Unit 4, 12/F
Vanta Industrial Centre.
21-33 Tai Lin Pai Road
Kwai Chung, N.T. Hong Kong SAR

Regor
Host Library SDK
Getting Started Guide for .NET Developers

Disclaimer

Star Systems International and the **Star Systems International logo** are trademarks of **Star Systems International Ltd**. in Hong Kong and other countries.

Microsoft, **Windows**, the Windows logo are trademarks of Microsoft Corporation in the U.S. and other countries. All other products names mentioned herein may be trademarks of their respective companies.

Star Systems International Ltd. shall not be liable for technical or editorial errors or omissions contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material. The information in this document is provided “as is” without warranty of any kind - including but not limited to, the implied warranties of merchantability and fitness for a particular purpose, and is subject to change without notice. The warranties for Star Systems International products are set forth in the express limited warranty statements accompanying such products. Nothing herein should be construed as constituting an additional warranty.

This product is not designed, intended, authorized or warranted to be suitable for life support applications or any other life critical applications which could involve potential risk of death, personal injury, property damage, or environmental damage.

Regor Host Library SDK API Reference Manual

Version 13.12.24

Contents

Introduction

This document is to describe reader control API required for application program developer or system integrator desiring to apply IDRO “RFID reader”.

The description of **Host Library SDK** in this document is implanted using C# Language of Visual Studio .NET.

Therefore, this can be used for all the developers using Visual Basic .NET, C++ with managed extension with .NET Development Environment.

Please refer to the separted document for “Packet size and

Refer to the additional documents for **“Packet standards or specific command description”**

Able to use the module or fixed type readers with the same **Host Library SDK**

Type	Model Name	TCP/IP	Serial
Module	IDRO 900MA	X	O
	IDRO 900EA (Embedded Antenna)	X	O
Fixed	IDRO 900F / 900F4 / 900V	O	O

Therefore, it can be used for modules and fixed type readers with same **Host Library SDK.**

Reference documents

Document	Document No.
Reader Control Protocol Specifications Guide V#.#.# - Model Name	
API Reference Manual V#.#.#	

Getting Started

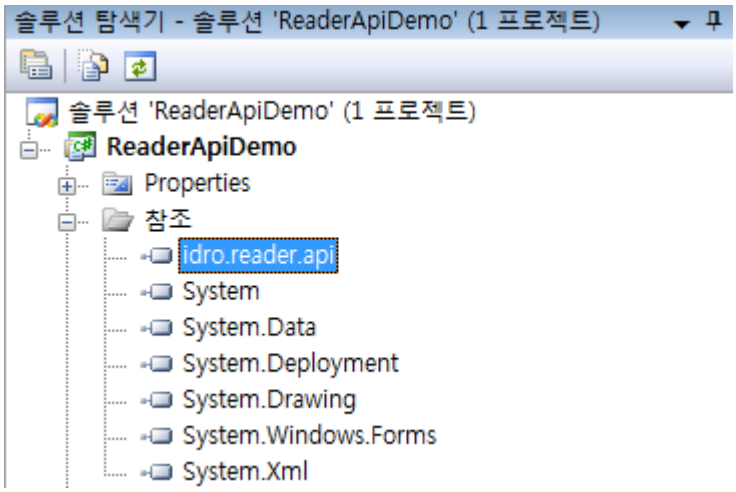
Reader control and example program which are convenient for application program developers or system integrators desiring to apply IDRO “RFID Reader” are provided. This chapter describes centering on example programs provided for how to use reader control. Provided example programs are described in C# language in development environment of visual studio.NET and they can be quickly applied to following sequence.

- Addition of reader control reference
- Creation and initialization of reader control
- To connect reader
- Disconnection and end of reader connection
- Preparation of reader event handler
- Run Inventory
- Stop Inventory
- To change reader properties

To add reader control references

It **adds references** first to project in order to use provided reader control.

If you add references by selecting provided “idro.reader.api.dll”, reference items are added to project as follows.



And following using directive are added in order to conveniently access to reader control in code.

```
using idro.reader.api;
```

Provided reader control as NON-Visual Control – is not used by registering in Toolbox.

Creation and initialization of reader control

It creates and initializes reader control as below.

```
Reader reader = new Reader();
```

It adds handler in order to handle events generated in reader.

```
reader.ReaderEvent += new ReaderEventHandler(OnReaderEvent);
```

If you want to change reader attributes, you can designate as follows.
If it is not separately set, then default value will be applied..

```
reader.ModelType = ModelType.IDRO900MA;
reader.ConnectType = ConnectType.Serial;
reader.TagType = TagType.ISO18000_6C_GEN2;
```

To connect reader

IP address, port number, serial communication speed of RFID reader of IDRO are set as shown below at the time of release of the product. It can start to connect with reader according to connection type as follows.

```
if (ConnectType == ConnectType.Tcp)
     reader.Open("192.168.9.6", 5578);
else
     reader.Open("COM3", 115200);
```

Type	Model Name	Ethernet	Serail
Fixed Type	IDRO 900F / 900F4 / 900V	O	O
Module Type	IDRO 900MA	X	O
	IDRO 900EA	X	O

Open(…) function starts to connect asynchronously, if connection is completed then following event occurs..

`EventType.Connected`:
If connection is successful, then it occurs
`EventType.Disconnected`:
If connection is failed then it occurs and failed reason is contained in the Message of ReaderEventArgs class of event handler parameters.
If required IP address, etc. can be changed using separately provided Reader@ExpressTM program.

Disconnection and end of reader connection

Following function is called in order to disconnect the connection with reader and to reclaim required resources.

```
reader.Close(CloseType.Close);
```

If above function is executed then the connection with reader is disconnected and separate processing thread ends and following event occurs.

`EventType.Disconnected`:

In case if user closes window under the condition of connection with reader then correct handling is as follows.

```
//In case if user closes window.
//After reader thread is normally ended then it allows to close window.
//If connection is disconnected, reader thread generates event and it ends.
//Reader event handler can handle to close window.
protected override void OnClosing(CancelEventArgs e) protected override
void OnClosing(CancelEventArgs e)
{
            //Is reader thread operating?
//Then first it ends reader thread.
if (reader.IsHandling)
{
e.Cancel = true;
reader.Close(CloseType.FormClose);
}
//Is it trying to connect?
//If it is trying to connect then it does not allow to close window.
//Thread's pool thread requires the target to inform result.
else if (reader.IsConnecting)
{
MessageBeep(0);
e.Cancel = true;
}
else
{
base.OnClosing(e);
}
}

//It handles event generated in reader.
//Please note that the event occurred in separate thread.
private void OnReaderEvent(object sender, ReaderEventArgs e)
{
…
switch (e.Type)
{
case EventType.Disconnected:
//State message is displayed.
StateMessage = e.Message;
//If it occurred through window close then it closes //window.
if (e.CloseType == CloseType.FormClose) Close();
break;
}
}
```

Preparation of reader event handler

It prepares handler to handle events generated in reader. It is function to be called in the thread created separately internally therefore it handles for main thread to handle UI. It prepares in following structure.

```
//It handles event occurred in reader.
//Note that event is generated in separate thread.
Private void OnReaderEvent(object sender, ReaderEventArgs e)
{
      if (this.InvokeRequired)
      {
            this.BeginInvoke(new ReaderEventHandler(OnReaderEvent), new object[]
{ sender, e });
            return;
      }

      switch (e.Type)
      {
            case EventType.Connected:
            case EventType.Disconnected:
            case EventType.Timeout:
      ////////////////////////////////////////////////////////////////////////
      // BASIC OPERATIONS EVENTS
      ////////////////////////////////////////////////////////////////////////
            case EventType.Inventory:
            case EventType.ReadMemory:
            case EventType.WriteMemory:
            case EventType.Lock:
            case EventType.Kill:
      ////////////////////////////////////////////////////////////////////////
      // CONFIGURATIONS EVENTS
      ////////////////////////////////////////////////////////////////////////
            case EventType.Power:
            case EventType.Version:
            case EventType.AccessPwd:
            case EventType.GlobalBand:

            case EventType.Buzzer:
            case EventType.ContinueMode:

            case EventType.Port:
            case EventType.Selection:
            case EventType.Filtering:
            case EventType.Algorithm:
            case EventType.Gpio:
            case EventType.TcpIp:

            case EventType.Command:
                  break;
      }
}
```

Run Inventory

It calls following function in order to read tags.

```
reader.InventoryMultiple();
```

If tags are recognized, `EventType.Inventory` event occurs, it can handle as follows in order to parse each tag ID.

```
- Hereafter it is selected from function of OnReaderEvent() –

switch (e.Type)
{
```

```
…
case EventType.Inventory:

//Received data are in the byte array (e.payload), it decodes into string using
//following function.
string szPayload = Encoding.ASCII.GetString(e.Payload);

//Many responses can be contained in the generated event and it separates first.
//Received data may contain tag ID, set value, response code, etc. according to call
//function, byte array(e.payload) may contain more than 1 response therefore it
//separates using following separator.
////////////////////////////////////////////////////////////////////////////////
// Tag Memory : [#]T300011112222333344445555666[##]
// Response Code : [#]C##
// Settings Values : p0, c1, ...
////////////////////////////////////////////////////////////////////////////////
string[] szResponses = szPayload.Split(new string[] { "\r\n>" },
StringSplitOptions.RemoveEmptyEntries);

char code;
string szValue;
string rsValue;
bool bFixedType = reader.IsFixedType();
int nPos = bFixedType ? 1 : 0; //Port Number is excluded.

//It handles for each response
foreach (string szResponse in szResponses)
{
      code = szResponse[nPos];
      switch (code)
      {
             case 'T'://Tag Memory
                    szValue = szResponse.Substring(nPos + 1, szResponse.Length -
(nPos + 1 + (bFixedType ? 2 : 0)));//exclude [#]T/C, CheckSum
                    lbxResponses.Items.Insert(0, szValue);
                    break;
             case 'C'://Response Code
                    szValue = szResponse.Substring(nPos + 1, szResponse.Length -
(nPos + 1));//exclude [#]T/C
                    szValue = szValue + "-" + reader.Responses(szValue);
                    lbxResponses.Items.Insert(0, szValue);
                    break;
                    case 'R'://RSSI
                    rsValue = szResponse.Substring(nPos + 1, szResponse.Length -
(nPos + 1));//exclude [#]
                    int changeover = Convert.ToInt16(rsValue, 16);
                    double rsResult = (double) changeover / 10;
                    lbxResponses.Items.Insert(0, rsResult);
                    break;
default://Settings Values
                    lbxResponses.Items.Insert(0, szResponse);
break;
      }
}
break;
      …
}
```

It provides following inventory types.

Inventory Type	Function Name	Fixed Type	Module Type
Multiple	InventoryMultiple()	O	O
Single	InventorySingle()	O	O
Selection	InventorySelection()	O	O
Filtering	InventoryFiltering()	O	O

Stop Inventory

It calls following function and stops to conduct inventory.

```
reader.StopOperation();
```

Separate reader event does not occur.

Trigger inventory by GPIO

Inventory mode can be trigger by GPIO. To trigger the

To change reader attributes

Following functions can be used to get and set reader attributes. Each function is described in separated document.

Function	Cmd szType	Fixed Type	Module Type
GENERAL			
void GetOperationMode() void SetOperationMode(int nValue)	x	O	O
void GetPower() void SetPower(int nValue)	p	O	O
void GetFwVersion()	v	O	O
void GetAccessPwd() void SetAccessPwd(string szPassword)	w	O	O
void GetGlobalBand() void SetGlobalBand(int nValue)	f	O	O
OPTIONS			
void GetBuzzer() void SetBuzzer(bool bState)	b	O	△
void GetContinueMode() void SetContinueMode(bool bContinue)	c	O	O
void GetReportingType() void SetReportingType(ReportingType type)	i	O	O
PORT			
void GetPortActive() void SetPortActive(int nValue)	e	O	X
void GetPortPower(int nPort)//1,2,3,4 void SetPortPower(int nPort, int nValue)	p1~4	O	X
void GetPortInventoryCount() void SetPortInventoryCount(int nValue)	k	O	X
void GetPortInventoryTime() void SetPortInventoryTime(int nValue)	j	O	O
void GetPortIdleTime() void SetPortIdleTime(int nValue)	0	O	O

SELECTION			
void GetSelectionBank() void SetSelectionBank(MemoryType nType)	9	O	O
void GetSelectionOffset() void SetSelectionOffset(int nOffset)	;	O	O
void GetSelectionAction() void SetSelectionAction(SelectionActionType nType)	8	O	O
void GetSelectionSession() void SetSelectionSession(int nValue)	s	O	O
ALGORITHM			
void GetAlgorithmParameter(Algorithm a, int nType) void SetAlgorithmParameter(Algorithm a, int nType, int nValue)	0 q 1 [2]	O	△
CONNECTION			
void GetConnectionSetting(ConnectionSetting nType) void SetConnectionSetting(ConnectionSetting nType, string szValue)	r, r1,r2 2 7 1	O	△
G P I O			
void GetInputDetect() void SetInputDetect(int nValue)	-	O	X
void GetOutputLevel() **void SetOutputLevel(string szValue)**	**g 0** **g 1**	**O**	**O**
void GetFlashOutputLevel() void SetFlashOutputLevel(int nValue)	5	O	X
void GetFlashDuration() void SetFlashDuration(int nValue)	4	O	X
V L C			
void GetVisibleCapability() void SetVisibleCapability(int nValue)	V	O	X
void GetVisibleLight() void SetVisibleLight(bool blight)	W	O	X
ENGINEER MODE			
void GetEngineerPwd() void SetEngineerPwd(string szPassword)	m	O	X
void GetEngineerMode() void SetEngineerMode (bool bEnginerr,string szPassword)	E	O	X
void GetChHoppingSWT() void SetChHoppingSWT(bool bState)	o	O	X
void GetChNumber() void SetChNumber(string szNumber)	n	O	X
void GetLinkProfile() void SetLinkProfile(int nValue)	L	O	X
void GetReaderMode() void SetReaderMode(int nValue)	x	O	X

Reader API Specifications

Please refer to separated document.

Hardware Products Warranty Statement

WARRANTY.

All Hardware Products sold by STAR Systems International Limited (SSI) are warranted against defects in material and workmanship under normal use and service for one (1) year from the original date of purchase (the "Warranty"). Any Extended Warranties must be documented on the original invoice as a separate line item. For defects covered by this Warranty, SSI will repair the defect or replace the product, at its sole option and return the product to you.

EXCLUSIONS.

If the defect was caused by any of the following, the Warranty shall not apply and an estimate for repair or replacement will be submitted for your approval prior to work being performed: abuse, mishandling, acts of God, vandalism, accident, electrostatic discharge damage, failure to follow installation or operating instructions, failure to provide a suitable environment, unauthorized modification of the product modification of the printed circuit board by parties other than SSI, and damage that is caused during shipping for warranty service and any product that is returned with the security seal broken.

RMA PROCEDURE.

For Warranty service, the Customer must comply with STAR Systems International Return Materials Authorization ("RMA") policy, which is published on the STAR Systems International website at www.starint.net, and may be updated from time to time. Prior to shipping a product to STAR Systems International for warranty inspection, replacement or repair, an RMA number must be obtained from STAR Systems International's RMA department at +852 3691 9925 or by email at support@star-int.net. RMA forms can be downloaded from the STAR Systems International website or the Customer can receive the form by fax (+852 37474065) or email by contacting the RMA department. One RMA form must be used for each RMA submission and the product should be shipped to the address below. For products covered by this Warranty, the Customers are responsible for payment of shipping costs to the STAR Systems International repair center and STAR Systems International will be responsible for the cost of returning the item. The standard return shipment is "Speed Post". Any other desired "expedited" or overnight shipping costs for warranty repairs will be the customer's responsibility.

DISCLAIMER OF WARRANTIES.

OTHER THAN SET FORTH ABOVE, SSI HEREBY DISCLAIMS ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING WITHOUT LIMITATION, THE WARRANTIES OF Equipment Warranty (Rev 2-2017) MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.

LIMITATION OF LIABILITY.

IN NO EVENT WILL SSI BE LIABLE FOR ANY CONSEQUENTIAL, INDIRECT, EXEMPLARY, SPECIAL, OR PUNITIVE DAMAGES, WHETHER ARISING OUT OF CONTRACT, TORT, NEGLIGENCE, STRICT LIABILITY OR OTHERWISE. IN NO EVENT WILL STAR SYSTEMS INTERNATIONAL'S TOTAL CUMULATIVE, AGGREGATE LIABILITY, WHETHER ARISING OUT OF CONTRACT, TORT, NEGLIGENCE, STRICT LIABILITY, OR OTHERWISE, EXCEED THE PRICE ACTUALLY PAID BY THE CUSTOMER FOR THE PRODUCT FROM WHICH THE CLAIM ARISES.

This warranty gives the Customer specific legal rights, and the Customer may also have other rights that may vary from local jurisdiction. If the Customer has questions concerning the product or warranty, contact the dealer from which it was purchased. The Customer may also contact STAR Systems International at the following address and ask for warranty assistance.

Appendix

This section provides you with safety information, technical support information, and sources for additional product information.

Safety Information

Your safety is extremely important. Read and follow all warnings and cautions in this document before handling and operating RFID equipment. You can be seriously injured, and equipment and data can be damaged if you do not follow the safety warnings and cautions.

A caution alerts you to an operating procedure, practice, condition, or statement that must be strictly observed to prevent equipment damage or destruction, or corruption or loss of data.

Note: Notes either provide extra information about a topic or contain special instructions for handling a particular condition or set of circumstances.

Global Services and Support

Web Support

Visit the SSI website at www.star-int.net to download our current manuals (in PDF).

Visit the Star Systems University at www.star-int.net and click **Tech Support > Star Systems University** to review technical information or to request technical support for your RFID product.

Send Feedback

Your feedback is crucial to the continual improvement of our documentation. To provide feedback about this manual, please visit the **Contact Us** at www.star-int.net.

Telephone Support

In Hong Kong, Call **+852-3691-9925**. In the U.S.A., call +**1-888-457-7755**.

Outside Hong Kong and the U.S.A., contact your local SSI representative. To search for your local representative, from SSI website, click **Contact Us** at www.star-int.net.

Related Documents

The SSI website at www.star-int.net contains our documents (as .pdf files) that you can download for free.

To download documents

1 Visit the SSI website at www.star-int.net.

2 Click the **Tech Support > Download**.

3 According to your product category, choose **Readers / Antennas / Tag Labels**.