Quick start guide

iX serial ReadWrite - iX script module

SER0043 - Reading and Writing serial data

Beijer ELECTRONICS

1 Function and area of use

This example script shows how Tags (or any other information) can be sent serially as a formatted string. For receiving serial data it is important either to know the exact structure of a received string or to have unique separators as delimiters in order to split the received string and extract the important data.

This example is implemented to be tested with two iX devices or iX Runtimes.

2 About this document

This quick start document should not be considered as a complete manual. It is an aid to be able to startup a normal application quickly and easily.

Use the following hardware, software, drivers and utilities in order to obtain a stable application:

In this document we have used following software and hardware

- iX Developer 2.40 SP4 / SP5
- X2 series, C2 series and iX PC RT (iX Runtime)

For further information refer to

- iX Developer Reference Manual (MAxx831)
- iX Developer User's Guide (MAxx832)
- Beijer Electronics knowledge database, HelpOnline

This document and other quick start documents can be obtained from our homepage. Please use the address *support.europe@beijerelectronics.com* for feedback.

3 Table of Contents

4 Sending and Receiving serial data

This example should only be a starting point or orientation aid when it comes to sending and/or receiving serial data.

This example shows how two iX devices exchange Tags utilizing serial communication.

Surely very often only writing serial data – e.g. to a serial printer or receiving data – e.g. from a card-reader or any other automation device is needed.

4.1 The script module SCM_Serial_COM

At the beginning of the script module a Tag Array is specified.

These are the Tags that are formatted and sent serially later on.

```
//define the Tags to be sent/received
GlobalDataItem[] myTags = new GlobalDataItem[]
    {
    Globals.Tags.Tag_Int16,
    Globals.Tags.Tag_Int32,
    Globals.Tags.Tag_Float,
    Globals.Tags.Tag_Double,
    Globals.Tags.Tag_String,
    Globals.Tags.Tag_Bool,
    Globals.Tags.Tag_UInt16,
    Globals.Tags.Tag_UInt32,
    Globals.Tags.Tag_BIT
    };
```

4.2 The “SCM_Serial_COM_Created” method

The created method of a script module is worked on exactly once, thus it is the ideal place to initialize objects like strings, timers and other stuff.
In this method also the COM-Port is opened.

4.3 The “Write_COM” method

At the beginning of this method all Tags are read in a loop in order to ensure that the latest values of these Tags are sent.
The Read( ) method is a synchronous read – this means that the script is worked on after all Tags are read.

```
foreach (GlobalDataItem tag in myTags){
        tag.Read();
}
```

The formatting of the string is also done in a loop. Every Tag information consists of a leading “**@**”, then the Tagname followed by a “**:**” and the TagValue.

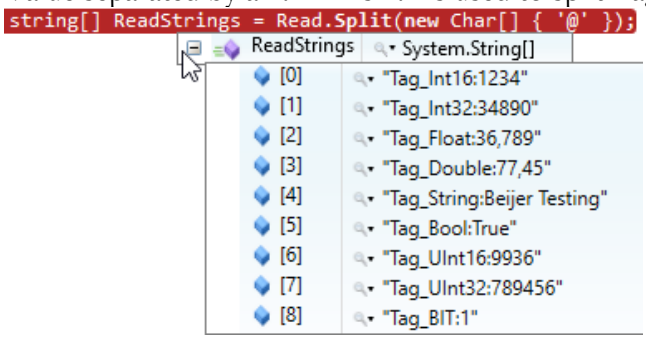
```
StringBuilder FormattedString = new StringBuilder();
foreach (GlobalDataItem tag in myTags){
        FormattedString.AppendFormat("@" + tag.DisplayName + ":" + tag.Value.ToString());
}
```

4.4 The “Read_COM” method

This method is called cyclic by the timer that was initialized in the Created method of the script module. Here the extracting of serial data is done. If there are data inside the buffer of the serial port, they are read out and a basic validity check is done.

In this case the script checks whether the first received character is a ”@”. If not the data are ignored, if yes the first “@” is removed as the other “@” characters are used for splitting the received string in Tag information.

The result of a string.Split is a string Array – each string contains the Tag information Name and Value separated by a “:”. This “:” is used to split Tagname and Value later on.



After splitting the received string into an string array, the assignment to iX Tags is done.

Therefore two nested loops are used to loop through all the Tags and the string array.
Inside the second loop is checked whether the respective string contains the proper Tagname and if yes the proper conversion for the TagValue is done.

4.5 Adjusting COM-Port settings

The COM-Port settings can be adjusted in line 70 of the script module.

```
69      // adjust your COM port settings here:
70      OpenPort("COM1", 115200,Parity.None,StopBits.One,8);
```

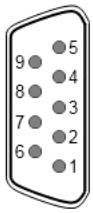
4.6 Adjust reading cycle

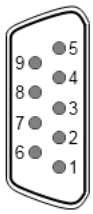
The setting for the cyclic reading can be configured in 59 of the script module.

```
58      // configuration of the reading cycle
59      int iTimerCycle = 500; //ms
```

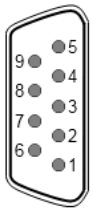
4.7 COM-Port Pinning

TxA/B/C QTERM X2 Base

Female DB-9	Pin	COM1 signal	COM2 signal
	1	-	RS422 TX+ / RS485 TX+/RX+
	2	RS232 RX	-
	3	RS232 TX	-
	4	-	RS422 RX+
	5	GND	GND
	6	-	RS422 TX- / RS485 TX-/RX-
	7	RS232 RTS	-
	8	RS232 CTS	-
	9	-	RS422 RX-

Female DB-9	Pin	COM3 signal	COM4 signal
	1	-	RS422 TX+ / RS485 TX+/RX+
	2	RS232 RX	-
	3	RS232 TX	-
	4	-	RS422 RX+
	5	GND	GND
	6	-	RS422 TX- / RS485 TX-/RX-
	7	-	RS422 RTS+
	8	-	RS422 RTS-
	9	-	RS422 RX-

X2 Pro X2 Control X2 Motion X2 Marine

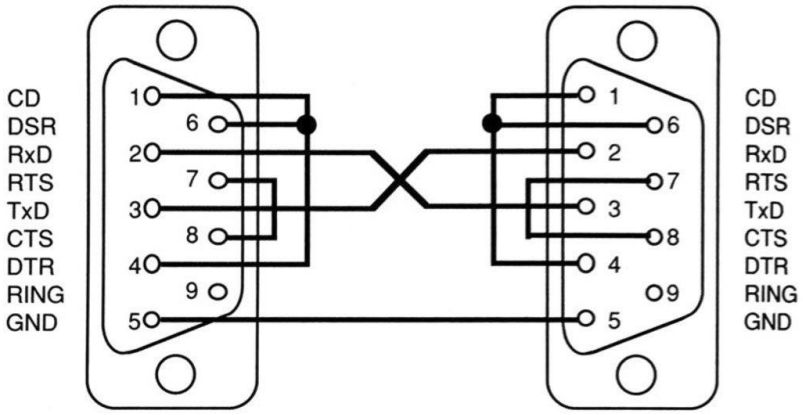
Female DB-9	Pin	COM1 signal	COM2 signal	COM3 signal
	1	-	RS422 TX+/RS485 TX+/RX+	-
	2	RS232 RX	-	-
	3	RS232 TX	-	-
	4	-	RS422 RX+	RS485 TX+/RX+
	5	GND	GND	GND
	6	-	RS422 TX- / RS485 TX-/RX-	-
	7	RS232 RTS	-	-
	8	RS232 CTS	-	-
	9	-	RS422 RX-	RS485 TX-/RX-

4.8 Null modem cable

For a serial point-to-point connection typically a null modem cable is used.

The most important thing is that the cable crosses RxD and TxD – so the one side is sending via TxD, the other side is receiving data on RxD and vice versa.

In the below picture you see that RTS/CTS and DSR/DTR are brigged on both sides. It depends on the used devices if they need these information from the other side or not.
For a simply serial connection only RxD/TxD and GND are necessary.



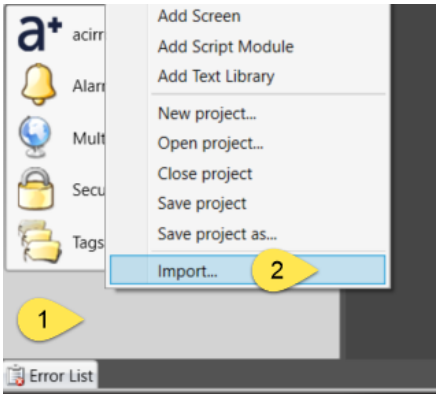
5 Adding the serial Reading/Writing Functionality

Implementation

1. Import the script module (SCM_Serial_COM) to your present iX application.
   - Open your iX application (or make a working copy) and import.
2. Optionally Import "Screen1", see example project (iX_Serial_ReadWrite_V1_0_0).
3. Adjust the Tags that should be read / written inside the script module.
4. Adjust the formatting / extraction inside the Write_COM and Read_COM methods.
5. Run the application.

5.1 Import the project parts

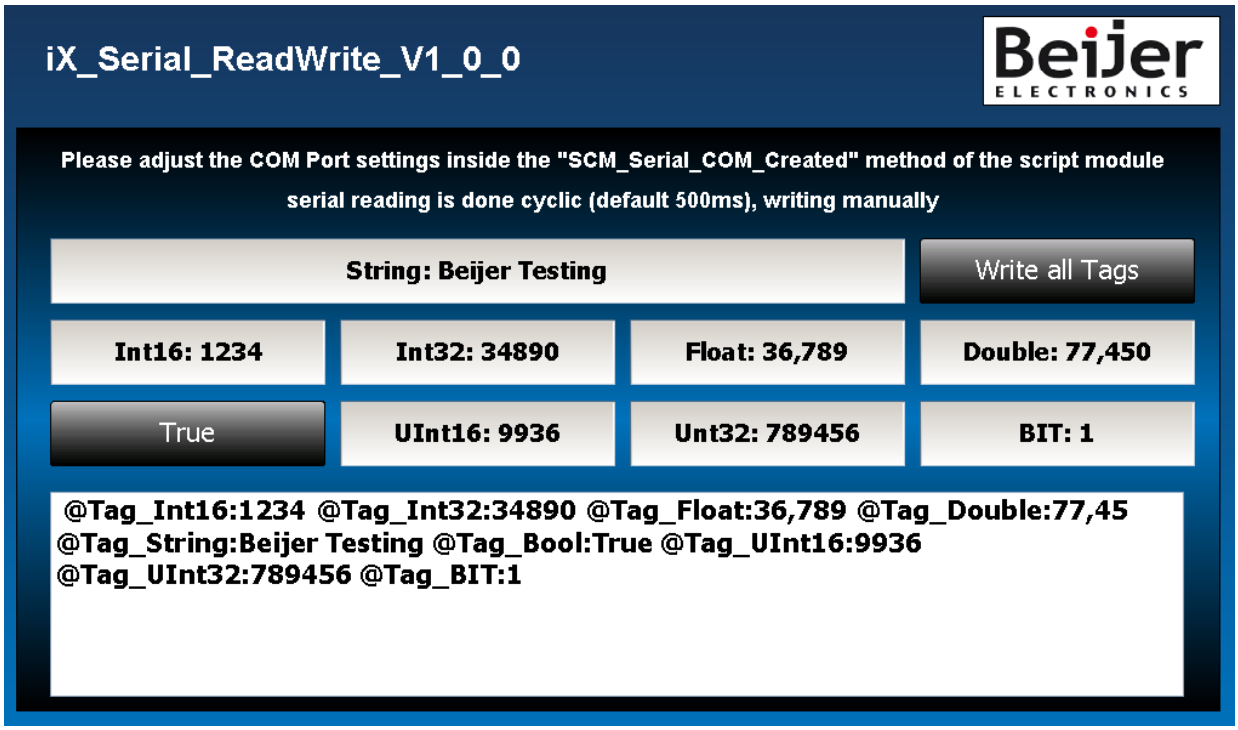
Follow the steps to add the enclosed screens and the script module to your iX project:



1. Unpack the enclosed example ZIP-file to a temporary folder.
2. Start iX Developer and open your project.
3. In the Project Explorer, right-click in the lower left corner (1. in the picture)
4. In the list, select Import... (2. in the picture)
5. Navigate to the temporary folder, where you unpacked the ZIP-file and select SCM_Serial_COM.neo, click [Open].
6. Optionally select Screen1.neoxaml, click [Open].
7. Answer Yes to the following questions.

6 Operation

6.1 Write and Receive Tags



In Runtime:

The specified serial port is checked cyclic in the interval specified in the script module (see 4.6)

If unchanged the serial buffer is read every 500ms and the received string is evaluated and the Tag values are assigned.

To send the actual Tag data, change the Tag values and press the "Write all Tags" button.
In the TextBox on the lower half you see the string that is send serially.

7 About Beijer Electronics

Beijer Electronics is a multinational, cross-industry innovator that connects people and technologies to optimize processes for business-critical applications. Our offer includes operator communication, automation solutions, digitalization, display solutions and support. As experts in user-friendly software, hardware

and services for the Industrial Internet of Things, we empower you to meet your challenges through leading-edge solutions.

Beijer Electronics is a Beijer Group company. Beijer Group has a sale over 1.6 billion SEK in 2019 and is listed on the NASDAQ OMX Nordic Stockholm Small Cap list under the ticker BELE. **www.beijergroup.com**

China	**NORWAY**	**TAIWAN**
Shanghai	Drammen	Taipei
DENMARK	**SOUTH KOREA**	**TURKEY**
Roskilde	Seoul	Istanbul
FRANCE	**SWEDEN**	**UNITED KINGDOM**
Paris	Göteborg Malmö Stockholm	Nottingham
GERMANY		**USA**
Nürtingen		Salt Lake City

7.1 Contact us

Global offices and distributors