TCP Client with ESP and DIA

Digi Technical Support

21 June 2016

Contents

1 INTRODUCTION

1.1 Assumptions

This example will assume that the ConnectPort X4 is using factory default settings and ZB peripherals are already connected.

1.2 Software

Digi ESP can be downloaded from: http://ftp1.digi.com/support/driver/40002839_E.exe

This example uses putty (any other terminal application works) and netcat as a TCP server.

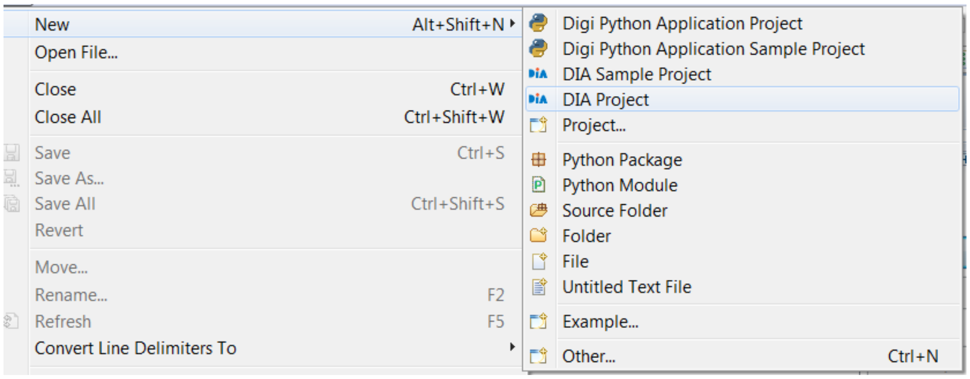
1.3 Overview

This example will use a ZB RS232 Adapter and transmit channel data in CSV format using a simple TCP client. The last part of the example will also show how to push this data to Device Cloud.

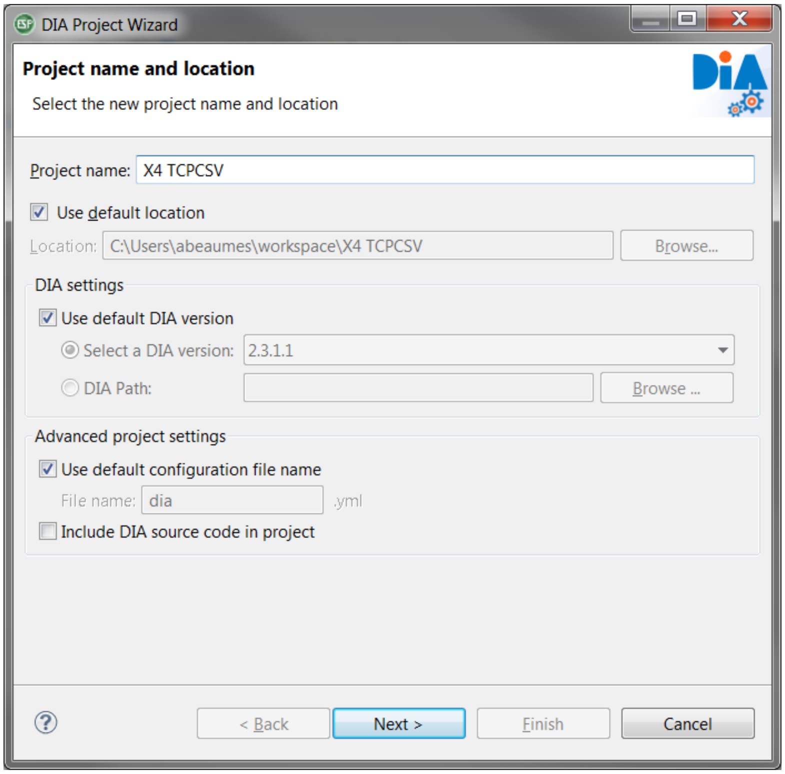
2 DIGI ESP CONFIGURATION

2.1 Project creation

Start Digi ESP and under File > New select **DIA Project**



Chose a Project Name and click **Next**



Project name and location

Select the new project name and location

Project name: X4 TCPCSV

☑ Use default location

Location: C:\Users\abeaumes\workspace\X4 TCPCSV

DIA settings

☑ Use default DIA version

● Select a DIA version: 2.3.1.1

○ DIA Path:

Advanced project settings

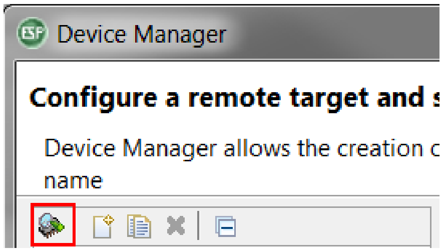
☑ Use default configuration file name

File name: dia .yml

☐ Include DIA source code in project

If this is the first start of Digi ESP, click **New** to create a new Remote Device.

Click on the Digi Device Discovery button



Device Manager

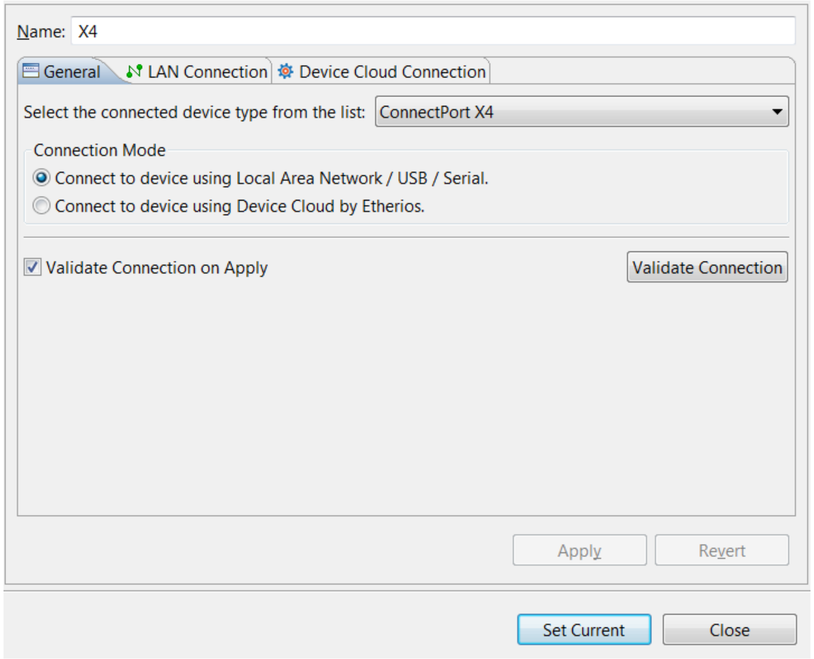
Configure a remote target and s

Device Manager allows the creation c name

Check **Local Arena Network** and click **OK**

Select the X4 in the device list and click on **Create Configuration**

Click on **Set Current**



Name: X4

Select the connected device type from the list: ConnectPort X4

Connection Mode

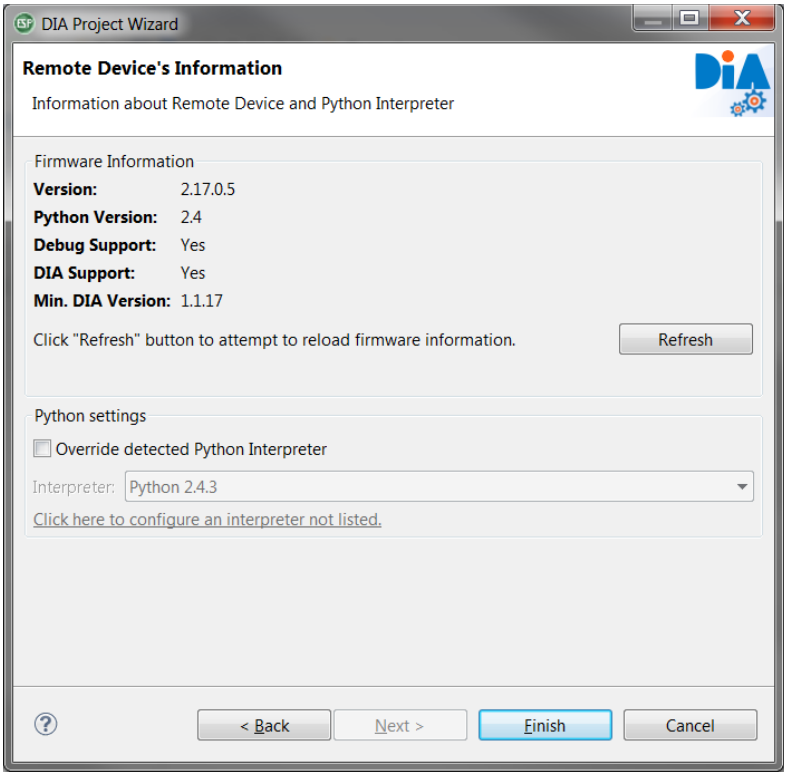
● Connect to device using Local Area Network / USB / Serial.

○ Connect to device using Device Cloud by Etherios.

☑ Validate Connection on Apply

Click **Next**

The project is now ready to be created. Click **Finish**



Remote Device's Information

Information about Remote Device and Python Interpreter

Firmware Information

Version: 2.17.0.5
Python Version: 2.4
Debug Support: Yes
DIA Support: Yes
Min. DIA Version: 1.1.17

Click "Refresh" button to attempt to reload firmware information.

Python settings

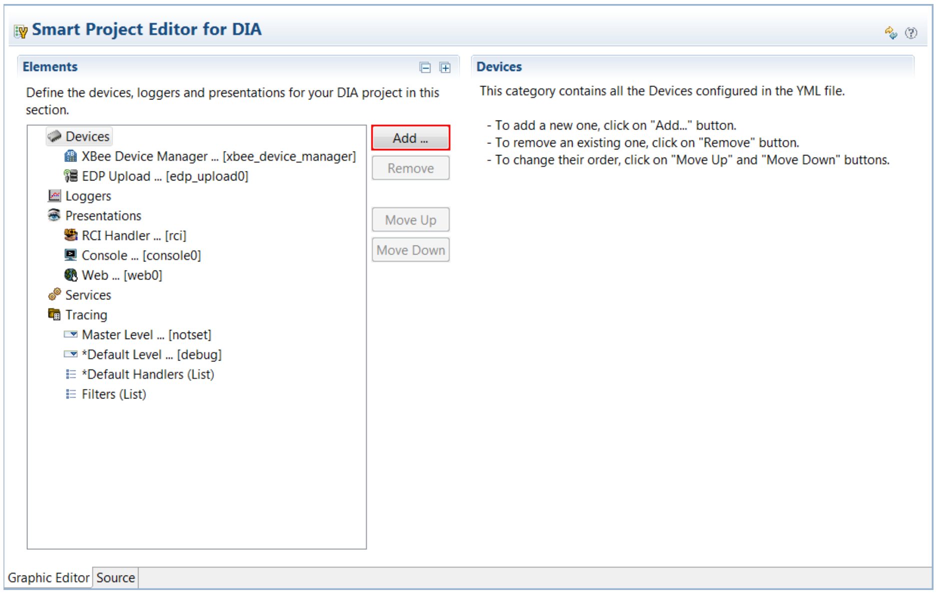
□ Override detected Python Interpreter

Interpreter: Python 2.4.3

Click here to configure an interpreter not listed.

2.2 Add the device(s) to the project

Click on the **Add** button and select the desired device. Repeat for each devices.



Define the devices, loggers and presentations for your DIA project in this section.

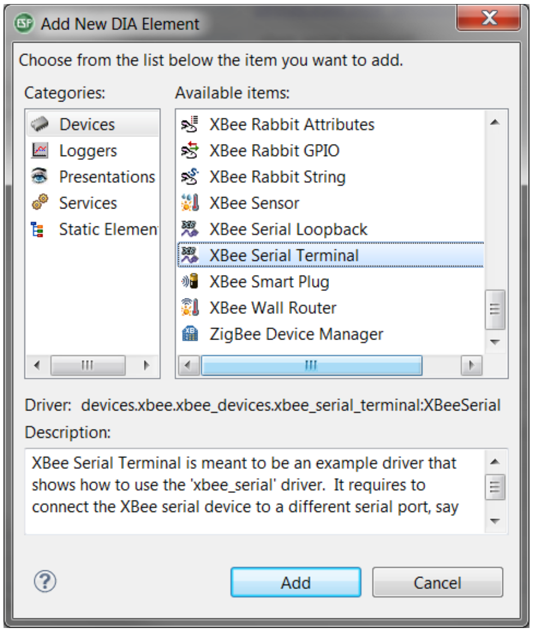
- Devices
  - XBee Device Manager ... [xbee_device_manager]
  - EDP Upload ... [edp_upload0]
- Loggers
- Presentations
  - RCI Handler ... [rci]
  - Console ... [console0]
  - Web ... [web0]
- Services
- Tracing
  - Master Level ... [notset]
  - *Default Level ... [debug]
  - *Default Handlers (List)
  - Filters (List)

Devices

This category contains all the Devices configured in the YML file.

- To add a new one, click on "Add..." button.
- To remove an existing one, click on "Remove" button.
- To change their order, click on "Move Up" and "Move Down" buttons.

In this example, the XBee Serial Terminal is chose for the XBee RS232 Adapter. Click **Add**



Add New DIA Element

Choose from the list below the item you want to add.

Categories:

- Devices
- Loggers
- Presentations
- Services
- Static Elemen

Available items:

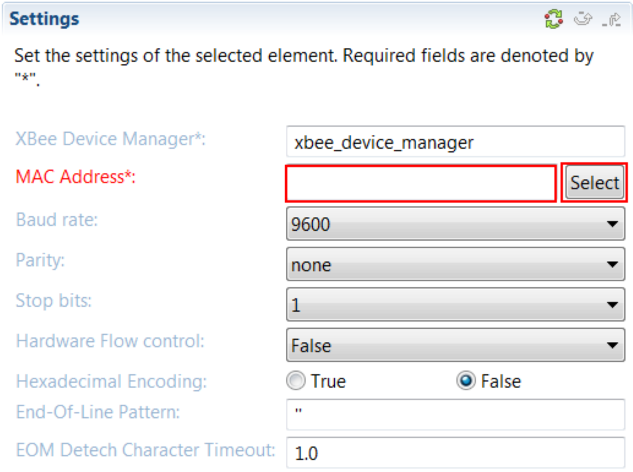
- XBee Rabbit Attributes
- XBee Rabbit GPIO
- XBee Rabbit String
- XBee Sensor
- XBee Serial Loopback
- XBee Serial Terminal
- XBee Smart Plug
- XBee Wall Router
- ZigBee Device Manager

Driver: devices.xbee.xbee_devices.xbee_serial_terminal:XBeeSerial

Description:

XBee Serial Terminal is meant to be an example driver that shows how to use the 'xbee_serial' driver. It requires to connect the XBee serial device to a different serial port, say

For each devices, it is necessary to enter the MAC Address corresponding. Click on the **Select** button to choose the corresponding device:

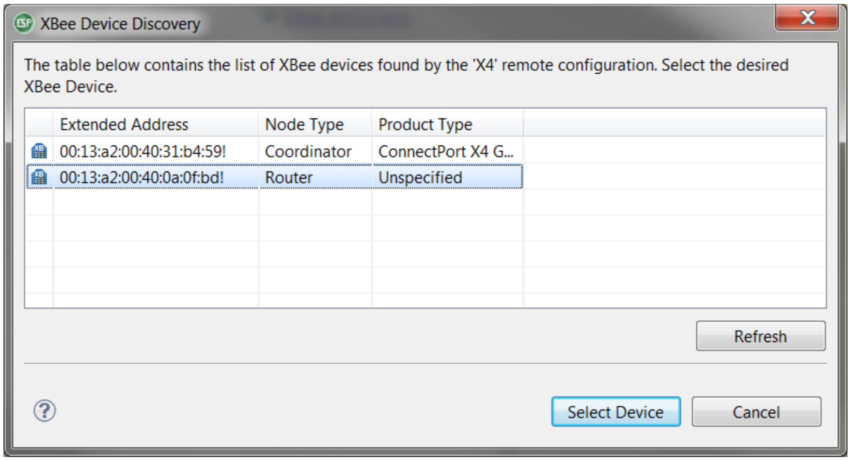


Settings

Set the settings of the selected element. Required fields are denoted by "*".

XBee Device Manager*:	xbee_device_manager
MAC Address*:	Select
Baud rate:	9600
Parity:	none
Stop bits:	1
Hardware Flow control:	False
Hexadecimal Encoding:	○ True ● False
End-Of-Line Pattern:	''
EOM Detech Character Timeout:	1.0

Click on **Select Device**



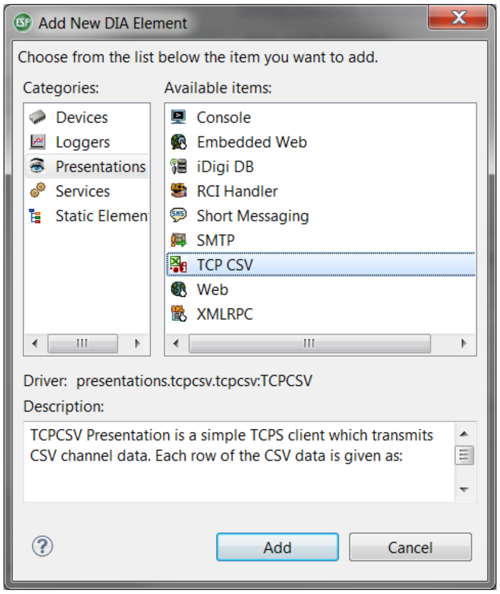
The table below contains the list of XBee devices found by the 'X4' remote configuration. Select the desired XBee Device.

Extended Address	Node Type	Product Type
00:13:a2:00:40:31:b4:59!	Coordinator	ConnectPort X4 G...
00:13:a2:00:40:0a:0f:bd!	Router	Unspecified

2.3 TCP Client CSV

Click on **Add** and select **Presentations**

Click on **TCP CSV** and click **Add**



Choose from the list below the item you want to add.

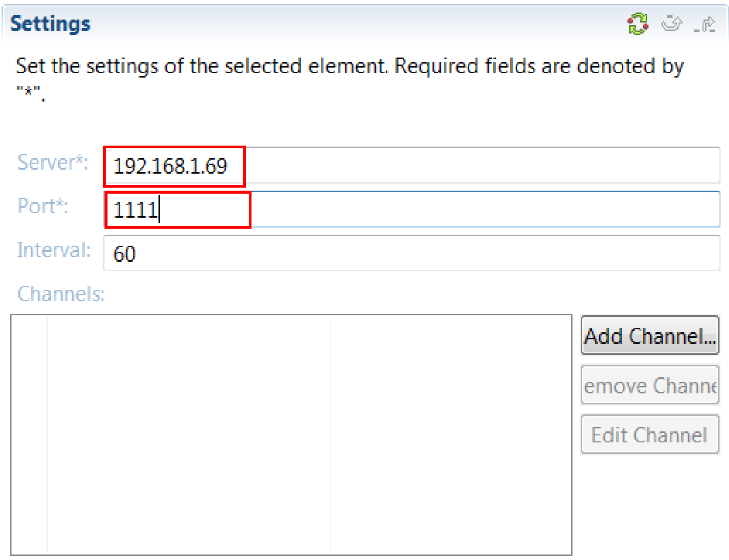
Driver: presentations.tcpcsv.tcpcsv:TCPCSV

Description:

TCPCSV Presentation is a simple TCPS client which transmits CSV channel data. Each row of the CSV data is given as:

Fill in the Server and Port details. In this test we use a simple TCP server listening on port 1111

It is also possible to specify the interval, how often the data should be sent. By default this is done every 1 minute.



Settings

Set the settings of the selected element. Required fields are denoted by "*".

Server*: 192.168.1.69

Port*: 1111

Interval: 60

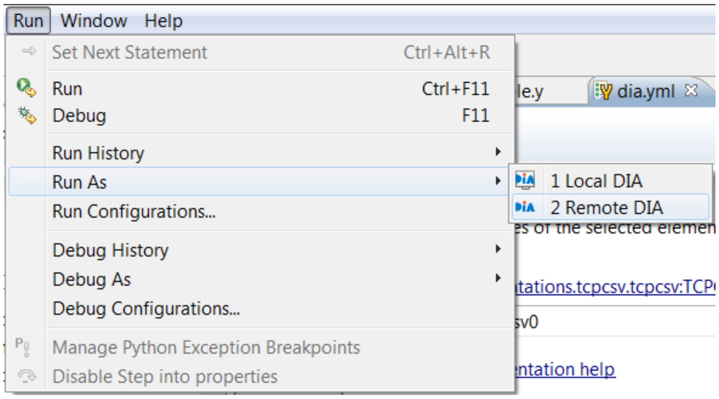
Channels:

The Channels section can be configured to only upload selected channels, or by default will push every channel in this project.

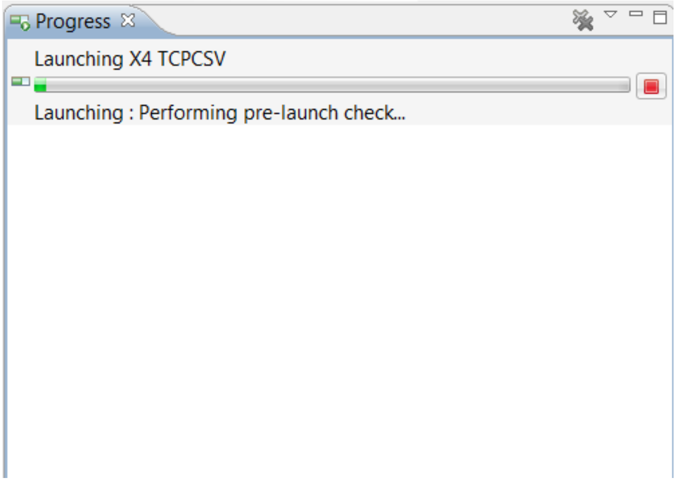
3 RUN PROJECT

3.1 Run the project

Click on **Run** > **Run As** and select **Remote DIA**



Digi ESP will start building the project and upload it to the X4, progress can be seen on the right side



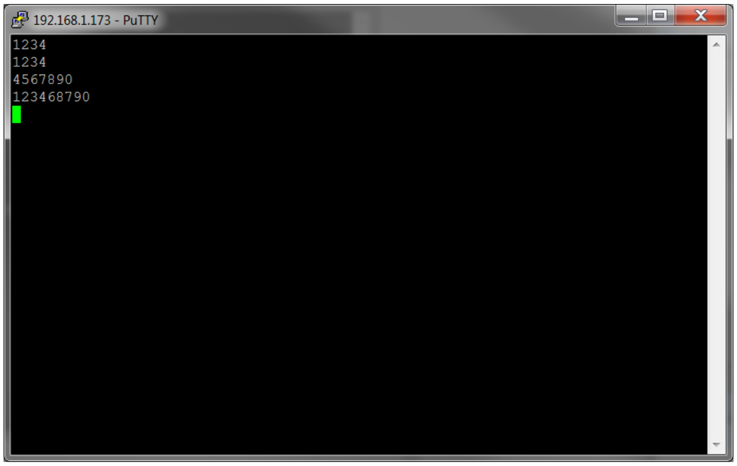
Launching X4 TCPCSV

Launching : Performing pre-launch check...

4 TESTING

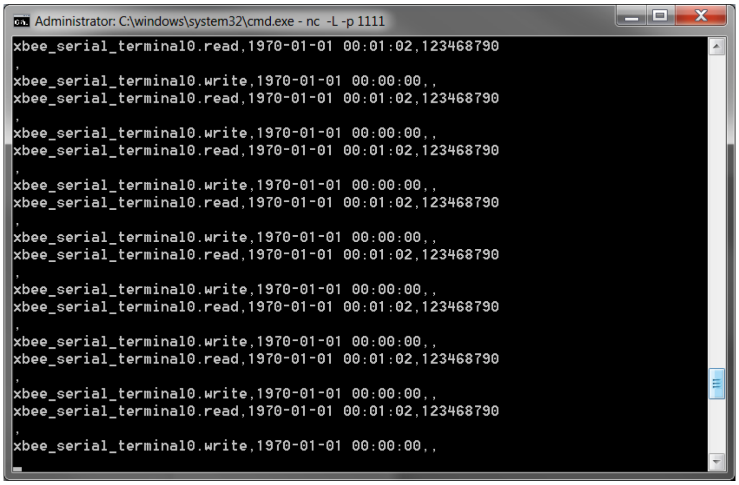
In this example, the XBee Serial Adapter is connected directly to a computer terminal sending serial data.

Data Sent:



```
1234
1234
4567890
123468790
```

Data received by the TCP server in CSV format:



```
xbee_serial_terminal0.read,1970-01-01 00:01:02,123468790
,
xbee_serial_terminal0.write,1970-01-01 00:00:00,,
xbee_serial_terminal0.read,1970-01-01 00:01:02,123468790
,
xbee_serial_terminal0.write,1970-01-01 00:00:00,,
xbee_serial_terminal0.read,1970-01-01 00:01:02,123468790
,
xbee_serial_terminal0.write,1970-01-01 00:00:00,,
xbee_serial_terminal0.read,1970-01-01 00:01:02,123468790
,
xbee_serial_terminal0.write,1970-01-01 00:00:00,,
xbee_serial_terminal0.read,1970-01-01 00:01:02,123468790
,
xbee_serial_terminal0.write,1970-01-01 00:00:00,,
xbee_serial_terminal0.read,1970-01-01 00:01:02,123468790
,
xbee_serial_terminal0.write,1970-01-01 00:00:00,,
xbee_serial_terminal0.read,1970-01-01 00:01:02,123468790
,
xbee_serial_terminal0.write,1970-01-01 00:00:00,,
```

The CSV format is as follow:

```
xbee_serial_terminal0.read,1970-01-01 00:01:02,123468790
```

xbee_serial_terminal0.read: Channel Name

1970-01-01 00:01:02: Date and Time

123468790: Data