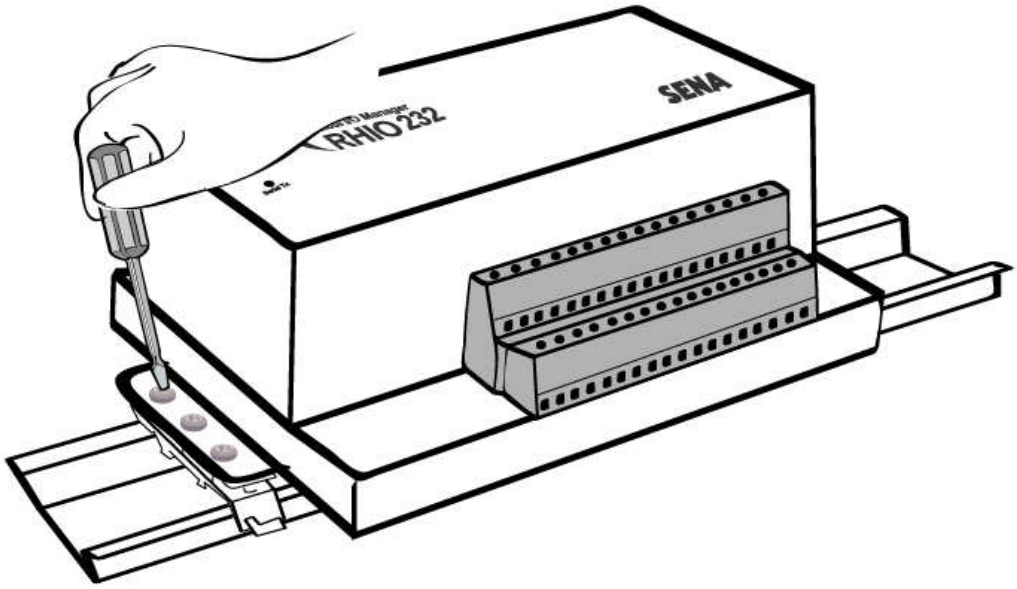


Figure 2-4. Setting up the Rhio232 to DIN Rail

2.2.2 Connecting the Power

Supply the proper power according to the power specification of the Rhio232, i.e. 9V~48VDC, MAX. 5W. If the power is properly applied, **[Power]** indicator will maintain RED. Be sure not to use the cable longer than 3m for normal operation.

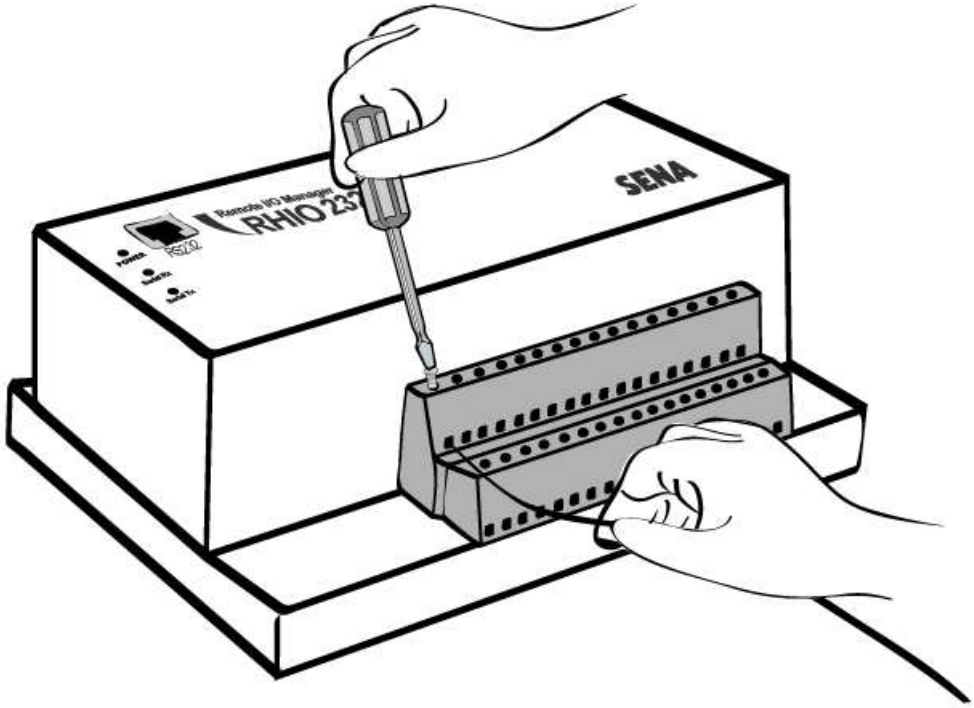


Figure 2-5. Connecting the power to the Rhio232

2.2.3 Connecting to the serial device

Connect the one end of the CAT5 cable to RS232 port of the Rhio232 and the other to host or device server. If connector type of host or device server is DB9, connect the other end using RJ45 to DB9 Female adaptor. If the cable is properly hooked up, the Rhio232 will have a valid connection to the device server(or host) by indicating:

- **[Serial Rx]**, **[Serial Tx]** green lamps continuously blink to indicate the incoming/outgoing data stream through serial port of the Rhio232

If any of the above does not happen, the Rhio232 is not properly connected to the RS232 communication.

Note: User must configure serial parameters a host or device server in such a way that it should be same with the Rhio232's serial parameters. Serial parameters of the Rhio232 are as follows: 9600 Baud rate, Data bits 8, Parity None, Stop bits 1, No flow control

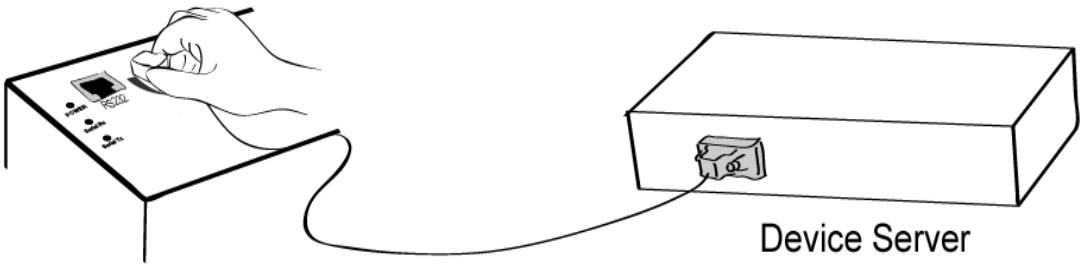


Figure 2-6. Connecting host or device server to the Rhio232

2.3 Rhio Manager Installation

Rhio Manager is a Windows Utility program for system configuration and I/O test of the Rhio232.

2.3.1 Rhio Manager Installation

Users may install the Rhio Manager software within the CD-ROM or by downloading at Sena web site `www.sena.com/support/downloads`. If it is installed normally, then it will be placed on to the menu of `[Start]->[Program]->[SENA]->[RHIO Manager]`. The screen layout is shown in Figure 2-7.

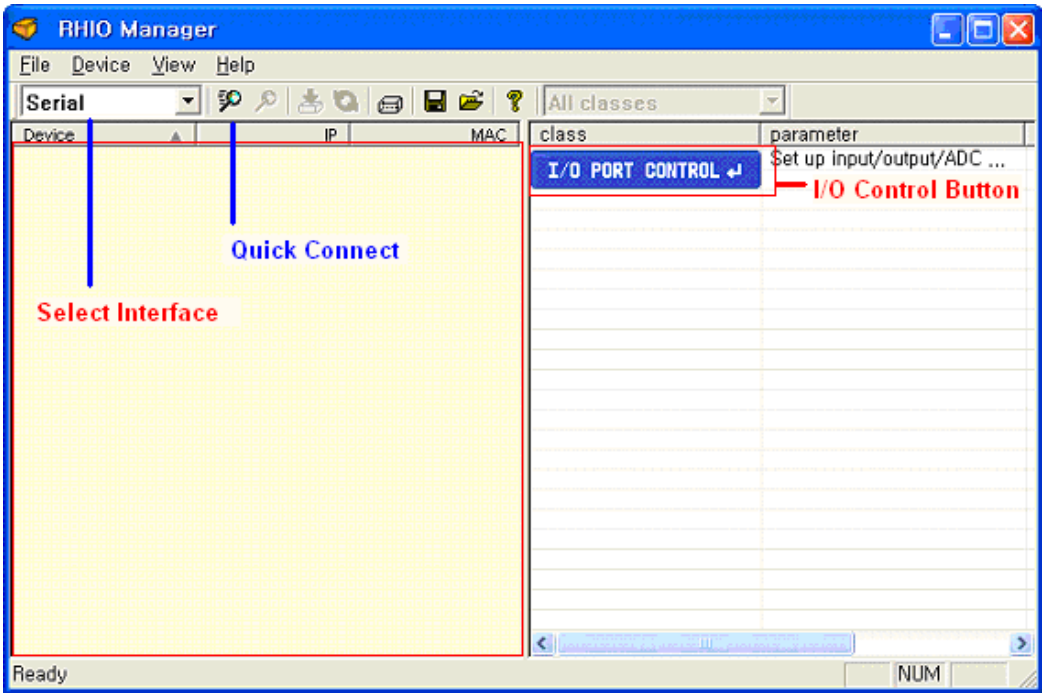


Figure 2.7 Rhio Manager screen layout

2.3.2 Basic configuration using Rhio Manager

Select Interface

It specifies the communication method between the Rhio232 and Rhio Manager. Be sure to set it up as “Serial” mode.

Quick Connect

By using [Quick Connect], user can select the RS232 serial port that is connected to the Rhio232.

Figure 2.8 Quick connect dialog box

I/O PORT CONTROL

Users may monitor, control and configure the Rhio232’s I/O ports by clicking [I/O PORT CONTROL] button.

2.4 Restoring Factory Default

Users may restore the Rhio232 parameters into factory default value by pressing factory reset switch on the hole of the Rhio232 side panel. They will have to put the sharp pin into the hole and press it for around 1 sec to reset the Rhio232. The Rhio232 will be rebooted after the operation.

The following is the factory default value of the parameters.

I/O Port status: Enable
ADC Operation mode : Level Mode
Power-out Post Recovery: Enable
ADC Threshold value : 512
Output Port operation condition : None
Run/Stop status : Run

3. I/O Setting and Application

3.1 I/O Monitoring and Control

You can monitor, control and set I/O states by pressing the [I/O PORT CONTROL] button.

Once [I/O PORT CONTROL] is invoked, Rhio Manager begins to monitor the I/O state by connecting to a serial port of the Rhio232 via a RS232 serial interface.

3.1.1 LED

- ON : Red Icon
- OFF : Blue Icon
- Disable : Grey Icon
- Condition ON/OFF : Green Icon
- Macro: M
- Delay ON: Red D
- Delay OFF: Blue D
- Pulse: P
- Level Mode ADC Port : Green

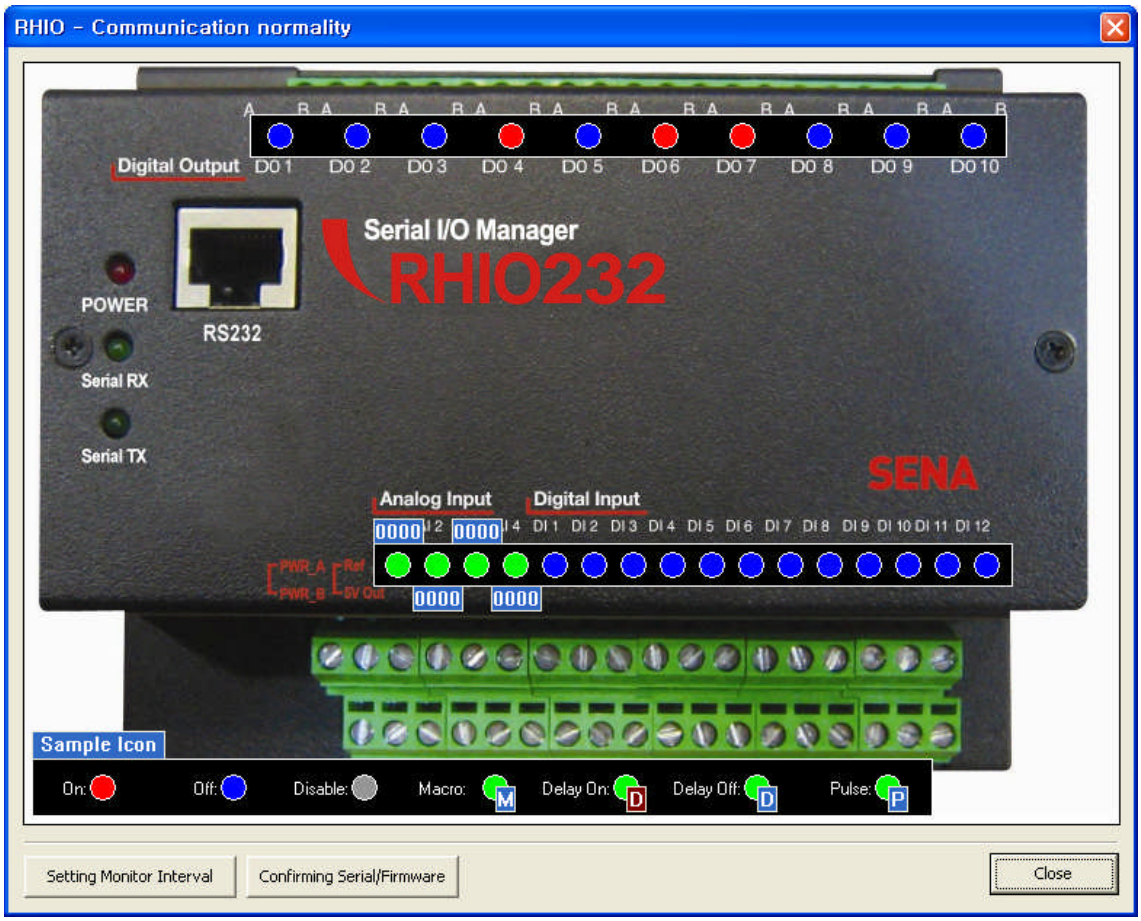


Figure 3-1 The I/O port Control screen

3.1.2 Specifying monitoring interval

You can continue to monitor the I/O state of the Rhio232 at a specified time interval by setting [Setting Monitor Interval]. The valid value for monitoring intervals is any number between 2 and 10 seconds. In the specified time interval, Rhio Manager sends a state request command and receives a response from the Rhio232 and displays it on the screen.

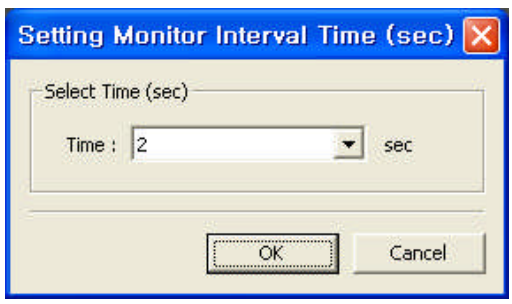


Figure 3-2 Setting Monitoring Interval

3.1.3 Monitoring I/O port

Upon receiving a state request command from Rhio Manager, the Rhio232 returns information on the overall states of the digital input, analog input and digital output ports.

- Digital Input Port shows the ON/OFF state of input.
- In Level mode, ADC Input Port converts the analog value retrieved to a digital value in 1,024 steps and displays the converted value (“0000”-“1023”).
- In Switch mode, ADC Input Port compares the input value to a specified threshold value and displays ON if it is higher and OFF if not.
- Digital Output Port displays the ON/OFF state, operation condition for an output port, and standby state.

3.1.4 Controlling digital output port

You can place your mouse over the Digital Output Port LED of Rhio Manager and left-click it to control ON/OFF state.

- When the operation condition for a digital output port is not specified, the ON/OFF state for the port is toggled each time you left-click your mouse.
- When the operation condition is specified, the port is set to ON if it is met, and it is set to OFF and displayed as a standby state if not.
- After output control is completed, the Rhio232 returns the states of all ports to the host computer.

3.2 Digital Input Setting

The Rhio232 has 12 digital input ports. You can enable/disable each of these digital input ports with Rhio Manager or by issuing the commands specified in "*Ch. 4 I/O Port Related Protocols*". When setting the digital input with Rhio Manager, place your mouse cursor over the Digital Input LED on the I/O Port Control screen and right-click it to display the Setting window.

3.2.1 Setting Enable/Disable

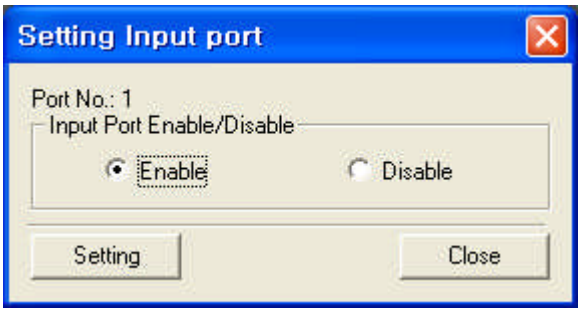


Figure 3-3 Setting digital input

Digital input setting has two options: Enable and Disable. After selecting either of the options, press the [Setting] button to apply it in the system. A port cannot be set while it is operating in Run mode. Therefore, Rhio Manager sends a command that switches its mode to Setting mode first and then issues the set command when its operation is stopped.

When set to Enable, Rhio Manger receives the ON/OFF state from the device connected to a digital input port and then displays it. When set to Disable, it displays Disable regardless of the ON/OFF state of the device connected to the input port.

3.3 Digital Output Port Setting

The Rhio232 has 10 digital output ports. You can set each of these digital output ports with Rhio Manager or by issuing the commands specified in "*Ch. 4 I/O Port Related Protocols*". Place your mouse cursor over the Digital Output Port LED on the I/O Port Control screen and right-click it to display the Setting Output Port window.

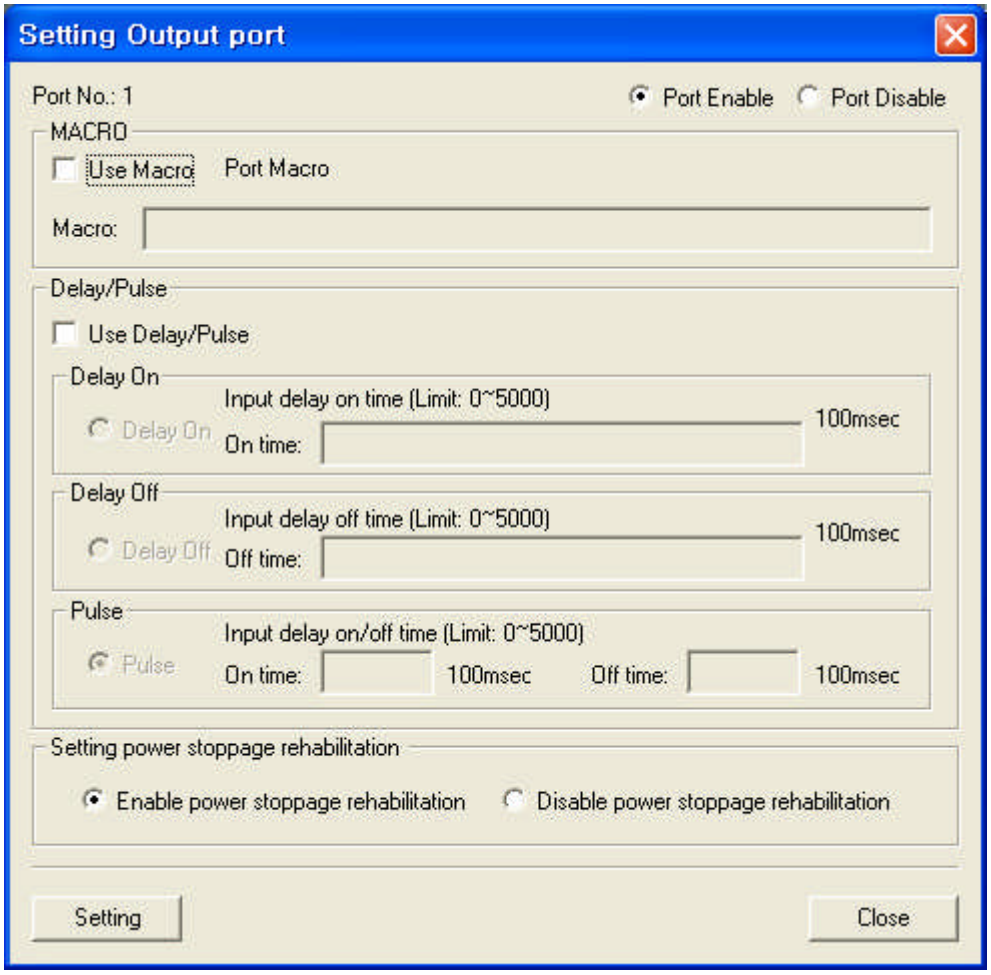


Figure 3-4 Setting digital output port window

3.3.1 Setting Enable/Disable

Set a specified port to Enable/Disable by selecting the [Port Enable] and [Port Disable] check boxes. If it is set to Disable, the digital output port becomes fixed to OFF.

3.3.2 Setting run condition

When the [Macro] check box is selected, you can enter a conditional expression and then use it to control operation of an output port. Enter the desired conditional expression in the [Macro] box.

- Specify an operation condition for each output port.
- An output port that has not been set is regarded as a port available for direct control.
- The final value obtained from a logical operation on the listed expression becomes the state of an output port.
- If the operation condition expression is cleared, a port becomes available for direct access.
- Operation condition expressions can be specified as follows:
    1) Port No. + Logic Expression (&,|) + Port No.
    2) Logic Expression (!) + Port No.
    3) Port No. + Logic Expression (&,|) + Logic Expression (!) + Port No.
    (Logical operator "!" can only be effective before the relevant port no.)

    예) - Input #1 AND Input #2
    I1&I2
    - Input #1 OR Output #2
    I1|O2
    - The Inverse of Input #1
    !I1
    - Output #2 AND Output #3 AND the inverse of Output #4
    O2&O3&!O4

- A single logical expression can contain up to 21 ports.
- If a port is directly set to ON when its operation condition is not met, it goes into standby state.
- A port goes into the ON state if its operation condition is met and into OFF and standby states if not.
- If a port is directly set to OFF, it does not operate even if the operation condition is satisfied.

3.3.3 Delay & Pulse Operation

If the [**Use Delay/Pulse**] check box is selected, the output port executes Delay and Pulse operation. The setting value for Delay and Pulse can be entered in 100 ms.

- When Delay ON is selected, you can send the *Output Port ON* command to set an output port to ON after a specified delay time has passed.
- When Delay OFF is selected, you can send the *Output Port OFF* command to set an output port to OFF after a specified delay time has passed.
- When Pulse is selected, the port continues to toggle between ON and OFF according to the specified ON/OFF time.

3.3.4 Setting Power-out Post Recovery

Power-out Post Recovery can be set for an output port using the [Setting power stoppage rehabitation] pane in the Setting window.

- If it is enabled, the Rhio232 retains its output port states prior to power-out when power goes out and back on
- If Power-out Post Recovery is enabled for an output port that has been set with an operation condition, the port becomes ON when power is restored if the operation condition is satisfied.
- If it is disabled, the output port state becomes OFF when power goes out and back on.

3.4 ADC Input Port Setting

The Rhio232 has 4 ADC input ports. You can set an ADC input port with Rhio Manager or by issuing commands via an I/O port protocol (refer to "*Ch.4 I/O Port Related Protocols*").

When using Rhio Manager, place the mouse cursor over the ADC Port LED on the [I/O Port Control] screen and right-click it to display the Setting ADC port windows as shown below:

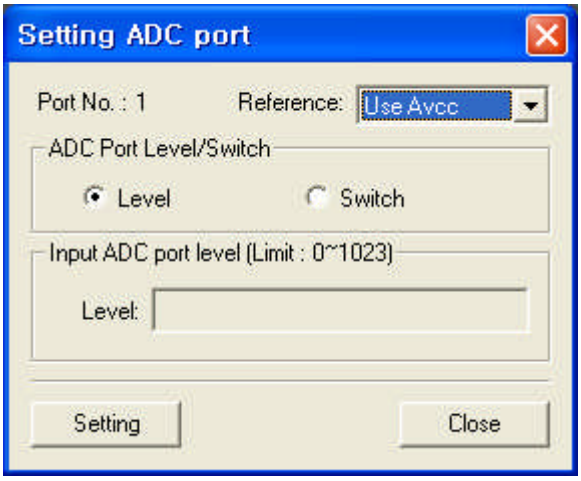


Figure 3-5 Setting ADC Input Port

3.4.1 Setting Reference

Specify a reference voltage for analog input data. Rhio represents voltage from 0 V to a reference voltage in 1,024 steps. A reference voltage may not exceed 5 V. Specify a reference voltage in the Reference list box.

- **Use Avcc**: Specify Avcc (5V) as a reference voltage.
- **Use inside**: Specify the internal reference voltage (2.56V) as a reference voltage.
- **Use Outside** : Specify voltage issued to Aref as a reference voltage.

3.4.2 Setting ADC Input Port

Analog Input Port has two modes: Level mode and Switch mode.

- **Level mode**: Display voltage from 0 V to a reference voltage in 1,024 steps (“0000” ~ “1023”).
- **Switch mode**: Compare input voltage level to threshold level setting and send a state change response when the input level is higher or lower than the threshold. An ADC input recognizes it is changed only when it is changed larger than 8 steps from the pre-configured threshold.

3.5 I/O Port Connection

This section describes how to make a required connection with the digital output, digital input and ADC input port of the Rhio232 for the users' devices. The length of cables used for I/O port connection should be less than 3 meters to ensure normal operation.

3.5.1 Digital Output Port

All digital output ports are equipped with a status LED, which illuminates when a relay point is set to ON. An electric load can be connected as shown below by using OUTA1 and OUTB1 as driving switches.

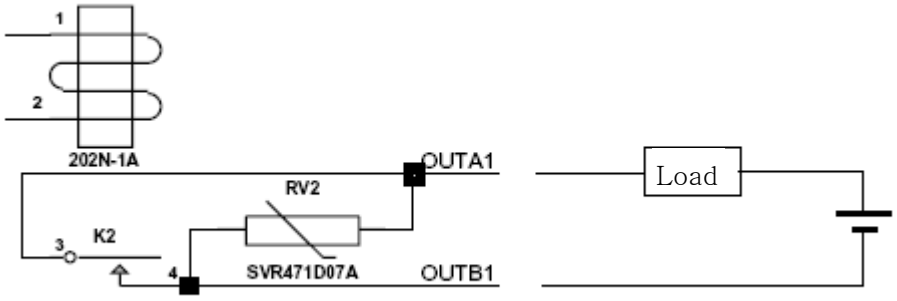


Figure 3-6 Connection of Digital Output Port

3.5.2 Digital Input Port

A digital input system operates regardless of the ± polarity of the voltage and is insulated from the internal circuits in the system. It can be configured as shown in Figure 3-7 and has a status LED for each input, which illuminates when input voltage is issued.

Note)

Whilst it may operate in a voltage other than that specified (ON > ±3.2V / OFF < ±1.3V), be sure to use the specified voltage to ensure the stable operation.

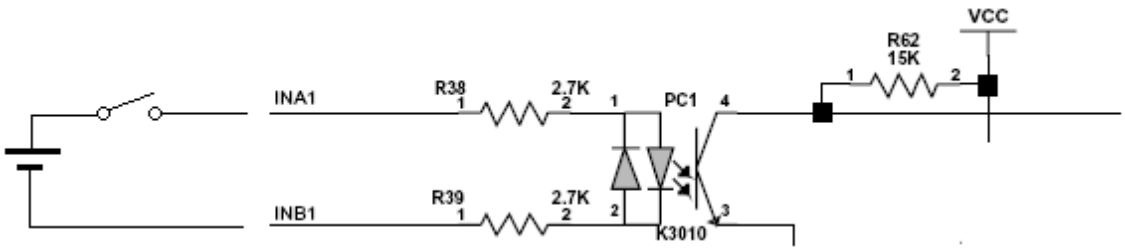


Figure 3-7 Connection of Digital Input Port

3.5.3 ADC Input Port

An ADC input port is a non-insulated input port. If possible, a circuit should be made using

AVCC (+5V) voltage supplied from internal circuits. When operating in Switch mode, a threshold voltage should be specified. Input is set to ON and the ADC status LED is ON if the input voltage is higher than the specified threshold value. Conversely, input is set to OFF and the LED turns off when the input voltage is lower than the threshold. The ADC status LED operates only when it is set as Switch mode.

1) Connecting reference voltage (AREF) in analog input

AREF can be set as either internal 2.56 V, internal AVCC or external AREF point. Internal input can be set using command. The external input can be set by splitting the AVCC voltage into R1 and R2 as shown below. The ideal resistance of split resistors R1 and R2 should be within the range of 1㏀ - 5㏀.

Note) AREF voltage cannot be set to the value less than 2 V.

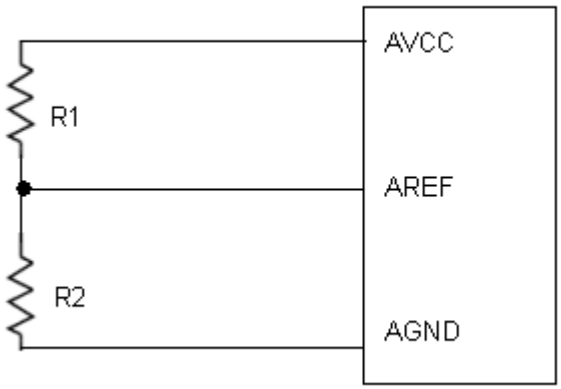


Figure 3-8 Connecting Aref

2) When using an Analog Input Potentiometer

An analog input operates in reference to input voltage and has an impedance of 100㏀. When using a Potentiometer as shown in Figure 3-9, an impedance in the range of 1㏀ - 5㏀ is ideal. When using an external signal source, lower impedance ensures stable operation against various noises.

Note: Make sure that analog Input voltage does not exceed AREF voltage.

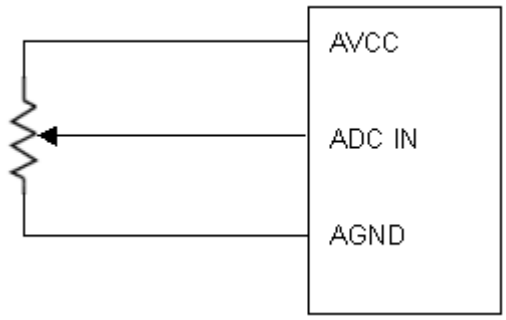


Figure 3-9 Circuit connection when a Potentiometer is used

3) Connecting when a voltage higher than AVCC voltage is used

For voltage input, the circuit should be split as shown in Figure 3-10. In case that the input wire is long or there is a strong noise nearby, it is recommended to have additional clamp diode in order to minimize the effect to other ADC channels although there is an internal clamp diode available.

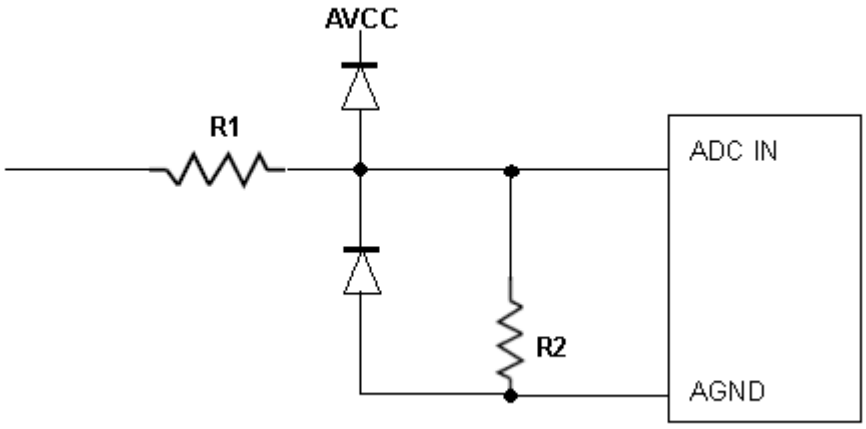


Figure 3-10 Circuit connection when a voltage higher than Avcc is used

4. Software Development & Application

You can use the Rhio library to develop application software that is used to communicate with the Rhio232.

- The Rhio Library Files

`RHIO_Proc.dll, RHIO_Process.h`

You must link these two files in order to develop software using the Rhio library.

- Test Program

A sample test program that has been developed using the Rhio library is provided to you in the form of source and setup files (`RHIO_TEST_Setup.exe`). The test program shows developers how to utilize the Rhio library more easily.

4.1 The Rhio Library

4.1.1 Overview of the Rhio Library

The Rhio library is an MFC library that allows you to implement a communication protocol between Rhio and PC in a Windows environment. Since the Rhio library contains CSocket Class, it should be linked to a Microsoft Winsock component during program development. The `RHIO_CommProcessCreate` function must also be used to create `Process` Class for use of the library.

4.1.2 Reference

For definition of enumeration(s), structure(s) and function(s), refer to RHIO_Process.h.

1) Enumeration (See Appendix C.1)

Type	Description
`EOnOffFlag`	ON/OFF - a flag indicating run state
`SendStateFlag`	A flag indicating transmission state of a command sent to Rhio
`ESetOutputFlag`	A flag related to enable/disable state when setting output port macro/delay/pulse
`EADCMode`	A flag specifying whether ADC is in Level mode or in Switch mode

2) Structure (See Appendix C.2)

Structure	Description
SADCData	ON/OFF state of each port
SOnOffStateData	ON/OFF state of all ports
SSetOutput	Output port configuration information
SSetADC	ADC configuration information
SSetInput	Input port configuration information
SRHIOSetting	All ports configuration information

3) Function (See Appendix C.3)

Function	Description
RHIO_CommProcessCreate	A function that creates Process Class that must be created for use of the library.
RHIO_SockConnect	Connect to RHIO via a socket (TCP/IP).
RHIO_CommConnect	Connect to RHIO via a serial port.
RHIO_Close	Disconnect from RHIO.
RHIO_SndCmd_SetOnOff	Send a command that controls ON/OFF.
RHIO_SndCmd_GetOnOff	Send a command that checks ON/OFF setting.
RHIO_SndCmd_SetSettingMode	Send a command that sets setting mode.
RHIO_SndCmd_SetRunMode	Send a command that sets run mode.
RHIO_SndCmd_SetMACRO	Send a command that sets macro for input port.
RHIO_SndCmd_GetMACRO	Send a command that checks macro setting for input port.
RHIO_SndCmd_SetDelayPulse	Send a command that sets delay/pulse for input port.
RHIO_SndCmd_GetDelayPulse	Send a command that checks delay/pulse setting for input port.
RHIO_SndCmd_SetADC	Send a command that sets the level of all ADC ports (1-4).
RHIO_SndCmd_GetADC	Send a command that checks the level of all ADC ports (1-4).
RHIO_SndCmd_SetPortEnable	Send a command that enables/disables all ports.
RHIO_SndCmd_GetPortEnable	Send a command that checks enable/disable

	state of all ports.
RHIO_SndCmd_SetPwrStopEnable	Send a command that enables/disables power-out recovery for all ports.
RHIO_SndCmd_GetPwrStopEnable	Send a command that checks enable/disable state of power-out recovery for all input ports.
RHIO_SndCmd_SetFactoryReset	Send Rhio Factory Reset command.
RHIO_SndCmd_SetSerial	Send a command that sets Rhio serial number.
RHIO_SndCmd_GetSerial	Send a command that checks Rhio serial number.
RHIO_SndCmd_GetFirmware	Send a command that checks Rhio Firmware version.
RHIO_GetSettingData	Retrieve corresponding data when an event occurs such that a response on a check for setting is received from each port.
RHIO_GetOnOffData	Retrieve corresponding data when an event occurs such that responses from a change in and control of ON/OFF setting are received.

4.2 Creating and demonstrating a sample program with Rhio library

The sample program (RHIO_TEST) is a dialog box based application that has been created with Microsoft Visual Studio .NET linked to **Rhio** library (`RHIO_Proc.dll, RHIO_Process.h`).

4.2.1 Program UI Configuration and their related classes

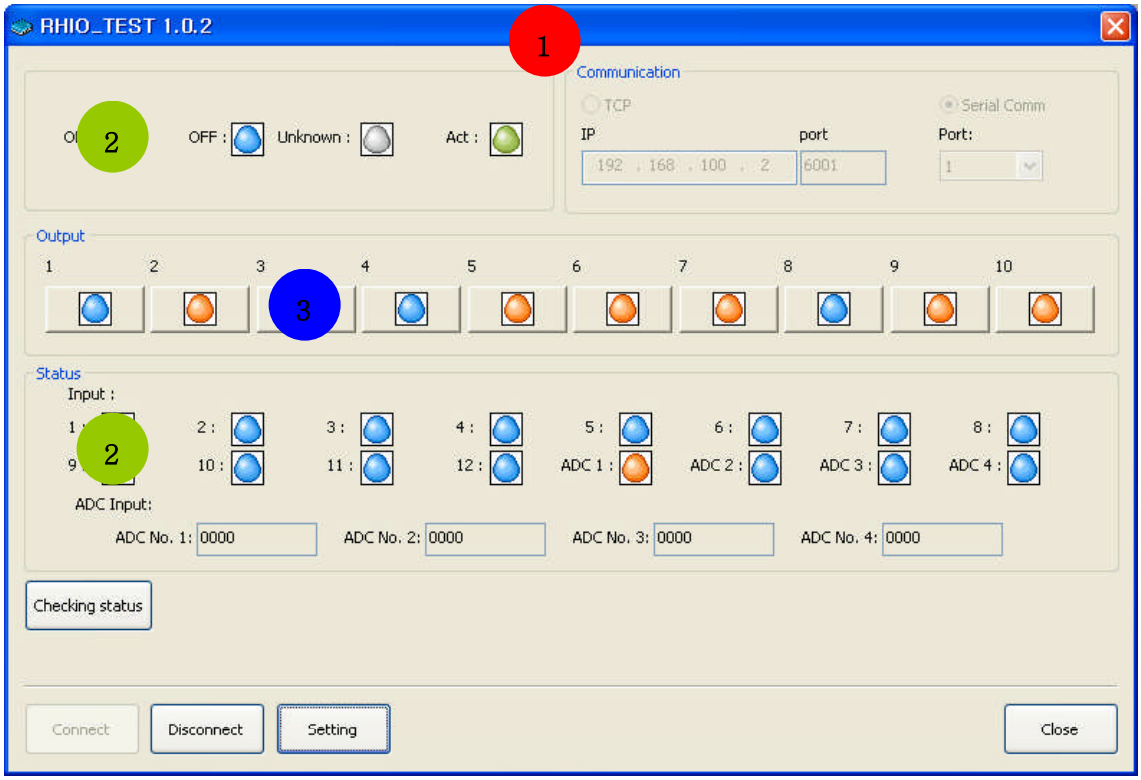


Figure 4-1 Main Window of a Sample Program

No.	Class	Related Files
1	`CWEB_IO_TESTDlg`	`WEB_IO_TESTDlg.h, WEB_IO_TESTDlg.cpp`
2	`CStateWnd`	`StateWnd.h, StateWnd.cpp`
3	`COutputButton`	`OutputButton.h, OutputButton.cpp`

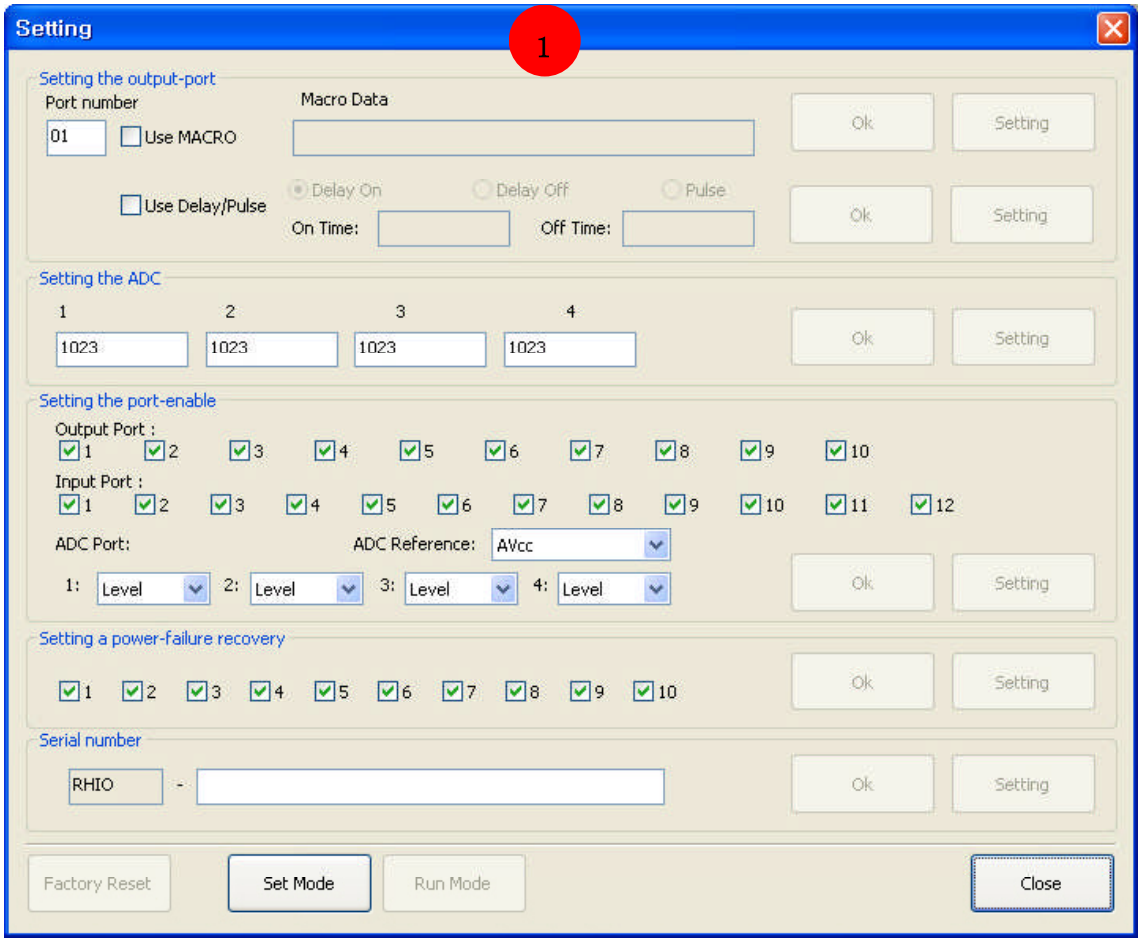


Figure 4-2 Setting Window of a Sample Program

No.	Class	Related Files
1	CSettingDlg	SettingDlg.h SettingDlg.cpp

4.2.2 Processing

4.2.2.1 Initializing Main window

1) Link the library and get the address of a required library function.

```
CWEB_IO_TESTDlg::RHIODllLoad()
```

2) Initialize dialog items in Main window.

```
CWEB_IO_TESTDlg::InitOutPutButton();
CWEB_IO_TESTDlg::InitInput(CPoint pntStart, int iWidth,
                           int iHeight, int iTerm)
CWEB_IO_TESTDlg::InitSample(CPoint pntStart, int iWidth,
```

```
                                int iHeight, int iTerm)
CWEB_IO_TESTDlg::InitSelComm();
```

3) Create Process Class.

```
m_rhCreate(CWnd *pParentWnd)
```

4.2.2.2 Event Handling procedure

1) When an event occurs in the main window of the program:

① The Connect button is clicked on.

```
CWEB_IO_TESTDlg::OnBnClickedButtonConnect()
```

- ✓ TCP Connection

```
m_rhSockConnect (BYTE bAddr1, BYTE bAddr2, BYTE bAddr3,
                 BYTE bAddr4, int iPort)
```

- ✓ Serial Connection

```
m_rhCommConnect (int iPort)
```

② The Disconnect button is clicked on.

```
CWEB_IO_TESTDlg::OnBnClickedButtonClose()
```

③ The Set Button is clicked on.

```
CWEB_IO_TESTDlg::OnBnClickedButtonSetting()
```

④ The State View button is clicked on.

```
CWEB_IO_TESTDlg::OnBnClickedButtonStateView()
```

⑤ The Output Port button is clicked on.

```
COutputButton::OnBnClicked()
```

2) When an event occurs in the Setting window of the program:

① The Factory Reset button is clicked on.

```
CSettingDlg::OnBnClickedButtonFactoryReset()
```

② The Set Mode button is clicked on.

```
OnBnClickedButtonSetmode()
```

③ The Run Mode button is clicked on.
`CSettingDlg::OnBnClickedButtonRunMode()`

④ The Monitor Serial button is clicked on.
`OnBnClickedButtonMonitorSerial()`

⑤ The Set Serial button is clicked on.
`CSettingDlg::OnBnClickedButtonSetSerial()`

⑥ The Monitor Power Stop button is clicked on.
`CSettingDlg::OnBnClickedButtonMonitorPwrStop()`

⑦ The Set Power Stop button is clicked on.
`CSettingDlg::OnBnClickedButtonSetPwrStop()`

⑧ The Monitor Enable State of the Port button is clicked on.
`CSettingDlg::OnBnClickedButtonMonitorEnable()`

⑨ The Enable Port button is clicked on.
`CSettingDlg::OnBnClickedButtonSetEnable()`

⑩ The Monitor ADC Input button is clicked on.
`CSettingDlg::OnBnClickedButtonMonitorInput()`

⑪ The Set ADC Input button is clicked on.
`CSettingDlg::OnBnClickedButtonSetInput()`

⑫ The Check Delay/Pulse State button is clicked on.
`CSettingDlg::OnBnClickedButtonMonitorOutput2()`

⑬ The Set Delay/Pulse button is clicked on.
`CSettingDlg::OnBnClickedButtonSetOutput2()`

⑭ The Check Macro Setting button is clicked on.
`CSettingDlg::OnBnClickedButtonMonitorOutput()`

⑮ The Set Macro button is clicked on.
`CSettingDlg::OnBnClickedButtonSetOutput()`

3) When an event occurs in a Rhio device:

```
CWEB_IO_TESTDlg::OnUpdateState(WPARAM wParam,
                                        LPARAM lParam)
```

① A port ON/OFF event occurs.
`m_rhGetOnOffData (SOnOffStateData &sOnOffData)`

② Events other than a port ON/OFF occur.
`CSettingDlg::OnReceive(WPARAM wParam, LPARAM lParam)`

4.3 Rhio Communication Protocol

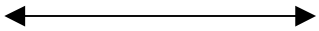
In this chapter, it covers how to send/receive the command/reply to/from Rhio device by using the Rhio communication protocol.

4.3.1 Overview

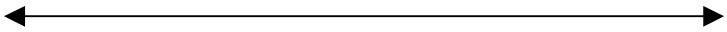
You can use the Rhio communication protocol to set, control and monitor RHIO.

4.3.1.1 Command Block

	START FLAG	**LENGTH**	**FUNCTION**	**DATA**	**LRC (BCC)**	**END FLAG**
Byte Size	1	2	2	N	2	2
	Start of the Command Block, **0x3A (":")**	The length from **FUNCTION** field to **DATA** field	Command Response	Data	XOR from **LENGTH** field to **DATA** field	CR (0x0D) LF (0x0A)



Length calculation



LRC calculation

- START FLAG

  Start of the command block

  0x3A (":")

- LENGTH

  The length of the FUNCTION and DATA Fields

- FUCTION

  Control/Set/Check/Status Command and Response Code

- DATA

  Control/Set/Check/Status data

- LRC (BCC)

It checks the Error of the command block.

The value by 1 byte XOR from **LENGTH** field to **DATA** field

- END FLAG

CR+LF (0x0D+0x0A)

-. The data of LENGTH and LRC is expressed as follows.

Each 4-bit nibble (upper 4-bit nibble and lower 4-bit nibble) is expressed as 1 Byte data. The 1-byte conversion of the 4-bit data is as follows.

0x0 ~ 0x9 → 0x30 ~ 0x39, 0xA ~ 0xF → 0x41 ~ 0x46

If the Rhio receives the data converted, then it converts it to original by inverse.

-. The timeout from the start of the frame to the time when it receives **LF** is 5 sec.

-. Rhio will discard the command if there is any error in the command received (BCC Error or Time Out) and will wait for the next command (Users have to write a code so that it will have to wait at least 5 sec if there is no response from Rhio after sending the command.)

4.3.1.2 NAK Response

- NAK Response condition

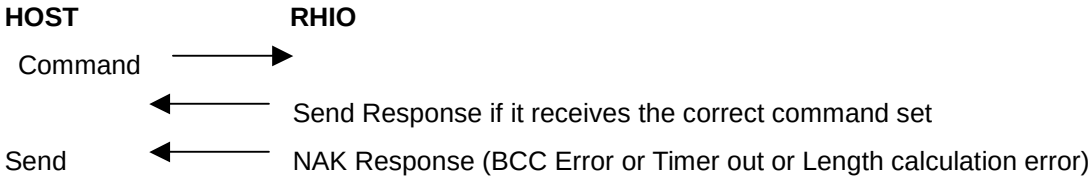
-. When there is Data BCC Error when receiving the command

-. When the command data is not completed within 1 sec after it is started (Time out)

-. When the length of the Data frame is not same as the one in the command (Frame Length error)

- NAK Response

	START FLAG	**LENGTH**	**FUNCTION**	**DATA**	**LRC (BCC)**	**END FLAG**
Byte Size	1	2	2	3	2	2
	Start of the Command Block, **0x3A (":")**	The length from **FUNCTION** field to **DATA** field	Response "00" 0x30,0x30	NAK Data "NAK" 0x4E, 0x41, 0x4B	XOR from **LENGTH** field to **DATA** field	CR (0x0D) LF (0x0A)



-. Users have to write code to send the command again or display NAK status if the program receives NAK Response.

* **NOTE**: In the following descriptions, each port is indicated as below:

- Input Port: I1 - I12
- ADC Input: A1 - A4 (Level Input and Switch Input modes)
- Output Port: O1 - O10

4.3.2 ON/OFF Control

4.3.2.1 ON/OFF Control Command

	Command	Data	Remark
Byte Size	2		
	"01" (0x30, 0x31) ON/OFF control	10-point output masking and ON/OFF control data	Send output points to set ON/OFF in a batch.

- Data

	MASK Data	Data Separator	ON/OFF Data
Data Order	1 - 10	11	12-21
Port No.	1 - 10	-	1-10
Data Content	Control: 0x31 Non-control: 0x30	0x2C (",") separates MASK from ON/OFF.	ON: 0x31 OFF: 0x30

MASK and ON/OFF Data correspond to one port per byte for each port in sequence.

- Port location by Data Order

Data	1	2	3	4	5	6	7	8	9	10
Order	12	13	14	15	16	17	18	19	20	21
Port	O1	O2	O3	O4	O5	O6	O7	O8	O9	O10

4.3.2.2 Response for ON/OFF Control

	Response	Data	Remark
Byte Size	2	52	
	"02" (0x30, 0x32) input/output overall state	ON/OFF state data for ADC Level 4, Input 12 Point and Output 10 Point.	Send state of input/output points in a batch.

- Data

Field	**Data** Order	**Port**	Data Content	
Control State	1	-	0x30: Normal Control, 0x31: No Run Mode, 0x39: Abnormal Control	
ADC Input LEVEL	2-6	A1	0x30: OFF, 0x31: ON (Switch Input), 0x39: Level Mode	"0000"-"1023": Level
	7	-	Field Separator 0x2C (",")	
	8-12	A2	0x30: OFF, 0x31: ON (Switch Input), 0x39: Level Mode	"0000"-"1023": Level
	13	-	Field Separator 0x2C (",")	
	14-18	A3	0x30: OFF, 0x31: ON (Switch Input), 0x39: Level Mode	"0000"-"1023": Level
	19	-	Field Separator 0x2C (",")	
	20-24	A4	0x30: OFF, 0x31: ON (Switch Input), 0x39: Level Mode	"0000"-"1023": Level

	25	-	Field Separator 0x2C (“,”)
Input State	26-29	I1-I4	0x30: OFF, 0x31: ON
	30	-	Field Separator 0x2C (“,”)
	31-34	I5-I8	0x30: OFF, 0x31: ON
	35	-	Field Separator 0x2C (“,”)
	36-39	I9-I12	0x30: OFF, 0x31: ON
	40	-	Field Separator 0x2C (“,”)
Output State	41-44	O1-O4	0x30: OFF, 0x31: ON, 0x32: Wait for a conditional execution, 0x33: Wait for Delay ON, 0x34: Wait for Delay OFF, 0x35: Run PULSE
	45	-	Field Separator 0x2C (“,”)
	46-49	O5-O8	0x30: OFF, 0x31: ON, 0x32: Wait for a conditional execution, 0x33: Wait for Delay ON, 0x34: Wait for Delay OFF, 0x35: Run PULSE
	50	-	Field Separator 0x2C (“,”)
	51,52	O9, O10	0x30: OFF, 0x31: ON, 0x32: Wait for a conditional execution 0x33: Wait for Delay ON, 0x34: Wait for Delay OFF, 0x35: Run PULSE

- Indication that the system is standby when conditional execution for output port, pulse/delay operations are active (with ON command).

4.3.3 Input/Output State

4.3.3.1 I/O State Request Command

	Command	Data	Remark
Byte Size	2	1	
	“03” (0x30, 0x33) State Request	0x30: Fixed to a dummy value	

4.3.3.2 Response for I/O State Request Command

- Identical with ON/OFF control response.
- Sent even if the state of input port or ADC (in Switch Input mode) has been changed (based on a threshold voltage). A change of input state is recognized when the input value changed is maintained for 15 ms.
- In case of state response, the control state field returns 0x30 in Run mode and 0x31 in Setting mode.

4.3.4 Set/Run

4.3.4.1 Set/Run Command

	Command	Data	Remark
Byte Size	2	1	
	“04” (0x30, 0x34) Set/Run	0x30: Set 0x31: Run	Switch between Set and Run Modes.

- The Rhio232 returns a state response once after set command is received. It does not return any state response until it receives run command even if it is switched to Setting mode.
- When setting output port, ADC input port, Port Enable, Port Power-out Recovery or serial number, send set command first to switch it to Setting mode. (Check command is working in both Setting and Run mode.)
- Once each setting is completed, send run command to switch it to Run mode.
- When a run command is received, it returns a state response once and continues with the operation paused.

4.3.4.2 Response for Set/Run command

- Same as "4*.3.2.2 Response for ON/OFF control*".
- In Setting mode, the Rhio232 returns a response once and it does not respond until run command is received.
- The control state field of the state response, the Rhio232 returns 0x31 in Setting mode and 0x30 in Run mode.

4.3.5 Output Port Setting

4.3.5.1 Set Output Port Command

	Command	Data			Remark
Byte Size	2	2	1	N	
	"05" (0x30, 0x35) Set output port.	Port No. "01" ~ "10"	0x30: Clear run condition 0x31: Set run condition 0x32: Clear Delay or Pulse 0x33: Set Delay/Pulse 0x39: Clear all settings	Setting Data (Max. 106 bytes)	Set the attribute of each output port. (When cleared, the setting data is 0x30 in 1 byte.)

- A setting can be done only in Setting mode.
  Users can set repeated run condition, pulse or delay. When a port is set for repeated actions, run condition set up has a priority, which means it is executed first. A setting can be cleared by mode, or the entire setting can be cleared as well. When cleared, data setting becomes 0x30 in 1 byte.

  1) Set run condition.
     It sets the run condition for the output port specified.
     (It is run only when an output port is ON and will set the port as OFF when it is configured as such.)
     - Input/Output port state + a conditional operator + Input/Output port state (AND/OR)
     - A conditional operator + Input/Output port state (NOT)
     - AND => &, OR => |, NOT => !
     - A conditional expression only takes the form of a single expression with AND, OR, NOT.
       Eg. AND operation of Input #1 and Input #2 : I1 & I2
       OR operation of Output #3 and Input #1 : O3 | I1
       AND operation of the inversed Input #10 and Output #10 : !I10 & O10
       Inverse operation of Input #1 : !I1
     - The number of points that can be specified for run condition per 1 output point in setting data must be less than 21 points.
       Eg. I1&I2&I3|I4|!I5|I6&I7&I8&I9&I10|!I11|I12|

O2|O3|O4|O5|O6|O7|O8|O9|O10

- An output port to be set should not be included in run condition.
  Eg. When setting O1, it should not be included in its run condition expression.

2) Set Delay/Pulse.

- It specifies the output to toggle between ON/OFF at a given time or repeatedly. (Delay ON and Pulse are enabled with ON control command and disabled with OFF control. Delay OFF is enabled with OFF control.)

- Setting Data

	Setting Data (time)	
	ON Time	OFF Time
Byte Size	5	5
Run Setting	“00000”-“50000”	“00000”-“50000”

A setting can be specified in 100 ms and allowed up to 500 sec.

Eg.

ON Time	OFF Time	Remark
“00000”	“00000”	No Delay/Pulse
“00001”	“00000”	ON after 100 ms delay
“00020”	“00000”	ON after 2 sec delay
“50000”	“00000”	ON after 5,000 sec delay
“00000”	“00001”	OFF after 100 ms delay
“00000”	“00020”	OFF after 2 sec delay
“00000”	“50000”	OFF after 5,000 sec delay
“00001”	“00001”	Repeat 100 ms ON and 100 ms OFF.
“00020”	“00020”	Repeat 2 sec ON and 2 sec OFF.
“50000”	“50000”	Repeat 5,000 sec ON and 5,000 sec OFF.
“00010”	“00030”	Repeat 1 sec ON and 3 sec OFF.
“00300”	“00150”	Repeat 30 sec ON and 15 sec OFF.

4.3.5.2 Response for Set Output Port command

	Response	Data				Remark
Byte Size	2	1	2	1	N	
	"06" (0x30, 0x36) Return setting.	Setting Flag	Port No. "01" - "10"	0x30: Clear run condition 0x31: Set run condition 0x32: Clear Delay/Pulse 0x33: Set Delay/Pulse 0x39: Clear all settings	Setting Data	Return the setting of an output port.

- Setting Flag
  0x30: Set OK
  0x39: Set NG
  0x31: Not in Setting mode (when in Run mode)
  0x32: Unspecified (If the user attempts to clear in unspecified state, it will return 0x32.)
  (When it is in unspecified state, the setting data is 0x30 in 1 byte.)
- When setting is cleared, the setting data is 0x30 in 1 byte.

4.3.5.3 Check Output Port Setting Command

	Command	Data		Remark
Byte Size	2	2	1	
	"07" (0x30, 0x37) Check setting.	Port No. "01" - "10"	0x31: Check the run condition setting. 0x33: Check the Delay/Pulse setting	Check the attribute of each output port.

4.3.5.4 Response for Check Output Port Setting command

- Same as "*4.3.5.2 Response for Set Output Port command*".

4.3.6 ADC Input Port Setting

4.3.6.1 Set ADC Input Port Command

	Command	Data	Remark
Byte Size	2	16	
	“08” (0x30, 0x38) Set ADC.	Set the threshold level value for an input ADC port. (“0000”-”1023”) X 4	Set it to a 10-bit ADC level value.

- A threshold value is set for all ADC inputs in Switch Input mode.
- When an input change is measured, only changes beyond the range of +8 - -8 are recognized as an input change (based on the threshold level). It is determined as ON when an input voltage is more than the threshold level by 8 or more and OFF when an input voltage is less than the threshold level by 8 or more.
  (When the threshold level is above 1015, a change between 1015-1023 is determined as ON. When it is set to less than 8, a change between 8-0 is determined as OFF.)

ADC Input No.	1	2	3	4
Settings	“0000”-”1023”	“0000”-”1023”	“0000”-”1023”	“0000”-”1023”

4.3.6.2 Response for Set ADC Input Port command

	Response	Data		Remark
Byte Size	2	17		
	“09” (0x30, 0x39) Return the ADC setting.	Setting FLAG	(“0000”-”1023”) X 4	The threshold level for input ADC port

- Setting Flag
  0x30: Set OK
  0x31: Not in Setting mode (in Run mode)
  0x39: Set NG

4.3.6.3 Check ADC Input Port Setting Command

	Command	Data	Remark
Byte Size	2	1	
	“10” (0x31, 0x30) Check the ADC setting	0x30: Fixed to a dummy value	Check the threshold level value of an input ADC port.

4.3.6.4 Response for Check ADC Input Port Setting command

- Same as "4.*3.6.2 Response for Set Port command*".
- The setting field is always OK (0x30).
- The factory default value in Level Input mode is set to "0000".

4.3.7 Port Enable Setting

4.3.7.1 Set Port Enable command

	Command	Data	Remark
Byte Size	2	34	
	“11” (0x31, 0x31) Set to Enable.	Port Enable/Disable setting data	Set all ports to Enable/Disable.

- Data

Field	Data Order	Port	Data Content
ADC Port	1	A1	0x31: Level Input mode 0x32: Switch Input mode
	2	A2	
	3	A3	
	4	A4	
	5	-	Field Separator 0x2C (“,”)
ADC Reference Setting	6	-	0x30: AVcc (Vcc 5V) -> default 0x31: Internal (2.56V) 0x32: External (2-4.5V)
	7	-	Field Separator 0x2C (“,”)

Input Port	8-11	I1-I4	0x31: Enable, 0x32: Disable
	12	-	Field Separator 0x2C (“,”)
	13-16	I5-I8	0x31: Enable, 0x32: Disable
	17	-	Field Separator 0x2C (“,”)
	18-21	I9-I12	0x31: Enable, 0x32: Disable
	22	-	Field Separator 0x2C (“,”)
Output State	23-26	O1-O4	0x31: Enable, 0x32: Disable
	27	-	Field Separator 0x2C (“,”)
	28-31	O5-O8	0x31: Enable, 0x32: Disable
	32	-	Field Separator 0x2C (“,”)
	33,34	O9, O10	0x31: Enable, 0x32: Disable

All I/O ports are initially set to Enable (including Factory Reset).

All ADC ports are initially set to Level Input mode (including Factory Reset).

4.3.7.2 Response for Set Port Enable command

	Response	Data	Remark
Byte Size	2	35	
	“12” (0x31, 0x32) Return the Enable setting.	Port Enable/Disable setting data	Return all port Enable/Disable settings.

- Data

Field	Data Order	Port	Data Content
Setting	1	-	0x30: Normal Enable, 0x31: Not in Setting mode 0x39: Abnormal Enable
ADC Port	2	A1	0x30: Avcc (Vcc 5V) -> default 0x31: Internal (2.56V) 0x32: External (2V-4.5V)
	3	A2	
	4	A3	
	5	A4	
	6	-	Field Separator 0x2C (“,”)

ADC Reference Setting	7	-	0x30: Avcc (Vcc 5V) -> default 0x31: Internal (2.56V) 0x32: External (2V-4.5V)
Input Port	9-12	I1-I4	0x31: Enable, 0x32: Disable
	13	-	Field Separator 0x2C (“,”)
	14-17	I5-I8	0x31: Enable, 0x32: Disable
	18	-	Field Separator 0x2C (“,”)
	19-22	I9-I12	0x31: Enable, 0x32: Disable
	23	-	Field Separator 0x2C (“,”)
Output State	24-27	O1-O4	0x31: Enable, 0x32: Disable
	28	-	Field Separator 0x2C (“,”)
	29-32	O5-O8	0x31: Enable, 0x32: Disable
	33	-	Field Separator 0x2C (“,”)
	34,35	O9, O10	0x31: Enable, 0x32: Disable

4.3.7.3 Check Port Enable Setting command

	Command	Data	Remark
Byte Size	2	1	
	“13” (0x31, 0x33) Check the Enable setting.	0x30: Fixed to a dummy value	Check all port Enable/Disable setting.

4.3.7.4 Response for Check Port Enable Setting command

- Same as "4.*3.7.2 Response for Set Port Enable command*".
- The setting field returns Normal (0x30) when it returns.
- The initial value is set to Enable for all ports at the time of shipment.
  (ADC is set to Level Input mode.)

4.3.8 Port Power-out Post Recovery Setting

4.3.8.1 Set Port Power-out Post Recovery command

	Command	Data	Remark
Byte Size	2	12	
	“14” (0x31, 0x34) Set Power-out Post Recovery to Enable/Disable.	The Port Power-out Post Recovery setting data	Set Power-out Post Recovery to Enable/Disable for an output port

- Data

Field	Data Order	Port	Data Content
Output State	1-4	O1-O4	0x31: Enable, 0x32: Disable
	5	-	Field Separator 0x2C (“,”)
	6-9	O5-O8	0x31: Enable, 0x32: Disable
	10	-	Field Separator 0x2C (“,”)
	11, 12	O9, O10	0x31: Enable, 0x32: Disable

4.3.8.2 Response for Set Port Power-out Post Recovery command

	Response	Data	Remark
Byte Size	2	13	
	“15” (0x31, 0x35) Return the Power-out Post Recovery setting.	The Port Power-out Post Recovery setting data	Return the Power-out Post Recovery setting for an output port.

- Data

Field	Data Order	Port	Data Content
Setting	1	-	0x30: Set OK, 0x31: Not in Setting mode,

			0x39: Set NG
Output State	2-5	O1-O4	0x31: Enable, 0x32: Disable
	6	-	Field Separator 0x2C (“,”)
	7-10	O5-O8	0x31: Enable, 0x32: Disable
	11	-	Field Separator 0x2C (“,”)
	12, 13	O9, O10	0x31: Enable, 0x32: Disable

4.3.8.3 Check Port Power-out Post Recovery Setting command

	Command	Data	Remark
Byte Size	2	1	
	“16” (0x31, 0x36) Check the Power-out Recovery setting.	0x30: Fixed to a dummy value	Check the Power-out Recovery setting for all ports.

4.3.8.4 Response for Check Port Power-out Recovery Setting command

- Same as "4.*3.8.2 Response for Set Port Power-out Recovery command*".
- The setting field returns Normal (0x30) when it returns.

The initial value is set to Enable for all ports at the time of shipment.

5. How to use the Rhio232 with Device Servers

The Rhio232 is a serial I/O manager that enables device servers to monitor and control I/O devices. It is designed to connect to a device server through the RS232 interface. The following is the typical way to use the Rhio232 with Sena device servers for remote I/O management application.

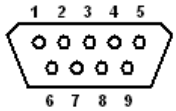
- Connection by TCP socket program using Rhio Library: Sena device server + Rhio232
- Connection by Serial program (COM port) using Rhio Library: Serial/IP COM Port Redirector + Sena device server + Rhio232

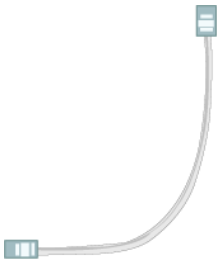
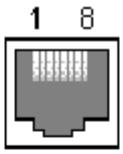
Please refer to the *tips_serial_ip_with_sena_ds-v1.0.0.pdf* or Serial/IP User Manual for detailed information on the Serial/IP software. Users may get the corresponding material from product package CD or http://www.sena.com/korean/support/downloads/.
This chapter covers how to connect the Rhio232 with the device server and the application configuration.

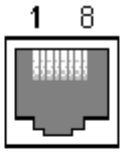
5.1 Connections

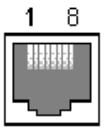
The Rhio232 is connected to the serial device server through RS232 serial port and its connector type is RJ45. Users should use RJ45-DB9 female straight adapter in the package in order to connect the Rhio232 to the Sena device servers that has DB9 serial port, i.e. LS100, PS100. Users may directly use CAT5 straight cable to connect it to the Super series or STS series models that have RJ45 serial port. Table 5-1 shows the summary of the connections between Sena device servers and the Rhio232.

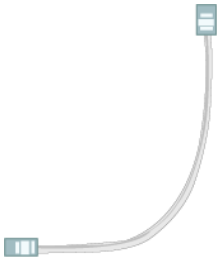
Table 5-1. Connections with the Sena device servers and the Rhio232

<table>
<tr><th>Sena device servers</th><th>Connection</th><th>Rhio232</th></tr>
<tr><td>DB9:
 

<table><tr><th>Pin</th><th>Description</th></tr><tr><td>1</td><td>-</td></tr><tr><td>2</td><td>Rx</td></tr><tr><td>3</td><td>Tx</td></tr><tr><td>4</td><td>DTR</td></tr><tr><td>5</td><td>GND</td></tr><tr><td>6</td><td>DSR</td></tr><tr><td>7</td><td>RTS</td></tr><tr><td>8</td><td>CTS</td></tr><tr><td>9</td><td>-</td></tr></table>
eg.
LS100, LS100W,
PS100/200/400/110/410/810,
SS100</td><td>RJ45-DB9 female straight adapter
 
+
CAT5 straight cable
  </td><td rowspan="2">RJ45:
 

<table><tr><th>Pin</th><th>Description</th></tr><tr><td>1</td><td>CTS</td></tr><tr><td>2</td><td>DSR</td></tr><tr><td>3</td><td>RxD</td></tr><tr><td>4</td><td>GND</td></tr><tr><td>5</td><td>DCD</td></tr><tr><td>6</td><td>TxD</td></tr><tr><td>7</td><td>DTR</td></tr><tr><td>8</td><td>RTS</td></tr></table>
 

<table><tr><th>Pin</th><th>Description</th></tr><tr><td>1</td><td>CTS</td></tr><tr><td>2</td><td>DSR</td></tr><tr><td>3</td><td>RxD</td></tr><tr><td>4</td><td>GND</td></tr><tr><td>5</td><td>DCD</td></tr><tr><td>6</td><td>TxD</td></tr><tr><td>7</td><td>DTR</td></tr><tr><td>8</td><td>RTS</td></tr></table></td></tr>
<tr><td>RJ45:
 

<table><tr><th>Pin</th><th>Description</th></tr><tr><td>1</td><td>CTS</td></tr><tr><td>2</td><td>DSR</td></tr><tr><td>3</td><td>RxD</td></tr><tr><td>4</td><td>GND</td></tr><tr><td>5</td><td>DCD</td></tr><tr><td>6</td><td>TxD</td></tr><tr><td>7</td><td>DTR</td></tr><tr><td>8</td><td>RTS</td></tr></table>
eg.
SS110/400/800
STS800/1600</td><td>CAT5 straight cable
  </td></tr>
</table>

5.2 Application

This chapter covers how to test the Rhio232 with the Sena's 8-port device server, SS800 by using Serial/IP software. The following is the units required to test and the corresponding configurations for each module.

Rhio232 and SS800

Serial/IP COM Port Redirector software

Rhio Manager software

Two CAT5 straight cable (One for the network connection of the SS800 and the other for the connection between the SS800 and the Rhio232)

- SS800 configuration

  IP address(DHCP): 192.168.222.21

  Serial port #1 configuration:

  Host mode: TCP

  TCP Local port: 7001

  Serial: RS232 type / 9600 Baud rate / 8 Data bit / None Parity / 1 Stop bit

- Serial/IP COM Port Redirector software configuration

  Use COM Port #10

It is assumed that Serial/IP COM Port Redirector and Rhio Manager software are already installed on the users' PC and that the Rhio232 is connected to the appropriate I/O devices. Figure 5-1 shows the application diagram of the SS800 with the Rhio232.

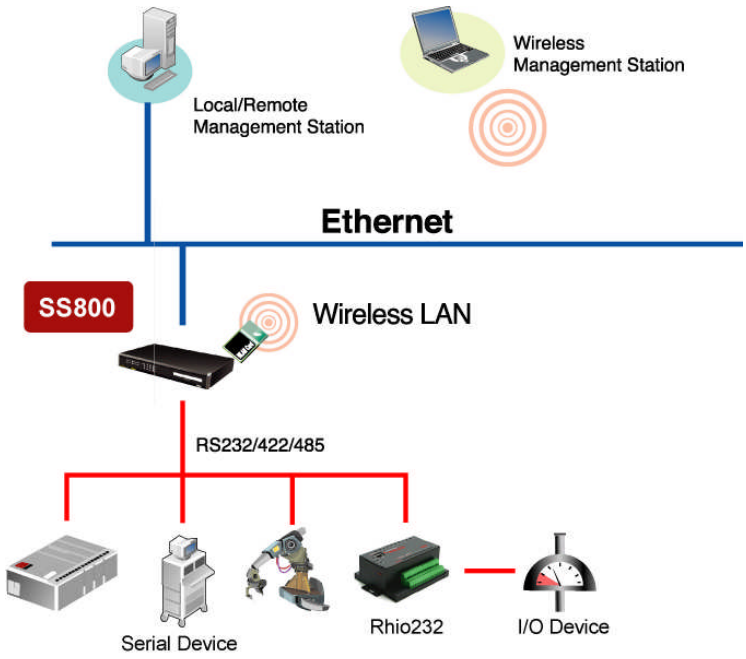
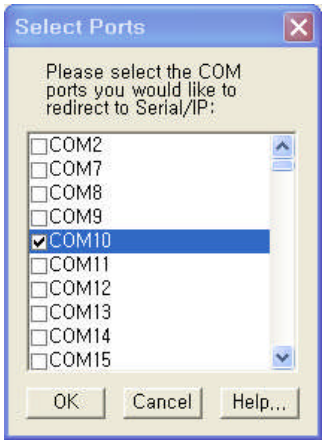


Figure 5-1 Application diagram of the Rhio232 and the SS800

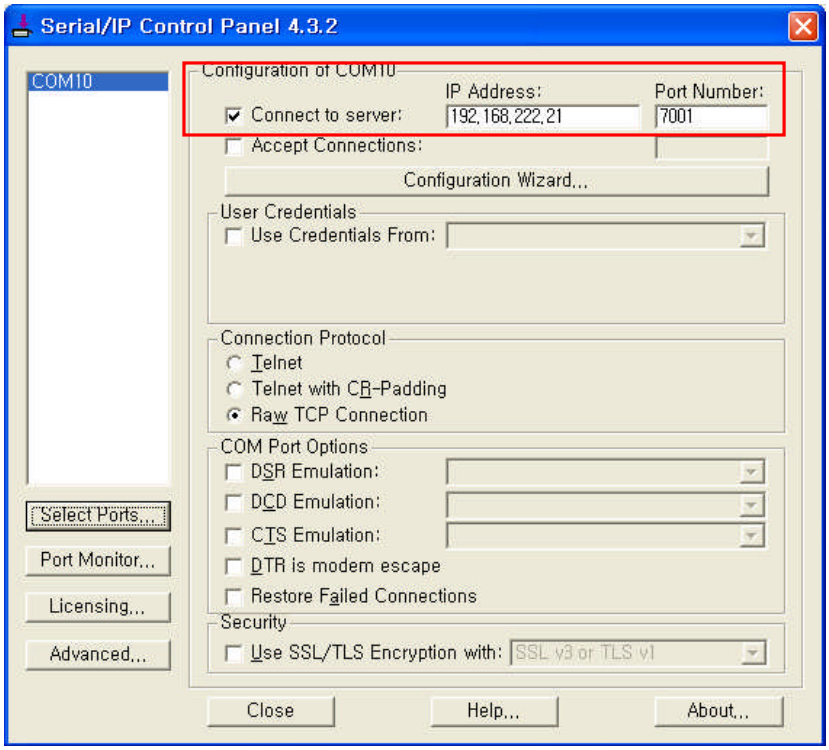
Step 1. Connect the Rhio232 to the port #1 of the SS800 using CAT5 cable.

Step 2. Serial/IP Configuration

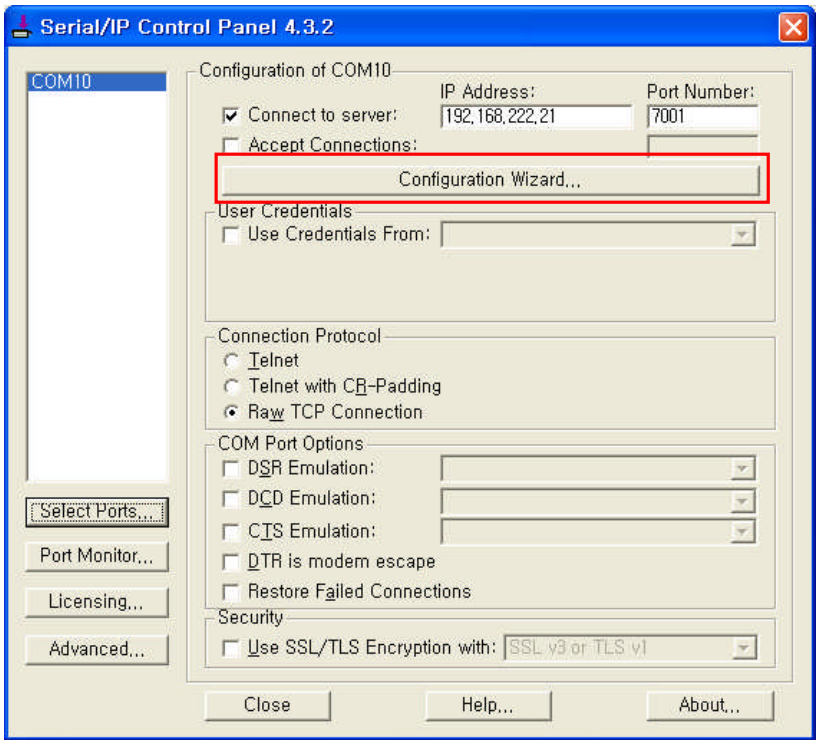
① Double click the Serial/IP Tray application and then select COM10 as a virtual serial port after clicking [Select Ports] on the panel.



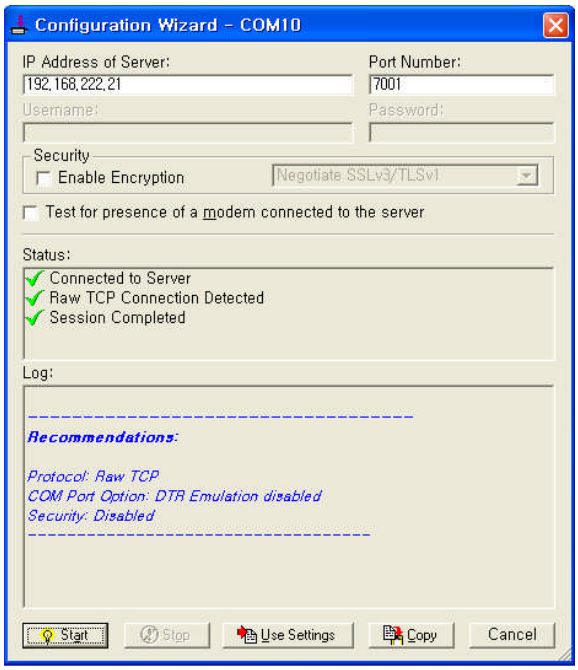
② Enter the IP Address and Port number so that it may be same as the one of the SS800.



③ Click the [Configuration Wizard] button on the Serial/IP Control Panel.



④ Complete COM port setting by clicking [Start] button on the Configuration Wizard panel. Close the window by clicking [Use Setting] button after the communication test.

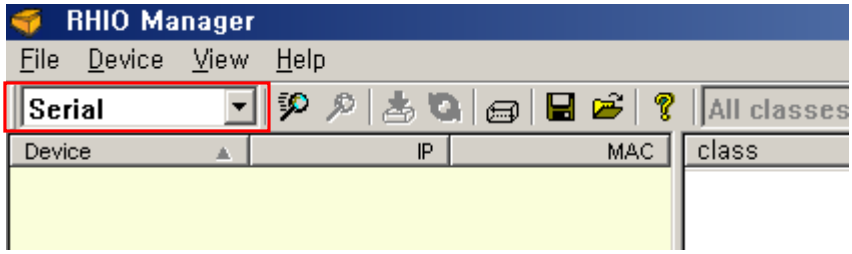


The configuration makes the system work as follows.

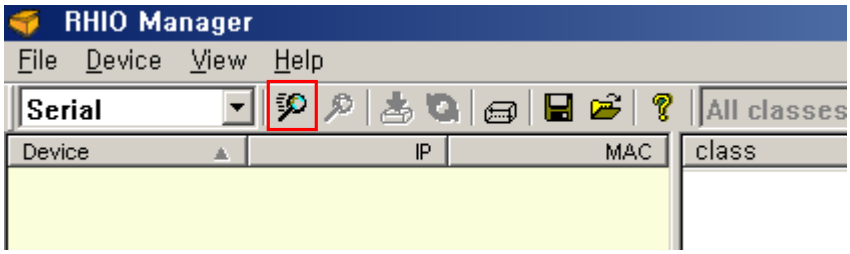
- The data stream from users' application is transferred to the virtual port COM10.
- The data stream from COM10 is transferred to the IP address/TCP port of the SS800, 192.168.222.21, TCP port 7001.

Step 3. I/O test using Rhio Manager

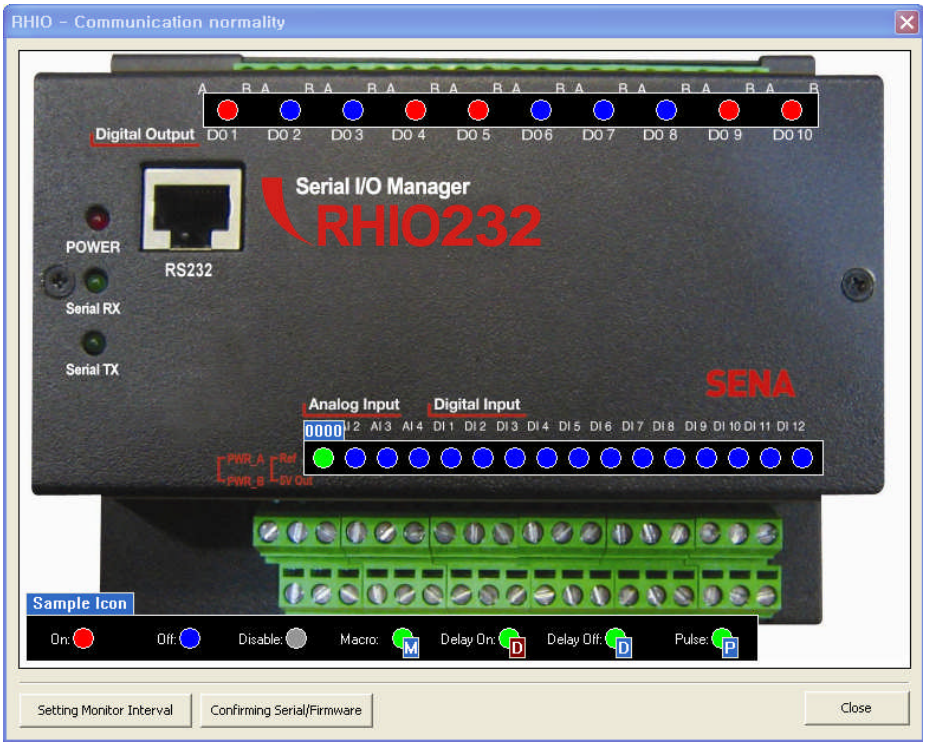
① Run the Rhio Manager and then select [Serial] for communication type.



② Configure COM port after clicking [Probe] button the toolbar Users can now connect to the Rhio232 which is connected to the SS800 by way of the SS800.



③ Click the [I/O CONTROL] button and then it will open Rhio Manager I/O management panel. Users can remotely control and monitor the I/O devices connected to the Rhio232 through the COM10 port of the PC.



Users can remotely manage the I/O devices by using such components as Serial/IP, SS800 and Rhio232. For the real world applications, users may write their own code to communicate with the Rhio232 by using Rhio Library and may integrate them with the overall system management code they have.

Appendix A. Connection

A.1 Serial Port Pin Outs

The pin assignment of the RHIO232 RJ45 connector is summarized in Table A-1. Each pin has a function according to the serial communication type configuration.

Note : When connecting serial port to host, users should use the other end of CAT5 cable using RJ45 to DB9 Female connector(included package).

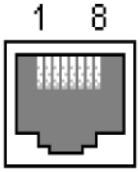


Figure A-1. Pin layout of the RJ45 serial connector

Pin	Description
1	RTS
2	DTR
3	TxD
4	GND
5	-
6	RxD
7	DSR
8	CTS

Table A-1. Pin assignment of the RJ45 serial connector

A.2 Serial Port Wiring Diagram

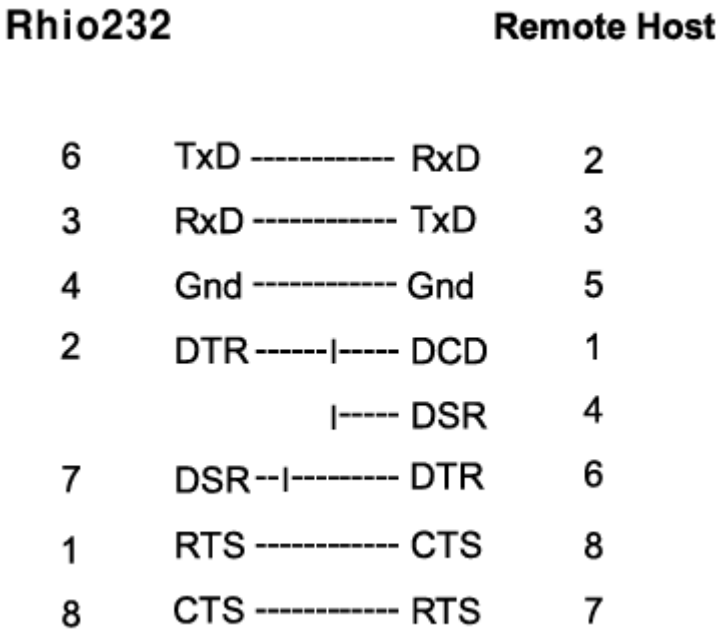


Figure A-5. RS232 serial port wiring diagram

Appendix B Troubleshooting

B.1 Power/LED Status Troubleshooting

Problem	Cause	Action
Power LED does not light up	Power cable is not connected	Check power connection. The Rhio232's power specification is DC 9V~48V.
Serial Rx LED does not light up	Serial cable is not connected	Check serial cable connection
	Invalid serial cable is used	Must use serial cable (or CAT5 Straight Cable + DB9F Straight adaptor).
Serial Tx LED does not light up	Serial cable is not connected	Check serial cable connection
	Invalid serial cable is used	Must use serial cable (or CAT5 Straight Cable + DB9F Straight adaptor).

B.2 Serial Console Troubleshooting

Problem	Cause	Action
The Rhio232 is not communicated	Serial cable is not connected	Check serial cable connection
	Invalid serial cable is used	Must use serial cable (or CAT5 Straight Cable + DB9F Straight adaptor).
	Serial Baud rate is incorrect	Check serial port configuration of host or device server: 9600 bps, 8 Data bits, No parity, 1 stop bit, Hardware flow control

Appendix C. Rhio Linrary

C.1 Enumeration Type

`EOnOffFlag`

Flag for ON, OFF, Operation condition

```
enum EOnOffFlag
{
        EOF_ON,                 //ON status
        EOF_ON_ADC_LEVEL,       //ADC Level ON status
        EOF_OFF,                //OFF status
        EOF_NOT,                // Status uncertain status
        EOF_ON_DELAY,           //Delay ON waiting status
        EOF_OFF_DELAY,          //Delay OFF waiting status
        EOF_ONOFF_MACRO,        //Macro(condition) waiting status
        EOF_ONOFF_PULSE         //Pulse status
};
```

`SendStatusFlag`

Transfer status flag for indicating which command is sent to The Rhio232

```
enum SendStatusFlag
{
        ESF_ONOFF,              //ON, OFF control command transfer status
        ESF_SET,                //
        ESF_SET_MODE,           // Set mode switch command transfer status
        ESF_RUN_MODE,           //operation mode change command transfer status
        ESF_SET_MACRO,          //Macro Set Command transfer status
        ESF_SET_DELAY_PULSE,    //Delay/Pulse Set Command transfer status
        ESF_SET_ADC,            //ADC Set Command transfer status
        ESF_SET_ENABLE,         // Port Enable/Disable Set command transfer status
        ESF_SET_PWR_STOP,       // Power-out Post Recovery Enable/Disable Set Command transfer status
        ESF_SET_FACTROT_RESET,  //Factory Reset Command transfer status
        ESF_SET_SERIAL,         //Serial number Set Command transfer status
        ESF_MON_MACRO,          //Macro check Command transfer status
        ESF_MON_DELAY_PULSE,    //Delay/Pulse check Command transfer status
```

```
        ESF_MON_ADC,            //ADC check Command transfer status
        ESF_MON_ENABLE,         //Enable/Disable check Command transfer status
        ESF_MON_PWR_STOP,       //Power-out Post Recovery check Command transfer status
        ESF_MON_SERIAL,         //Serial number check Command transfer status
        ESF_MON_FIRMWARE,       //Firmware Version check transfer status
        ESF_NONE
};
```

EsetOutputFlag

Output Port Maco/Delay/Pulse Set/Clear Flag

```
enum ESetOutputFlag
{
        ESOF_CLEAR,             // Port Set Clear
        ESOF_SETTING            // Port Set
};
```

EADCMode

Flag specifies whether the current ADC mode is Level or Switch mode

```
enum EADCMode
{
        EAM_LEVEL,
        EAM_SWITCH
};
```

C.2 Structure

ON/OFF status Data

Structure which contains Port ON/OFF status

```
typedef struct _ADCData                 //ADC Data
{
        EOnOffFlag eADC;                //ADC On/OFF Flag
        char cADC[5];                   //ADC Level storage
}SADCData;

typedef struct _OnOffStatusData         //All Port ON/OFF Data
{
        EOnOffFlag eOutput[10];         //Output    Port 1~10
        EOnOffFlag eInput[12];          //Input     Port 1~12
        SADCData sADC[4];               //ADC     Port 1~4
}SOnOffStatusData;
```

Set status Data

Structure for the Set status

```
typedef struct _SetOutput               //Output    Port Set Data
{
        ESetOutputFlag eMacro;          //MACRO Flag : ESOF_CLEAR, ESOF_SETTING
        char cMacro[106];               //MACRO Data
        ESetOutputFlag eDelayPulse;     //DelayPulse Flag : ESOF_CLEAR, ESOF_SETTING
        char cDelayPulse[11];           //DelayPulse Data
        bool isEnable[10];              //Output port Enable
        bool isEnablePowerStop[10];     //Output port power stoppage
}SSetOutput;

typedef struct _SetADC                  //ADC Set Data
{
        EADCMode eModeADC[4];           //ADC port Mode
        BYTE bReference;                //ADC Reference
        SADCData sADCData[4];           //ADC Level Data
}SSetADC;
```

```
typedef struct _SetInput                //Input    Port Set Data
{
        bool isEnable[12];              //Input port Enable
}SSetInput;

typedef struct _RHIOSetting             //All Port Set Data
{
       SSetOutput sOutput;              //Setting Output port Data;
       SSetADC sADC;                    //Setting ADC port Data;
       SSetInput sInput;                //Setting Input port Data;
       char cGetSerial[18];             // Model name and Serial number received
       char cGetFirmVer[9];             // Firmware Version received
}SRHIOSetting;
```

C.3 Function

`RHIO_CommProcessCreate`

Description: It creates Process Class of `RHIO_Proc Dll`. In order to use Rhio Dll, be sure to create the process using `RHIO_CommProcessCreate`.

Function Prototype:

```
extern "C" __declspec(dllexport) void RHIO_CommProcessCreate
                                        (CWnd *pParentWnd);
```

Parameter:

CWnd *pParentWnd : Window Pointer to Parent window of Process Class

Return : None

eg. `RHIO_CommProcessCreate(this)`

`RHIO_SockConnect`

Description: Used when users want to connect to RHIO using TCP Socket connection.

Function Prototype:

```
extern "C" __declspec(dllexport) bool RHIO_SockConnect
(BYTE bAddr1, BYTE bAddr2, BYTE bAddr3, BYTE bAddr4, int iPort);
```

Parameter:

```
  BYTE bAddr1 : 1st byte of the IP Address
  BYTE bAddr2 : 2nd byte of the IP Address
  BYTE bAddr3 : 3rd byte of the IP Address
  BYTE bAddr4 : 4th byte of the IP Address
  int iPort :  Port number.
```

Return: TRUE if successful, False if failed.

eg :

```
    if(RHIO_SockConnect(192, 168, 100, 2, 6001))
         AfxMessageBox("Connection Successful");
   else
         AfxMessageBox("Connection Failure");
```

`RHIO_CommConnect`

Description: Used when users want to connect to RHIO using RS232 serial connection.
(Parameter fixed as Baudrate:9600 bps, Parity: none, Data bit: 8, Stop bit: 1)

Function Prototype:

```
    extern "C" __declspec(dllexport) bool RHIO_CommConnect (int iPort);
```

Parameter: None

Return: TRUE if successful, False if failed.

eg.

```
      if(RHIO_CommConnect(1)) AfxMessageBox("Connection Successful.");
      else                    AfxMessageBox("Connection Failure.");
```

`RHIO_Close`

Description : Used when users want to disconnect the current connection with the Rhio in both serial and TCP connection. In order to reconnect to the Rhio, be sure to use Rhio-Close function before connection attempt.

Function Prototype:

```
       extern "C" __declspec(dllexport) bool RHIO_Close();
```

Parameter: None

Return: TRUE if successful, False if failed.

eg. : `RHIO_Close();`

RHIO_SndCmd_SetOnOff

Description : Used to send the command to turn ON/OFF an ouput port.

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_SndCmd_SetOnOff
                                  (int iOutputNum, EOnOffFlag eOnOff);
```

Parameter :

```
int iOutputNum :  Output port number (1~10)
EOnOffFlag eOnOff : Control flag (EOF_ON, EOF_OFF)
```

Return : TRUE if successful, False if failed.

Parameter when reaction event is received :

WPARAM : Command transfer status flag (ESF_ONOFF)

LPARAM : Control status flag of the command (False if Time Out Error)

(0x30: Normal operation, 0x31 : Not the Run Mode, 0x32: Abnormal operation)

eg : `RHIO_SndCmd_SetOnOff(1, EOF_ON);`

RHIO_SndCmd_GetOnOff

Description : Used to send the command to get ON/OFF status of all the ports

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_SndCmd_GetOnOff();
```

Parameter : None

Return : TRUE if successful, False if failed.

eg : `RHIO_SndCmd_GetOnOff();`

RHIO_SndCmd_SetSettingMod

Description : Used to send the command to switch to the Set Mode.

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_SndCmd_SetSettingMode()
```

Parameter : None

Return : TRUE if successful, False if failed.

Parameter when reaction event is received :

WPARAM : Command transfer status Flag (ESF_SET_MODE)

LPARAM : Control status flag of the command (False if Time Out Error)

(0x30: Normal operation, 0x31 : Not the Run Mode, 0x32:Abnormal operation)

eg : `RHIO_SndCmd_SetSettingMode();`

RHIO_SndCmd_SetRunMode

Description : Used to send the command to switch to the Run Mode

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_SndCmd_SetRunMode()
```

Parameter : None

Return : TRUE if successful, False if failed.

Parameter when reaction event is received :

WPARAM : Command transfer status Flag (ESF_RUN_MODE)

LPARAM : Control status Flag of the command (False if Time Out Error)

(0x30: Normal operation, 0x31 : Not the Run Mode, 0x32:Abnormal operation)

eg : `RHIO_SndCmd_SetRunMode();`

RHIO_SndCmd_SetMACRO

Description : Used to send the command to set/clear the MACRO of an input port

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_SndCmd_SetMACRO
                                (SRHIOSetting sSetData, int iPortNum);
```

Parameter :

```
SRHIOSetting sSetData : Data to set
SRHIOSetting.sInput.eMacro : Flag to determine to set or clear
                            (ESOF_CLEAR, ESOF_SETTING)
SRHIOSetting.sInput.cMacro : MACRO string
int iPortNum : Input port number to set
```

Return : TRUE if successful, False if failed.

Parameter when reaction event is received :

WPARAM : Command transfer status Flag (ESF_SET_MACRO)

LPARAM : Control status flag of the command (False if Time Out Error)

(0x30:Normal operation, 0x39:Abnormal operation, 0x31:Not the Set Mode, 0x32:Not set)

eg :

In case of Set command :

```
SRHIOSetting sSetData;
sSetData.sInput.eMacro = ESOF_SETTING;
memset(sSetData.sInput.cMacro, '\0', sizeof(sSetData.sInput.cMacro));
memcpy(sSetData.sInput.cMacro, "O2&O4|I1", 8);
RHIO_SndCmd_SetMACRO(sSetData, 1);
```

In case of Set Clear command :

```
SRHIOSetting sSetData;
sSetData.sInput.eMacro = ESOF_CLEAR;
RHIO_SndCmd_SetMACRO(sSetData, 1);
```

RHIO_SndCmd_GetMACRO

Description : Used to send the command to check whether it is possible to get the MACRO of an input port

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_SndCmd_GetMACRO
                                                    (int iPortNum);
```

Parameter :

```
int iPortNum : Input port number to check
```

Return : TRUE if successful, False if failed.

Parameter when reaction event is received :

WPARAM : Command transfer status Flag (ESF_MON_MACRO)

LPARAM : Control status flag of the command (False if Time Out Error)

(0x30:Normal operation, 0x39:Abnormal operation, 0x31:Not the Set Mode, 0x32:Not set)

※ When reaction event is received, get data using RHIO_GetSettingData function and check the MACRO on setting.

eg : `RHIO_SndCmd_GetMACRO(1);`

RHIO_SndCmd_SetDelayPulse

Description : Used to send the command to set the Delay/Pulse of an input port

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_SndCmd_SetDelayPulse
                                (SRHIOSetting sSetData, int iPortNum);
```

Parameter :

```
SRHIOSetting sSetData : Data to set
SRHIOSetting.sInput.eDelayPulse : Flag to determine to set or clear
                                  (ESOF_CLEAR, ESOF_SETTING)
```

SRHIOSetting.sInput.cDelayPulse : Time string in 10 bytes

Byte Size	5(ON Time)	5(OFF Time)
Operation Set	“00000” ~ “50000”	“00000” ~ “50000”

int iPortNum : Input port number to set

Return : TRUE if successful, False if failed.

Parameter when reaction event is received :

WPARAM : Command transfer status flag (ESF_SET_DELAY_PULSE)

LPARAM : Control status flag of the command (False if Time Out Error)

(0x30:Normal operation, 0x39:Abnormal operation, 0x31:Not the Set Mode, 0x32:Not set)

eg :

In case of Set command :

```
SRHIOSetting sSetData;
sSetData.sInput.eDelayPulse = ESOF_SETTING;
memset(sSetData.sInput.cDelayPulse,'\0',
       sizeof(sSetData.sInput.cDelayPulse));
memcpy(sSetData.sInput.cDelayPulse, "0050000000”, 10)
RHIO_SndCmd_SetDelayPulse(sSetData, 1);
```

In case of Set Clear command :

```
SRHIOSetting sSetData;
sSetData.sInput.eDelayPulse = ESOF_CLEAR;
RHIO_SndCmd_SetDelayPulse(sSetData, 1);
```

RHIO_SndCmd_GetDelayPulse

Description : Used to send the command to check whether it is possible to get the Delay/Pulse of an input port

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_SndCmd_GetDelayPulse
                                                    (int iPortNum);
```

Parameter :
`int iPortNum : Input port number to check`

Return : TRUE if successful, False if failed.

Parameter when reaction event is received :

`WPARAM` : Command transfer status flag (ESF_MON_DELAY_PULSE)

`LPARAM` : Control status flag of the command (False if Time Out Error)

(0x30:Normal operation, 0x39:Abnormal operation, 0x31:Not the Set Mode, 0x32:Not set)

※ When reaction event is received, get data using RHIO_GetSettingData function and check the Delay/Pulse on setting.

eg : `RHIO_SndCmd_GetDelayPulse(1);`

`RHIO_SndCmd_SetADC`

Description : Used to send the command to set the Level of all the ADC ports from 1 to 4.

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_SndCmd_SetADC
                                        (SRHIOSetting sSetData);
```

Parameter :

`SRHIOSetting sSetData : Data to set`

`SRHIOSetting.sADC.sADCData[index].cADC : ADC Level string in 4 bytes between "0000" and "1023"`

Return : TRUE if successful, False if failed.

Parameter when reaction event is received :

`WPARAM` : Commnad transfer status flag (ESF_SET_ADC)

`LPARAM` : Control status flag of the command (False if Time Out Error)

(0x30:Normal operation, 0x39:Abnormal operation, 0x31:Not the Set Mode)

eg :

```
SRHIOSetting sSetData;
```

```
for(int index = 0; index < 4; index++)
{
      memset(sSetData.sADC.sADCData[index].cADC,'\0',
             sizeof(sSetData.sADC.sADCData[index].cADC));
      memcpy(sSetData.sADC.sADCData[index].cADC, "0512", 4);
}
RHIO_SndCmd_SetADC(sSetData);
```

RHIO_SndCmd_GetADC

Description : Used to send the command to check whether it is possible to get the Level of all the ADC ports.

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_SndCmd_GetADC();
```

Parameter : None

Return : TRUE if successful, False if failed.

Parameter when reaction event is received :

WPARAM : Commnad transfer status flag (ESF_MON_ADC)

LPARAM : Control status flag of the command (False if Time Out Error)

(0x30:Normal operation, 0x39:Abnormal operation, 0x31:Not the Set Mode)

※ When reaction event is received, get data using RHIO_GetSettingData function and check the Level on setting.

eg : `RHIO_SndCmd_GetADC();`

RHIO_SndCmd_SetPortEnable

Description : Used to send the command to enable/disable all the ports.

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_SndCmd_SetPortEnable
                                        (SRHIOSetting sSetData);
```

Parameter :

```
SRHIOSetting sSetData : Data to set
SRHIOSetting.sADC.eModeADC[4] -> Value of the Level/Switch Mode
                                 (EAM_LEVEL, EAM_SWITCH)
SRHIOSetting.sADC.bReference -> Value of the ADC Reference
                           (0x30:Avcc, 0x31:Internal, 0x32:External)
SRHIOSetting.sInput.isEnable[12] -> Input Enable/Disable
                                    (Enable:true, Disable:false)
SRHIOSetting.sOutput.isEnable[10] -> Outnput Enable/Disable
                                    (Enable:true, Disable:false)
```

Return : TRUE if successful, False if failed.

Parameter when reaction event is received :

WPARAM : Command transfer status flag (ESF_SET_ENABLE)

LPARAM : Control status flag of the command (False if Time Out Error)

(0x30:Normal operation, 0x39:Abnormal operation, 0x31:Not the Set Mode)

eg :

```
SRHIOSetting sSetData;
for(int index = 0; index < 4; index++)
   SetData.sADC.eModeADC[index] = EAM_SWITCH;
sSetData.sADC.bReference = 0x30;
for(int index = 0; index < 12; index++)
   SetData.sInput.isEnable[index] = true;
for(int index = 0; index < 10; index++)
   sSetData.sOutput.isEnable[index] = true;
RHIO_SndCmd_SetPortEnable(sSetData);
```

`RHIO_SndCmd_GetPortEnable`

Description : Used to send the command to check whether it is possible to get the Enable/Disable Port data of all the ports

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_SndCmd_GetPortEnable();
```

Parameter : None

Return : TRUE if successful, False if failed.

Parameter when reaction event is received: :

WPARAM : Command transfer status flag (ESF_MON_ENABLE)

LPARAM : Control status flag of the command (False if Time Out Error)

(0x30:Normal operation, 0x39:Abnormal operation, 0x31:Not the Set Mode)

※ When reaction event is received, get data using RHIO_GetSettingData function and check the Enable/Disable Port on setting.

eg : `RHIO_SndCmd_GetPortEnable();`

`RHIO_SndCmd_SetPwrStopEnable`

Description : Used to send the command to enable/disable the Power Failure Recovery of all the input ports

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_SndCmd_SetPwrStopEnable
                                                (SRHIOSetting sSetData);
```

Parameter :

`SRHIOSetting sSetData : Data to set`

`SRHIOSetting.sOutput.sEnablePowerStop[10] : Enable/Disable Power Failure Recovery of the input port (Enable:true, Disable:false)`

Return : TRUE if successful, False if failed.

Parameter when reaction event is received :

WPARAM : Commnad transfer status flag (ESF_SET_PWR_STOP)

LPARAM : Control status flag of the command (False if Time Out Error)

(0x30:Normal operation, 0x39:Abnormal operation, 0x31:Not the Set Mode)

eg :

```
SRHIOSetting sSetData;
for(int index = 0; index < 10; index++)
    sSetData.sOutput.sEnablePowerStop[index] = true;
RHIO_SndCmd_SetPortEnable(sSetData);
```

`RHIO_SndCmd_GetPwrStopEnable`

Description : Used to send the command to check whether it is possible to get the Enable/Disable Power Failure Recovery of all the input ports

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_SndCmd_GetPwrStopEnable();
```

Parameter : None

Return : TRUE if successful, False if failed.

Parameter when reaction event is received :

`WPARAM` : Command transfer status flag (ESF_MON_PWR_STOP)

`LPARAM` : Control status flag of the command (False if Time Out Error)

(0x30:Normal operation, 0x39:Abnormal operation, 0x31:Not the Set Mode)

※ When reaction event is received, get data using RHIO_GetSettingData function and check the Enable/Disable Power Failure Recovery on setting.

eg : `RHIO_SndCmd_GetPortEnable();`

`RHIO_SndCmd_SetFactoryReset`

Description : Used to send the command to request RHIO FactoryReset

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_SndCmd_SetFactoryReset();
```

Parameter : None

Return : TRUE if successful, False if failed.

Parameter when reaction event is received :

WPARAM : Command transfer status flag (ESF_SET_FACTROT_RESET)

LPARAM : Control status flag of the command (False if Time Out Error)

(0x30:Normal operation 0x39:Abnormal operation)

eg : `RHIO_SndCmd_SetFactoryReset();`

`RHIO_SndCmd_SetSerial`

Description : Used to send command to set the RHIO serial number

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_SndCmd_SetSerial
                                        (CString strSerial);
```

Parameter :

`SRHIOSetting sSetData : Data to set`

`CString strSerial : Serial number string limited to 12 bytes`

Return : TRUE if successful, False if failed.

Parameter when reaction event is received :

WPARAM : Commnad transfer status flag (ESF_SET_SERIAL)

LPARAM : Control status flag of the command (False if Time Out Error)

(0x30:Normal operation, 0x39:Abnormal operation, 0x31:Not the Set Mode)

eg : `RHIO_SndCmd_SetSerial("0000003");`

`RHIO_SndCmd_GetSerial`

Description : Used to send the command to check whether it is possible to get the RHIO Serial number

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_SndCmd_GetSerial();
```

Parameter : None

Return : TRUE if successful, False if failed.

Parameter when reaction event is received :

WPARAM : Command transfer status flag (ESF_MON_SERIAL)

LPARAM : Control status flag of the command (False if Time Out Error)

(0x30:Normal operation, 0x39:Abnormal operation, 0x31:Not the Set Mode)

※ When reaction event is received, get data using RHIO_GetSettingData function.

eg : `RHIO_SndCmd_GetSerial();`

`RHIO_SndCmd_GetFirmware`

Description : Used to send the command to check whether it is possible to get the RHIO Firmware Version

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_SndCmd_GetFirmware();
```

Parameter : None

Return : TRUE if successful, False if failed.

Parameter when reaction event is received :

WPARAM : Command transfer status flag (ESF_MON_FIRMWARE)

LPARAM : Control status flag (False if Time Out)

(0x30:Normal operation, 0x39:Abnormal operation, 0x31:Not the Set Mode)

※ When reaction event is received, get data using RHIO_GetSettingData function.

eg : `RHIO_SndCmd_GetFirmware();`

`RHIO_GetSettingData`

Description : Used to send the command to get the setting data corresponding to the command transfer status flag

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_GetSettingData
                                        (SRHIOSetting &sSetData);
```

Parameter :

```
SRHIOSetting &sSetData : Data to get
```

Return : TRUE if successful, False if failed.

eg :

```
    SRHIOSetting sSetData;
    RHIO_GetSettingData(sSetData); // Setting data stored to sSetData
```

```
RHIO_GetOnOffData
```

Description : Used to send the command to get the ON/OFF status of all the ports

Function Prototype :

```
extern "C" __declspec(dllexport) bool RHIO_GetOnOffData
                                        (SOnOffStatusData &sOnOffData);
```

Parameter :

```
SOnOffStatusData &sOnOffData : Data to get
```

Return : TRUE if successful, False if failed.

eg :

```
    SOnOffStatusData sOnOffData;
    RHIO_GetOnOffData(sOnOffData);
        //ON/OFF status stored to sOnOffData
```