NEXCOM International Co., Ltd.

IoT Automation Solutions Business Group

Industrial IoT Remote Gateway
CPS200-RE/CPS200-DP
CPS100-RE/CPS100-DP/CPS100-M

User Manual

NEXCOM International Co., Ltd.
Version 2.0.1
Published October 2019

www.nexcom.com

Contents

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Chapter 1: Product Overview

CPS 200/100 series, an edge/remote Industrial IoT gateway, is fully integrated with fieldbus accessibility, Modbus TCP/RTU, OPC UA and Industrial IoT studio for extremely easy deployment of both centralized/decentralized field data implementation in automation process. Equipped with fieldbus accessibility, user not only can retrieve the data for live monitoring but also extract key information for custom process, like prediction and maintenance, yield rate of production and so on. Furthermore, Industrial IoT studio brings benefits of drag-and-drop data process, exchange field data over network securely between edge and the Cloud, flexible field data store/analytics/statistic and so on. CPS 200/100 series is a perfectly matched solution for remote field data processing in automation.

Specifications

Model Name	CPS100-M	CPS100-RE/ CPS100-DP	CPS200-RE/ CPS200-DP
Photo			
Category	IoT Remote Gateway	Fieldbus Enabled IoT Remote Gateway	Fieldbus Enabled IoT Edge Gateway
Communication Protocols for Local Devices	Modbus/TCP, Modbus/RTU (Master, OPC-UA client)	PROFINET-RT or PROFIBUS-DP or EtherNet/IP (Slave), Modbus/TCP, Modbus/RTU (Master, OPC-UA client)	PROFINET-RT or PROFIBUS-DP or EtherNet/IP (Slave), Modbus/TCP, Modbus/RTU (Master, OPC-UA client)
Communication for Cloud/Server	MQTT, SQLite, Https	MQTT, SQLite, Https	MQTT, SQLite, Https
Wireless Communication Interface Options	Wi-Fi, 3G, 4G/LTE (optional)	Wi-Fi, 3G, 4G/LTE (optional)	Wi-Fi, 3G, 4G/LTE (optional)

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Specifications Cont.

Model Name	CPS100-M	CPS100-RE/ CPS100-DP	CPS200-RE/ CPS200-DP
Photo			
Number of LAN ports	2	2	2
Type of LAN	RJ45	RJ45	RJ45
COM Port	1 x RS-232/485	1 x RS-232/485	2 x RS-232/485
USB	1 x USB 3.0 1 x USB 2.0	1 x USB 3.0 1 x USB 2.0	1 x USB 3.0 3 x USB 2.0
Display	N/A	N/A	1 x DVI
Mounting Style	Wall/DIN Rail	Wall/DIN Rail	Wall/DIN Rail
Temperature	-20°C ~ +65°C	-20°C ~ +65°C	0°C ~ +50°C
Dimension (mm)	63 x 100 x 151	63 x 100 x 151	85 x 157 x 214
DC Input	12VDC/24VDC	12VDC/24VDC	24VDC
Certification	CE, FCC	CE, FCC	CE, FCC
Storage	16GB eMMC	16GB eMMC	2.5”128GB SSD

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Chapter 2: Hardware Installation

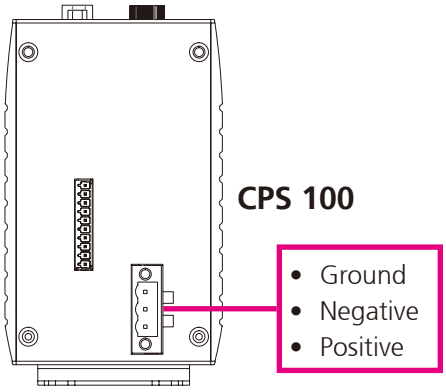
2.1 Cabling

Items to prepare

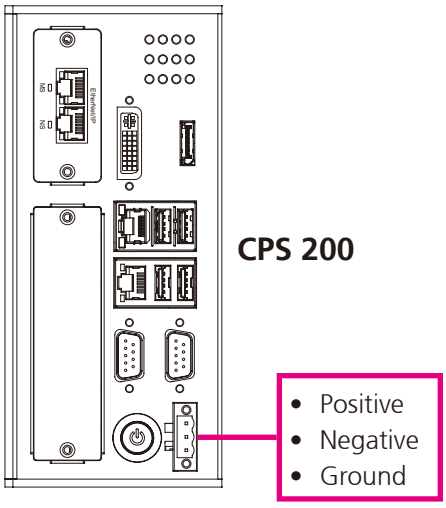
1. A power cable with DC 24 V output and a Phoenix Contact 1x3 3-pin terminal block.
   - ✓ For CPS 100 only, you can use a power cable with DC 12 V output and a Phoenix Contact 1x3 3-pin terminal block.
2. An Ethernet cable.
3. A set of CPS 200 or CPS 100.

Step 1

- ➢ While wiring the terminal block, please be aware, on CPS 100, from the top, that the sequence of power terminal definitions of the plug is ground, negative, and positive.



- ➢ On the other hand, on CPS 200, from the top, the sequence of power terminal definitions of the plug is positive, negative, and ground.



Once the wiring on the terminal block is set, insert the power cable to the power plug.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

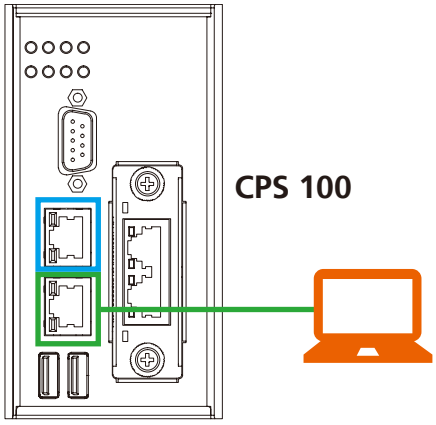
Step 2

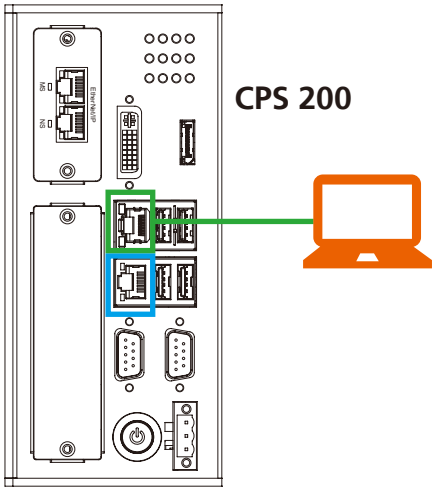
- If you have a DHCP enabled network infrastructure, plug the Ethernet cable into the RJ-45 jack circled in blue and labeled as 1.
- It is recommended to connect directly between the device and your computer, plug the Ethernet cable into the RJ-45 jack circled in green and labeled as 2.

⇨ The IP address: 192.168.253.1 is assigned to this jack by default.

⇨ Please set your computer's IP address in the same network segment such as 192.168.253.5 with subnet mask 255.255.255.0, and try to ping the IP address: 192.168.253.1.

⇨ If it is reachable from your computer, use your browser to log onto 192.168.253.1 for system setting.



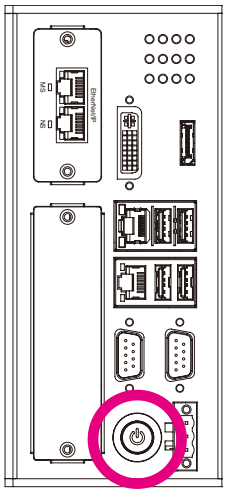


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Chapter 3: System Configuration

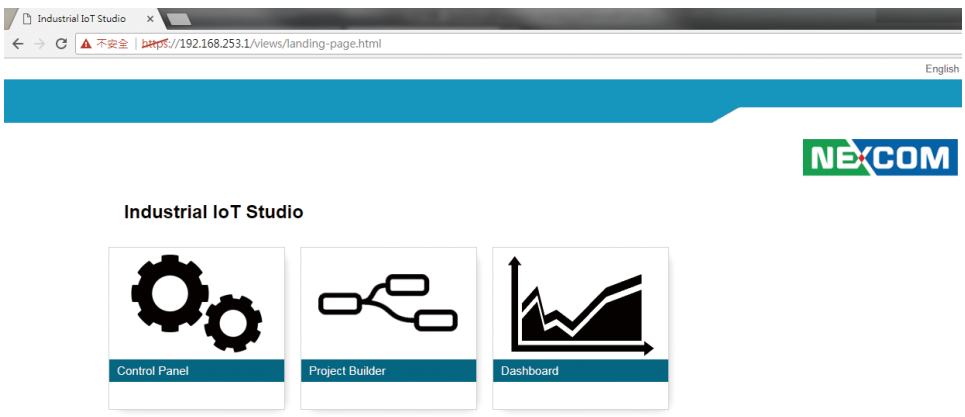
3.1 Start Up

- CPS 100 will start up automatically once the inserted power cable is live.
- CPS 200: Press the power button to start up the system.



3.2 Networking

It is recommended to log onto 192.168.253.1 with your browser and Ethernet cable connected to 2. If you are connected to 1, consult your DHCP server for the IP address of your CPS 200 or CPS 100, and log onto the IP address with your browser. You will see a landing page of IIoT Studio. Go to "Control Panel" for system setting, "Project Builder" for programming and "Dashboard" for the information graph.



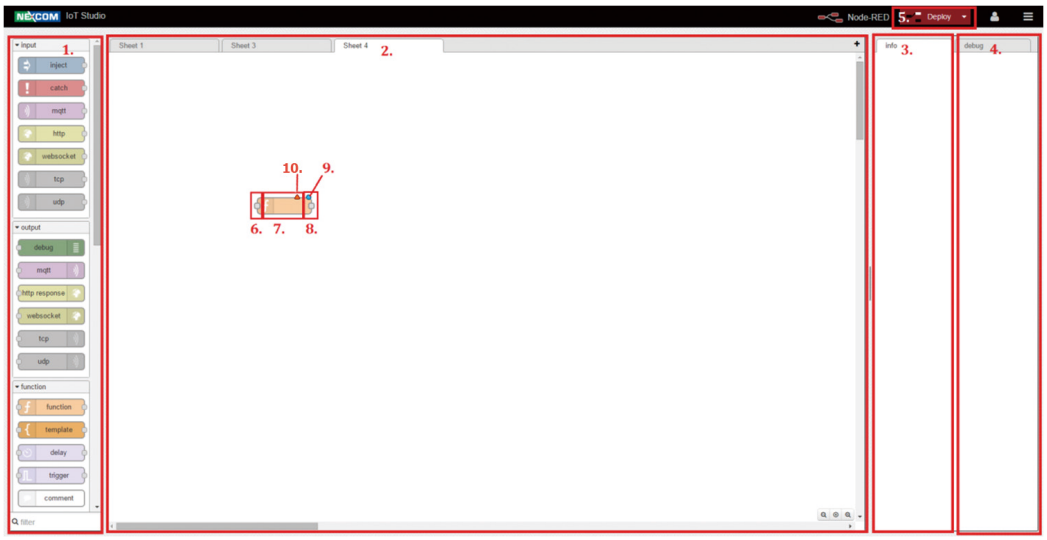
⇨ If you are unable to log onto the IP address because of the security issue with your browser, click on continue browsing this website to change settings of your browser.

- Use the default **Username**: *admin* and **Password**: *12345678* for login if you are logging in for the first time.
- Remember to change username/password and keep it in a safe area to avoid hacking.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Chapter 4: IoT Studio Introduction

This chapter introduces the user interface and the basic operation of NEXCOM IoT Studio. Once you log onto NEXCOM IoT Studio with your browser and click on "Project Builder", you will see the page as shown below.

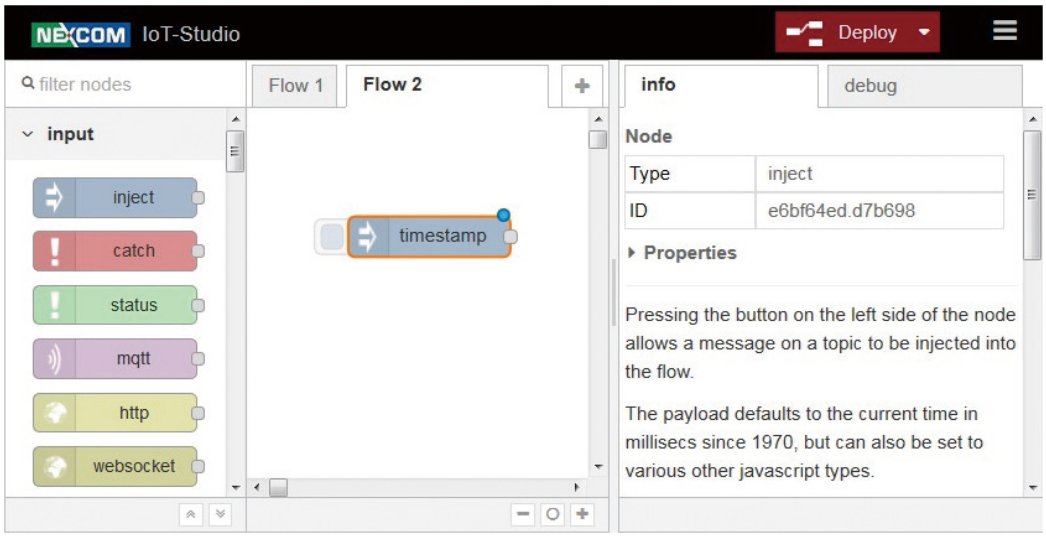


No.	Description	No.	Description
1	Nodes	6	Input port of the node, receiving from a connected note.
2	Workspace	7	Note title
3	Node information	8	Output port of the node, delivering to other node.
4	Debug message once in debugging a node	9	Status of the node
5	Press to start your flow	10	Missing parameters

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

4.1 Basic Operations

The icons at the left side of the page are nodes corresponding to different needs. You can drag and drop any of them to the workspace. Click on the **Info** tab on the upper right to view the information of the node you selected.



4.1.1 Drag and Drop

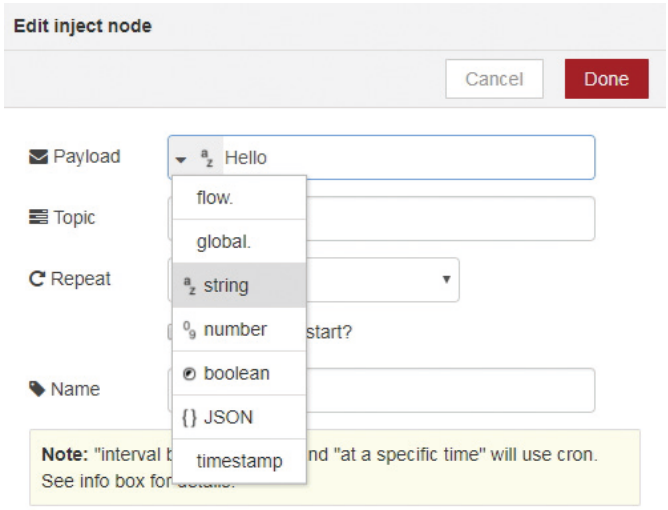
You can drag many nodes onto a sheet and make them a flow based on the functions and connections. Please note, by the nature of the node, some of them have both the input and the output ports while the others have either one of the ports.

You can make as many connections as you like between your start and end nodes. Furthermore, you can make one output port connect to different input nodes. Once the connection is set, you can code up the flow.

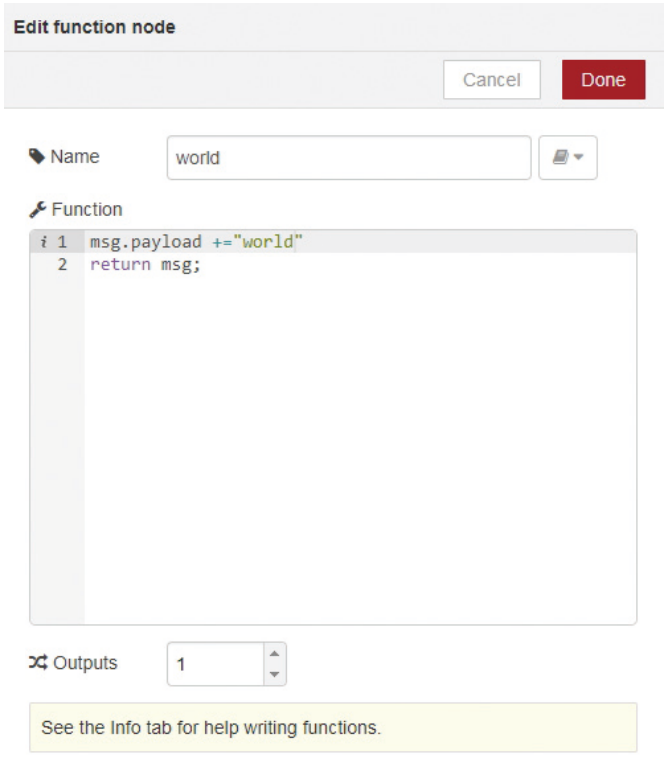
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

4.1.2 Code Up Your Flow

Double click on the inject node, and the edit dialogue should pop up. Select **string** in the drop-down menu next to Payload, fill the next field with "hello", and click **Done**.

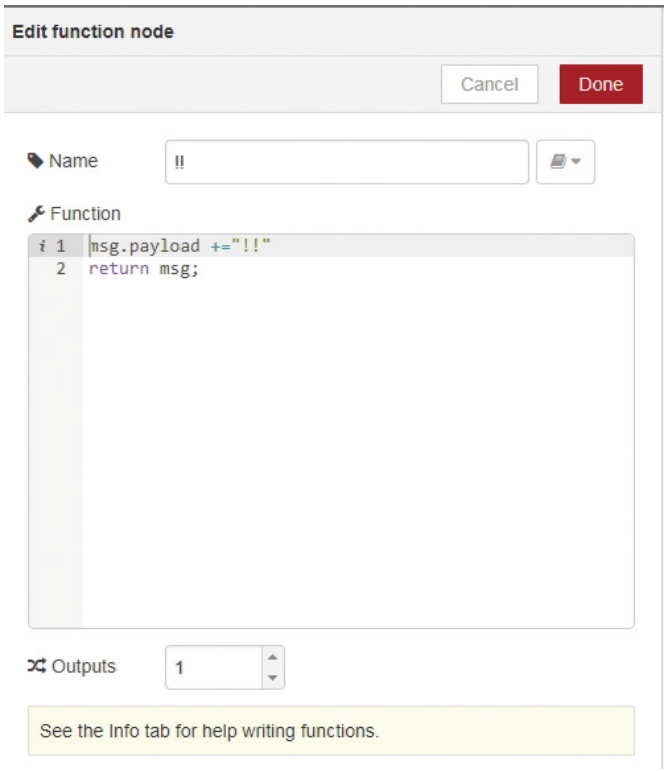


Double click on the second node, and name it "world". Write some codes to the node in **Function** field and click **Done**.

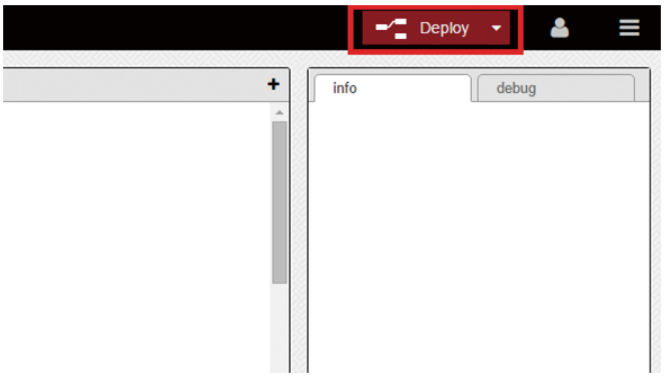


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Double click on the third node, and name it "!!". Write some codes to the node in **Function** field as shown below and click **Done**.



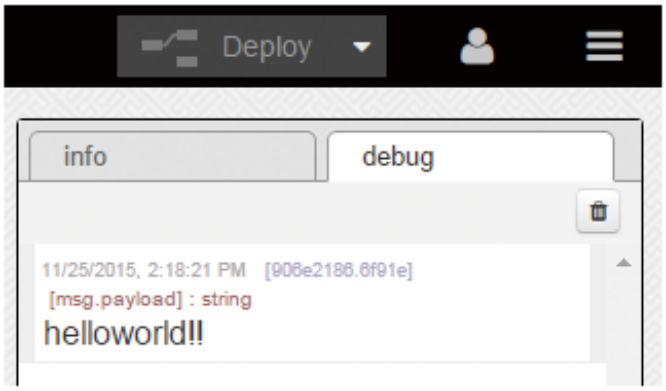
Then, click **Deploy** button on the upper right side to deploy your flow.



When you can read the message **Successfully deployed** on the top, the deployment is complete.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Click the button at the left side of the inject node to see the result in the **debug** tab.



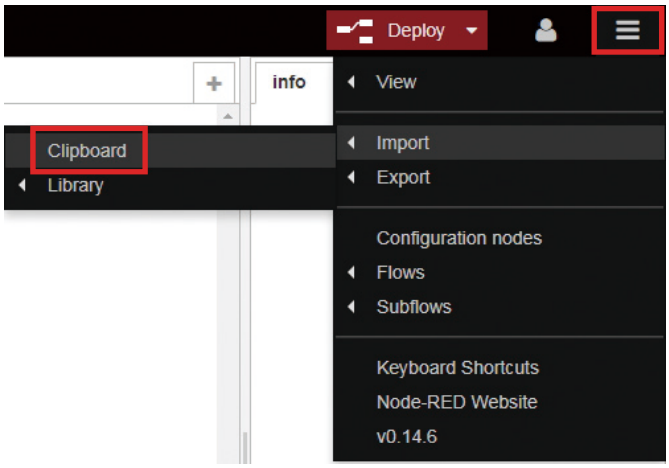
4.2 IoT Studio Administrations

You can share the codes by selecting your node, and then choose Export from the menu on the upper right corner either by copying the codes or exporting them to a file directly.

4.2.1 Import

To import the codes, you can choose Import from the menu on the upper right corner and paste the codes to the clipboard or select a file to import.

Click **Deploy** to execute the code in the flow.

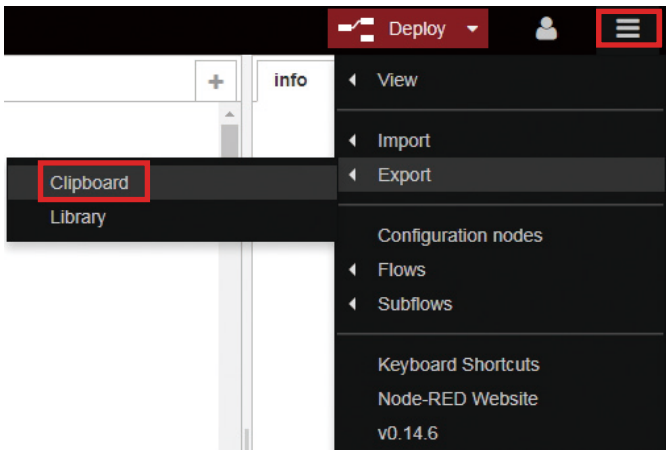


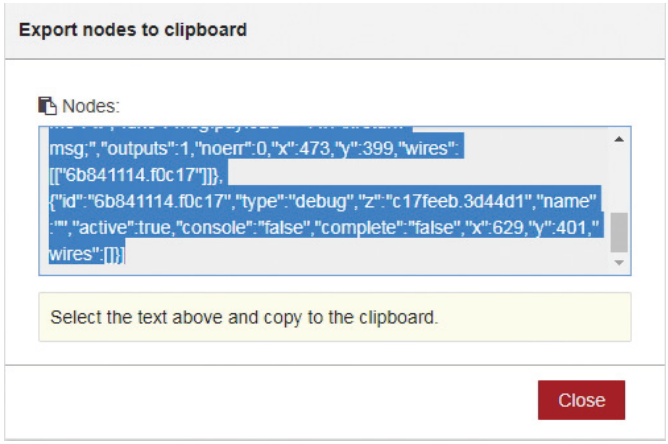
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

4.2.2 Export

To export your codes, select the node and choose **Export** from the menu on the upper right corner.

You can copy the code or export the code to a file directly.





Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Chapter 5: Fieldbus Configuration

Once you have logged onto the IP address, you should see a page titled "Edge Gateway manager". Click **NodeRed** to enter **Node-RED** page.

5.1 Fieldbus Input Node

The **fieldbusinput** node allows you to receive data from a fieldbus e.g. PROFINET® and uses signals to address the data. The fieldbus has to be configured and signals have to be defined before using the fieldbus node. This following example uses the **fieldbusinput** node together with a debug node to receive data from a fieldbus and to display the data thereafter in the **debug** tab of the IoT Studio™.

Prerequisite

1. A connection with the Edge Gateway is established.
2. The IoT Studio™ workspace is opened.

Steps

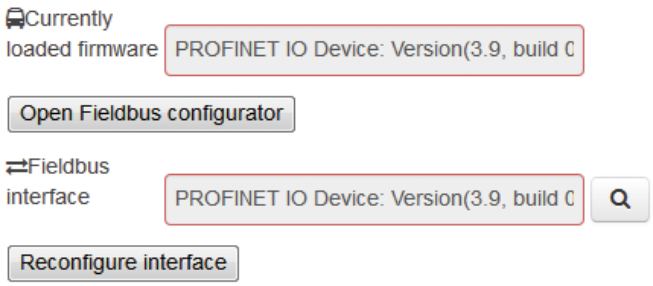
1. Inserting the fieldbus input node:
   - Drag a **fieldbusinput** node from the node library and drop it in the worksheet.
   - The fieldbus node shows a red triangle to indicate that parameters are missing. These parameters are configured during the next steps.

2. Showing the sidebar:
   - Show the sidebar via the IoT Studio™ menu **View**>**Show Sidebar**.
   - Click the **Info** tab.
   - Click on the **fieldbus input** node to display its properties and a functional description in the **Info** tab.

3. Editing the fieldbus input node:
   - To open the edit dialog, double-click on the fieldbus input node.
   - The edit dialog for entering the parameters will pop up.

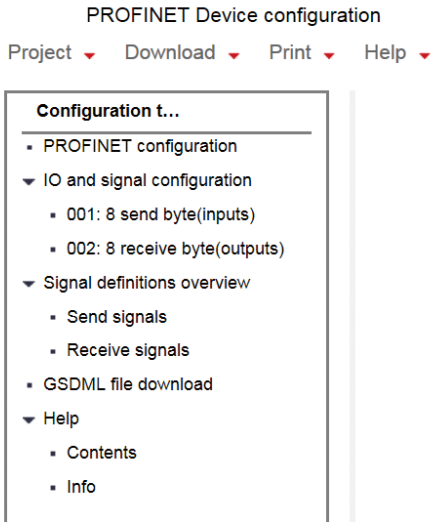
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

4. Adding a fieldbus interface:
   - If you use a fieldbus node for the first time (the **Fieldbus interface** list displays **Add new fieldbus interface**), click to add a new fieldbus interface.
   - Selecting the fieldbus interfaces:



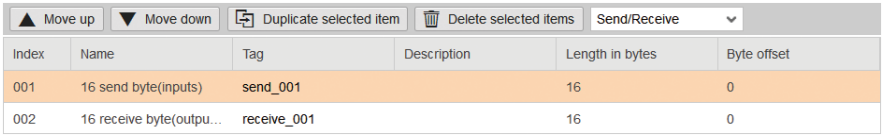
5. Fieldbus interface selection
   - Verify that at **Currently loaded firmware** the entry **PROFINET IO Device: Version ...** is displayed.
   - If this is not the case, then select **Fieldbus interface** using the entry **PROFINET IO Device** and then click **Reconfigure interface**.

6. Opening the fieldbus configurator (PROFINET configurator):
   - If **Currently loaded firmware** does not display the entry **PROFINET IO Device** then repeat step 5.
   - If **Currently loaded firmware** shows the entry **PROFINET IO Device** then click on **Open Fieldbus Configurator**.
   - A new tab opens displaying the PROFINET configuration user interface.

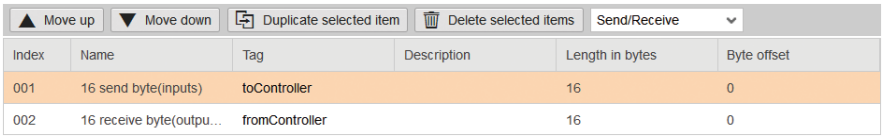


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

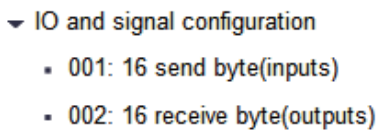
7. Configure PROFINET:
   - Select **IO and Signal configuration** in the configuration tree.
   - The **IO Configuration** page with the **Available IO items** list is displayed.
   - First, select the entry **16 send byte (inputs)** with a double-click from the **Available IO items** list.
   - Afterwards, select the entry **16 receive byte (outputs)** with a double-click from the **Available IO items** list.
   - The **IO items** list displays two PROFINET modules.



8. Defining signal names for modules:
   - In the **IO items** list, click on the **Tag** column on **receive_001** (Index **001** with the name **16 send byte (inputs)**).
   - Overwrite the existing **Tag** name by the new **Tag** name: **toController**.
   - In the **IO items** list, click on the **Tag** column on **send_001** (Index **002** with the name **16 receive byte (outputs)**).
   - Overwrite the existing **Tag** name by the new **Tag** name: **fromController**.
   - A new **Tag** is displayed for each module, which is part of the complete signal name.



9. Defining signals for module 1:
   - Select **001: 16 send byte (inputs)** in the configuration tree.

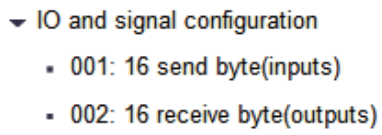


   - The signal configuration page with information about module 001 is displayed.
   - Mark the existing signal in the signal list with a left mouse click on it.
   - Click **Delete selected items**.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

- Click **Add new signals**.
- The **Add new signals** dialog box is displayed.
- For data type, select signed16.
- For quantity, select 8.
- Enter **Temperature** for the base tag name.
- Click **Ok**.
- **Temperature_1 u_8** are displayed in the **Tag** column.
- You can adapt the signal name in the **Tag** column if necessary.

10. Defining signals for module 2:
    - Select **002: 16 receive byte (outputs)** in the configuration tree.



    - The signal configuration page with information about module 002 is displayed
    - Mark the existing signal in the signal list with a left mouse click on it.
    - Click **Delete selected items**.
    - Click **Add new signals**.
    - The **Add new signals** dialog box is displayed.
    - For data type, select signed16.
    - For quantity, select 8.
    - Enter **Set_temperature** for the base tag name.
    - Click **Ok**.
    - **Set_temperature_1** up to **Set_temperature_8** are displayed in the **Tag** column.
    - You can adapt the signal name in the **Tag** column if necessary.

11. Saving the configuration:
    - Click **Project>Save** to save the configuration in the Edge Gateway.
    - The message "The configuration has been successfully saved" is displayed.
    - Click **Ok**.
    - The PROFINET configuration and the signal definitions are saved in the Edge Gateway, but not activated yet.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

12. Switch back to IoT Studio™:
    - Switch in your browser back to the **NodeRED** tab.
    - Click **Reconfigure interface** to take over a new/changes PROFINET configuration.
    - Click **Add** to add a new fieldbus interface.
      or
    - Alternatively, **Update** is displayed. Click **Update** if signal names are expanded or changed but the PROFINET configuration is not changed.
    - The display changes back to the edit dialog.

13. Enter name:
    - Enter the node name e.g. **MyInput**.

14. Selecting a signal:
    - Open with the signal list and select a **Signal** e.g. **input~from Controller~Set_temperature_2**. If **Signal** is not displayed then click **Done** to close the edit dialog and reopen the fieldbus node with a double-click.

15. Finishing the fieldbus input node:
    - Click **Done**.
    - The fieldbus input node is completed but not activated in the Edge Gateway.

16. Inserting a debug node:
    - Drag a Debug node from the node library and drop it in the worksheet.

17. Connecting the nodes:
    - To connect the fieldbus node with the Debug node, hold down the left mouse button and draw a connecting line (wire) from the output port of the fieldbus node to the input port of the Debug node.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

18. Deploy:
    - Click **Deploy** to transmit the nodes to the device and activate them.
    - The flow is activated in the Edge Gateway.

    - As soon as the value of the signal changes, the Debug output will display the new value/values and status information.

5.2 Configuring the Fieldbus Node

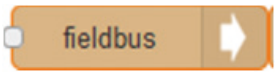
The fieldbus input node allows you to receive data from the fieldbus system (e.g. PROFINET). The fieldbus output node allows you to send data to the fieldbus system. Each fieldbus node needs signals to address data. At first, configure the fieldbus system, then define the signals, and thereafter use signals for the fieldbus input node or the fieldbus output node.

The **fieldbus input node** (fieldbus in) receives output data from the master: Fieldbus master (e.g. PROFINET IO controller) ➔ fieldbus input node

Standard Procedure

1. Inserting the fieldbus input node.
2. Configuring the fieldbus system (e.g. PROFINET).
3. Defining the signals.
4. Configuring the fieldbus input node.

The **fieldbus output node** (fieldbus out) sends input data to the master: Fieldbus output node ➔ fieldbus master (e.g. PROFINET IO controller)



Standard Procedure

1. Inserting the fieldbus output node.
2. Configuring the fieldbus system (e.g. PROFINET).
3. Defining the signals.
4. Configuring the fieldbus output node.

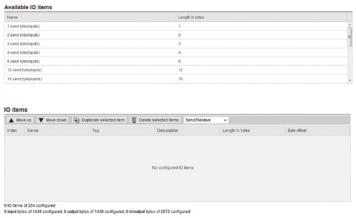
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

The following table shows the sequence of the configuration steps:

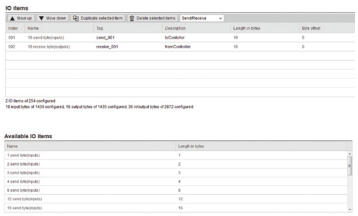
1. Open fieldbus node.

2. Open PROFINET configuration.

3. Configure modules.



4. Define signals.

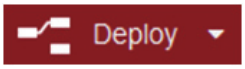


5. Project > Save.

6. Fieldbus node (configured).

7. Reconfigure/Update.

8. Deploy



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

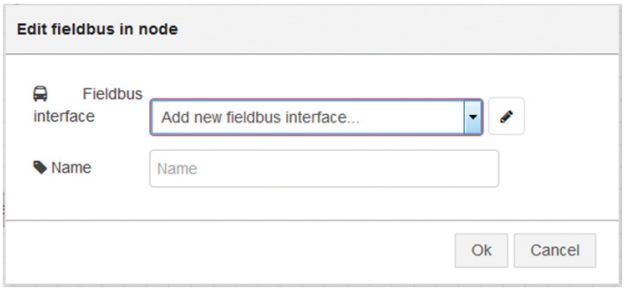
5.2.1 Creating a New Fieldbus Configuration

This section gives an example describing how to open the fieldbus configuration of the Edge Gateway from NodeRED using the PROFINET configuration.

Prerequisite:
The NodeRED workspace is launched.

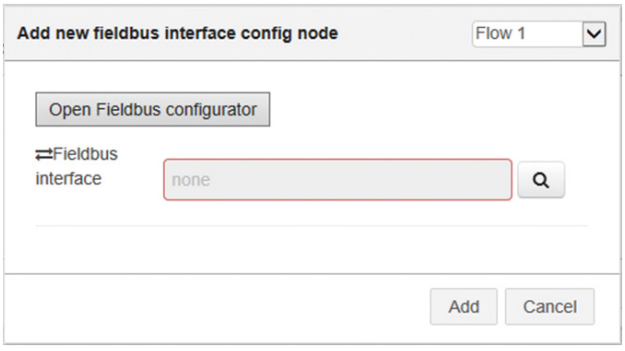
1. Inserting and opening the fieldbus node.
   - ➢ Drag a fieldbus node from the node library and drop it in the workspace.

   - ➢ Double click the fieldbus node.
   - ⇨ The edit dialog Edit fieldbus in node will be displayed.



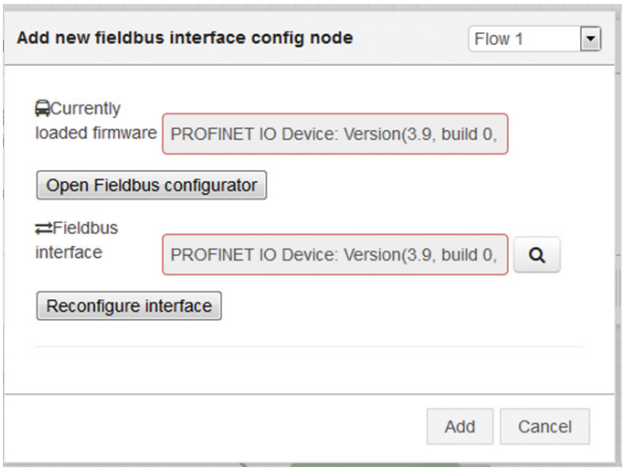
   - ➢ Select Add new fieldbus interface from the list.

2. Opening the PROFINET configuration
   - ➢ Click the square to the right of fieldbus interface.
   - ⇨ The dialog box *Add new fieldbus interface config node* opens. If no fieldbus interface has been defined yet, the box will look as follows:



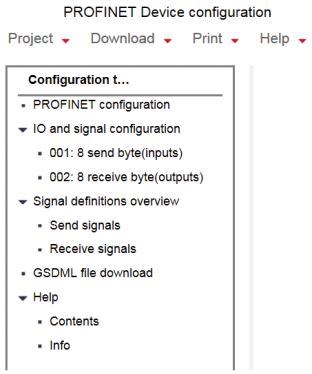
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

If one or several fieldbus interfaces have already been defined, the interface/s will be offered for selection in the list. In that case, the dialog box *Add new fieldbus interface config node* looks differently:



➢ Click **Open fieldbus configurator**. 

➢ The user interface for the fieldbus configuration of the Edge Gateway will be opened in a new browser window: (The example shows the PROFINET configuration)



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

3. Configuring the modules
   - ➢ Click **IO and signal configuration** in the configuration tree (left).
   - ➢ The tables **Available IO items** and **IO items** will be shown in the display area and the workspace.
   - ➢ To select the inputs and outputs to be configured, double click the appropriate entry in the upper table **Available IO items**, e.g. 4 byte input and 8 byte output.

Note: The table entries are not all visible at the same time, but by scrolling you should be able to find the desired number of input or output bytes quickly.

Available IO items

Name	Length in bytes
1 send byte(inputs)	1
2 send byte(inputs)	2
3 send byte(inputs)	3
4 send byte(inputs)	4
8 send byte(inputs)	8
12 send byte(inputs)	12
16 send byte(inputs)	16

- ➢ After each double click, the corresponding entry for defining an input or output signal will be displayed immediately in a new line of the lower table **IO items**.

IO items

Move up | Move down | Duplicate selected item | Delete selected items | Send/Receive

Index	Name	Tag	Description	Length in bytes	Byte offset
001	16 send byte(inputs)	toController		16	0
002	16 receive byte(outputs)	fromController		16	0

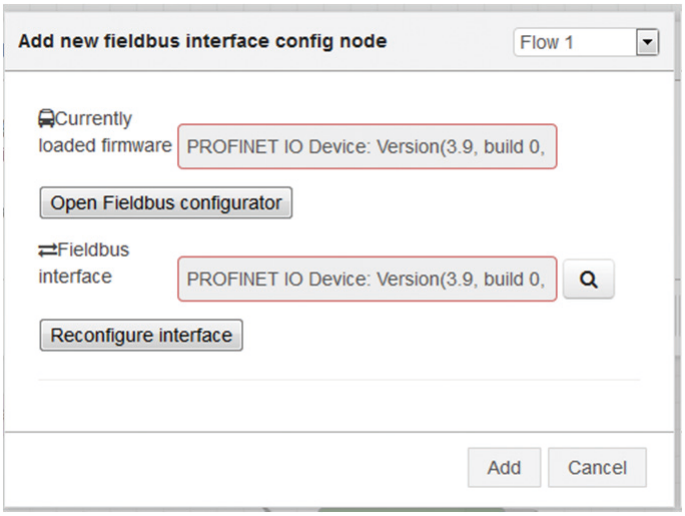
4. Defining signals
   - ➢ As standard a signal definition is performed during, which one single signal of the data type *OctetString* will be defined, which uses the entire available data length. If you wish to have another signal definition, you can delete this definition (with the button Delete selected items) and define new signals until the available data length is used up (with the button Add new signals). In the table columns **Tag** and **Description**, you can enter a short name and a description for each defined signal.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

To finish the definition, proceed as follows:

5. Saving the signal configuration
   - ➢ Save your signal configuration via the menu function **Project** > **Save**.
   - ➢ The steps to be made in the user interface for the fieldbus configuration of the Edge Gateway are thus completed.
   - ➢ Return to the NodeRED browser window that was originally used.

6. Reconfigure interface: Transferring firmware and configuration to the Edge Gateway
   - ➢ Click the button *Reconfigure* in the dialog *Edit fieldbus interface config node*.



This is necessary to make sure that the following information on the fieldbus interface in the Edge Gateway can be transmitted later: Firmware, configuration, and signal definitions.

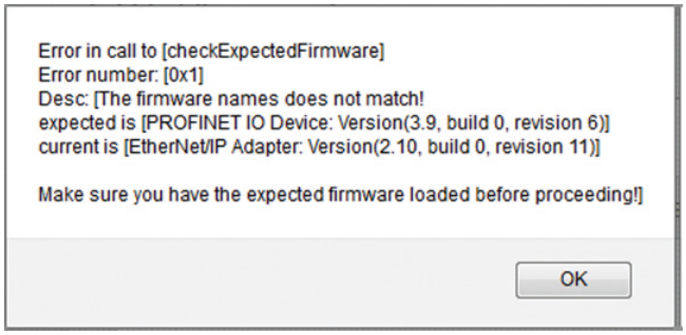
- ➢ Click the button *Add* in the dialog *Add new fieldbus interface config node*.
- ➢ This operation causes the transmission of the fieldbus configuration to the device and the updating of the signal definitions to the fieldbus node.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7. Deploy
   - ➢ Perform a Deploy in the NodeRED. For this purpose, select one of the three options of the list Deploy in the NodeRED window (top, right).
   - ⇨ The flow will thus be transmitted to the Edge Gateway, and the new configuration will be applied in the Edge Gateway.

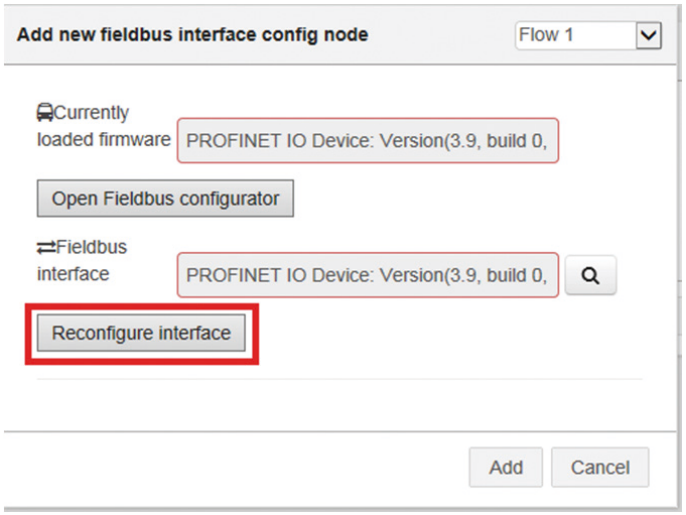
Message displayed when the expected firmware and the determined firmware do not match.

Should you receive a message like the one shown below when you open the user interface to the fieldbus configuration of the Edge Gateway (after step 5), the expected firmware and the actually determined firmware do not match (Ethernet/IP Adapter is expected in the example shown, but PROFINET IO Device is actually loaded):



In that case, check whether the desired firmware is set under *Fieldbus interface*.

If this is not the case, select the desired firmware in the list **Fieldbus interface** and click **Reconfigure interface**.



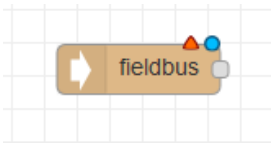
5.2.2 Changing the Existing Fieldbus Configuration

This section describes how to change the existing fieldbus configuration (in this case: PROFINET) for Edge Gateways from NodeRED.

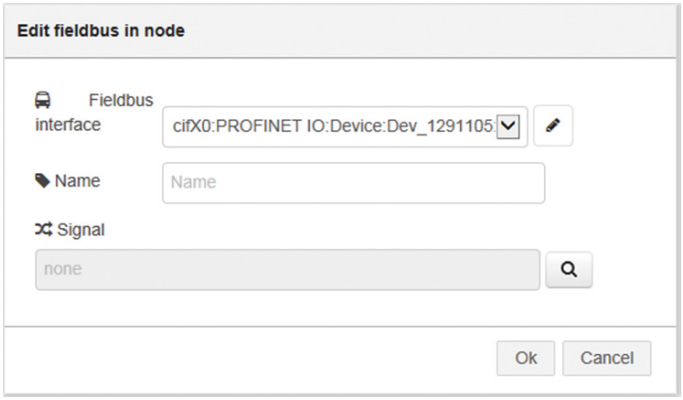
Prerequisite:

- The NodeRED workspace is open.
- There must be a flow contains a fieldbus node on the NodeRED screen. The node represents the previous fieldbus configuration of the Edge Gateway.

1. Opening the fieldbus node for changing
   - In the workspace, double click on the fieldbus node to be changed. It is assumed that the node shown in the examples is preconfigured for PROFINET.

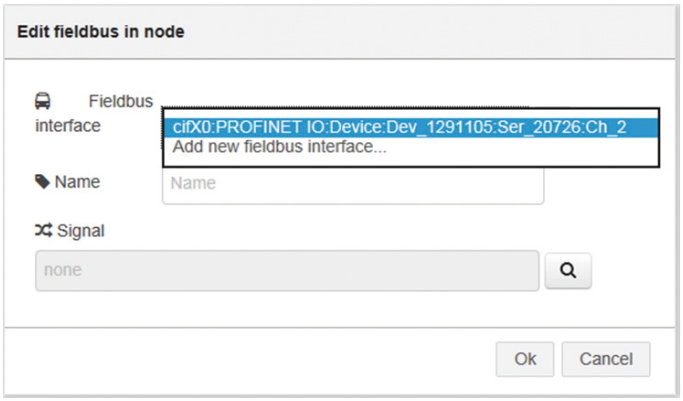


   - The edit dialog **Edit fieldbus in node** will be displayed.



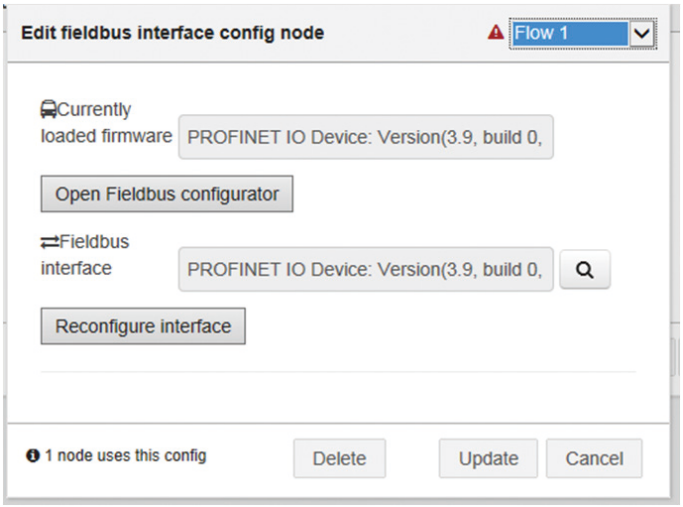
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

- The selection list Fieldbus Interface displays all fieldbus interfaces that are already configured. (In addition to that, there is an entry Add new fieldbus interface for adding a new fieldbus interface.)



- Select the interface to be changed (in the example this is cifX0 PROFINET...).

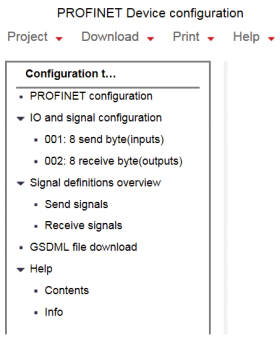
2. Opening the PROFINET configuration
   - Click the square to the right of fieldbus interface. 
   - The dialog box *Edit fieldbus interface config node* opens.



- Click Open fieldbus configurator.

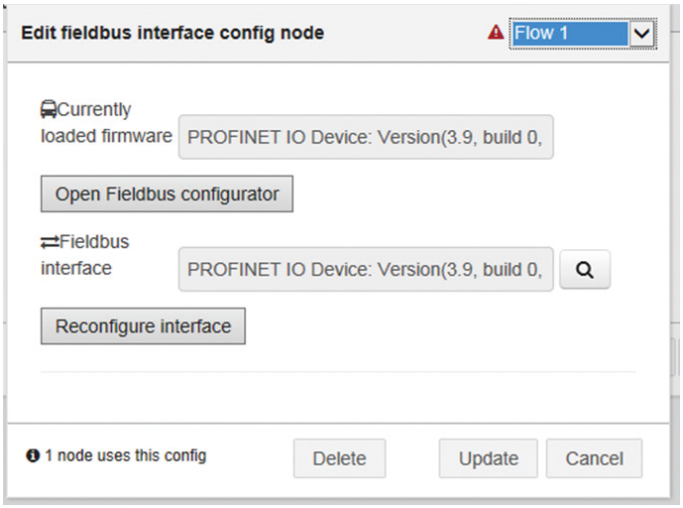
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

⇨ The user interface for fieldbus configuration of the currently loaded firmware of the Edge Gateway will be opened in a new browser window: If (as this example shows) PROFINET IO Device is the currently loaded firmware, the user interface for PROFINET configuration of the Edge Gateway will open, see figure:



3. Configuring modules
   - ➢ Make your changes in the fieldbus configuration. Refer to 5.3 Configuring the PROFINET and 5.3.3 IO and signal configuration for more information.
4. Defining signals
   - ➢ If required, adapt the signal definitions as you desire.
5. Saving the signal configuration
   - ➢ Once you have made all desired changes, save your fieldbus configuration as described in section 5.3.1 Main menu.
   - ⇦ The steps to be made in the user interface for fieldbus configuration of the Edge Gateway are thus completed.
   - ➢ Return to the NodeRED browser window that was originally used.
6. Reconfigure interface: Transferring firmware and configuration to the Edge Gateway
   - ➢ Click the button *Reconfigure* in the dialog *Edit fieldbus interface config node*.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.



This is necessary to make sure that the following information on the fieldbus interface in the Edge Gateway can be transmitted later: Firmware, configuration, and signal definitions.

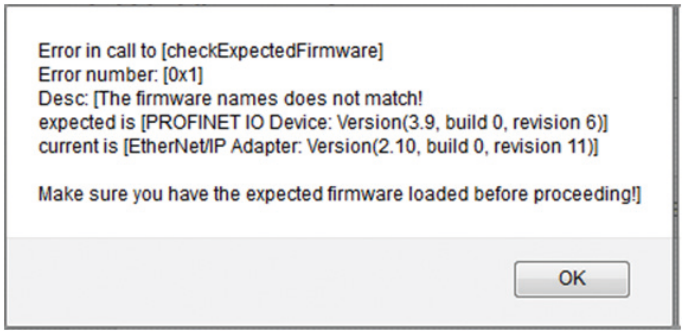
For detail of what happens when you click the button Reconfigure interface, refer to 5.2.3 Loading firmware and configuration.

- ➢ Click the button *Update* in the dialog *Edit fieldbus interface config node*.
- ⇜ This operation causes the transmission of the fieldbus configuration (in this case: PROFINET) to the device and the updating of the signal definitions to the fieldbus node.

7. Deploy
   - ➢ Perform a Deploy in the NodeRED. For this purpose, select one of the three options of the list *Deploy* in the NodeRED window (top, right).
   - ⇨ The flow will thus be transmitted to the netIOT Edge Gateway, and the new configuration will be applied in the Edge Gateway.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

If the user interface in step 4 cannot be started successfully, a message will be displayed which is similar to the following:



That means that the expected firmware and the actually determined firmware do not match. Check your settings in such a case!

5.2.3 Loading Firmware and Configuration

The button **Reconfigure interface** in the dialog Add new fieldbus interface config node serves to reconfigure the Edge Gateway completely including a firmware exchange. The previous firmware will be overwritten thereby.

In case of **Reconfigure interface**, the following information will be transmitted to the fieldbus interface in the Edge Gateway:

- Firmware
- Configuration
- Signal definitions

To exchange the firmware in the Edge Gateway for the firmware selected under **Fieldbus interface**, proceed as follows:

➢ Click **Reconfigure interface**.

↬ Two messages can appear depending on the situation. If the same firmware is selected under *Currently loaded firmware* and *Fieldbus interface*, a message box appears with the string "The firmware [xxxx] is already on board."

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

That means that the firmware has not been exchanged. The currently loaded firmware as well as its version number, build number, and revision number are displayed in the message box.

If the firmware loaded under *Currently loaded firmware* and *Fieldbus interface* differs, a message box appears with the wording of firmware mismatch.

This message draws the attention to the consequences the firmware exchange will have for the running operation (the network communication of the Gateway will stop, and modifications to the network configuration will be lost).

➢ If you are sure that you want to exchange the firmware although you are aware of the consequences for the running operation, click **OK**.

⇨ Clicking OK triggers the following actions:

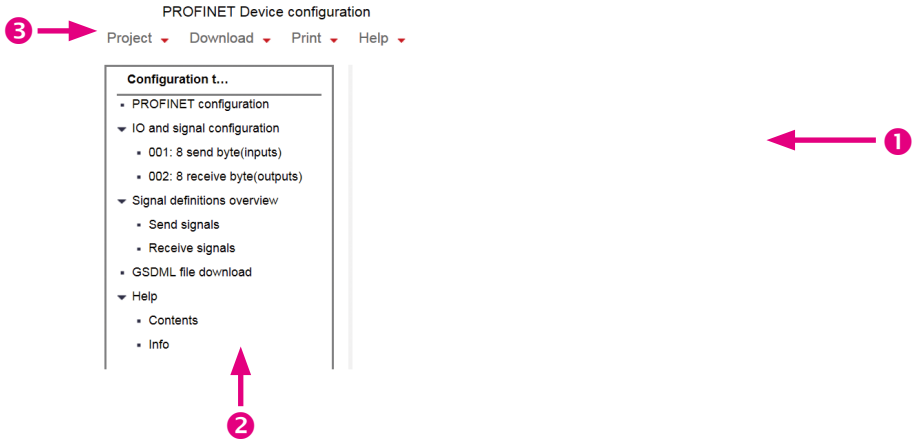
1. The previous firmware will be deleted.
2. The bootloader will be loaded into the Edge Gateway.
3. The bootloader will be executed to load the selected firmware into the Edge Gateway.
4. The new firmware loaded will be started.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

5.3 Configuring the PROFINET

5.3.1 Main Menu

The following figure and table describe the elements of the user interface.



Position number	Description
1	Display area and workspace
2	Configuration tree
3	PROFINET configuration menu

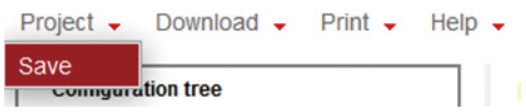
The menu bar of the PROFINET configuration of the netIOT Edge Gateway allows you to

- save the configuration in the Edge Gateway,
- print the configuration,
- download the GSDML file from the Edge Gateway,
- call the help page, and
- display the software version of the configuration surface.

1. Save

   To save the configuration data in the device, proceed as follows:

   ➢ Click **Project** > **Save**.



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

⇨ The configuration will be saved in the Edge Gateway.

2. GSDML-file download
   The GSDML file, required for configuring the PROFINET controller, contains the PROFINET properties of the Edge Gateway. Download the GSDML file from the Gateway in order to be able to use it in the configuration tool of the PROFINET controller.

   To download the GSDML file, proceed as follows:
   ➢ Click **Download** > **GSDML**.

   ⇨ A dialog for saving the GSDML file will be displayed.
   ➢ Select a folder and click **Save**.
   ➢ Use the saved GSDML file to configure the PROFINET controller.

3. Print the configuration
   To print configuration and signal definition, proceed as follows:
   ➢ Click **Print** > **Print configuration**.

   ⇨ The print dialog box will be displayed.
   ➢ Select a printer or an output file and set the printing parameters.
   ⇨ The current configuration and signal definition will be printed.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

4. Help

   To display the help page, proceed as follows:

   ➢ Click **Help** > **Contents**.

5. Version

   To display the version of the PROFINET configuration software of the netIOT Edge Gateway, proceed as follows:

   ➢ Click **Help** > **Info**.

   ⇨ The version will be displayed.

5.3.2 PROFINET Configuration

The PROFINET device name in factory state of this device is set to an empty string. The name must be assigned with a protocol known as PROFINET DCP from an engineering tool of the PROFINET IO controller or a standalone tool over any of its PROFINET ports.

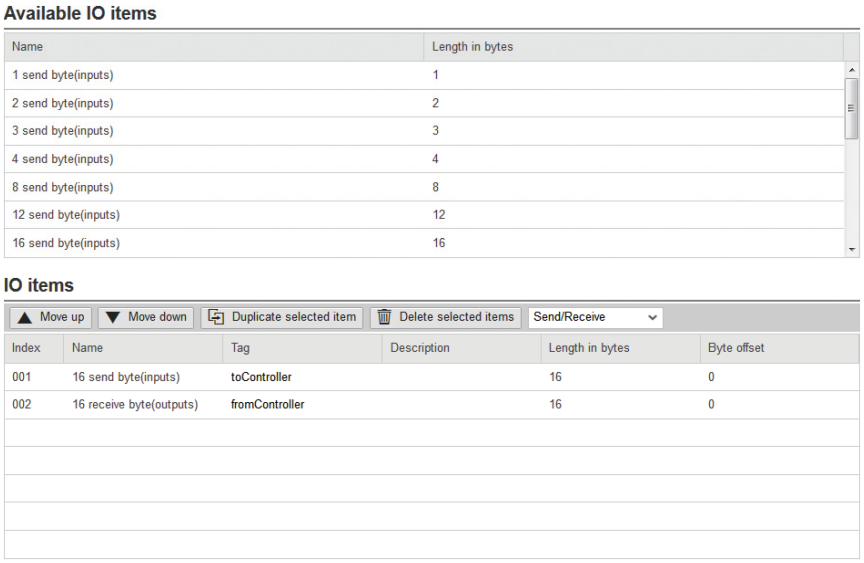
5.3.3 IO and Signal Configuration

On this page, you configure the PROFINET input and output modules for the Edge Gateway. The Edge Gateway is a PROFINET IO device.

The upper table (*Available IO items*) displays the list of the possible (selectable) input and output modules. The lower table (*IO items*) displays the PROFINET configuration and is empty at the beginning.

An "IO Item" corresponds to a PROFINET input or output module. The data length of PROFINET modules differs: Possible lengths are 1, 2, 3, 4, 8, 12, 16, 20, 32, 64, 128 and 256 bytes for input or output data. The configured modules ("IO Item"in the lower table) are the basis for the signal definition.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.



The table "IO items" shows you the configured PROFINET input or output modules.

Column name	Description
Index	Shows the module index
Name	Shows the module name
Tag	Editable short name required for the signal name. The signal name addresses the data.
Description	Editable module description
Length in bytes	Shows the module length in bytes
Byte offset	Shows the start address of the module (offset in bytes) in the input and output data memory The offset is calculated automatically.

The column *Tag* is editable for each module. The *Tag* is an essential part of the signal name and named in the following characters only.

- Uppercase letters (A to Z),
- lowercase letters (a to z),
- numbers (0 to 9), and
- underscore.

Do not enter two or more underscores in succession.

To open the signal configuration of the respective module, double click any line.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

The following table lists the controls of the table *IO Items*.

Control	Description
	Moves the marked IO item (module) upward by one line. **Note:** This changes the PROFINET configuration of the Edge Gateway and has to be taken into account when configuring the PROFINET controller.
	Moves the marked IO item (module) downward by one line. **Note:** This changes the PROFINET configuration of the Edge Gateway and has to be taken into account when configuring the PROFINET controller.
	Duplicates the marked IO item (module). **Note:** This changes the PROFINET configuration of the Edge Gateway and has to be taken into account when configuring the PROFINET controller.
	Deletes the marked IO item (module). **Note:** This changes the PROFINET configuration of the Edge Gateway and has to be taken into account when configuring the PROFINET controller.
	This selection allows you to filter the IO items. • Send/Receive shows you all IO items (modules). • Send shows you all send IO items (input modules). • Receive shows you all receive IO items (output module).
	Each column offers you sort and search functions. To open the list to select from, click the column heading. To activate the filter function, enter the text in the filter input field and press the Enter key. You can use * as a wildcard for filtering. To deactivate the filter function again, delete any text entered in the filter input field and press the Enter key.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

5.3.3.1 Signal Definition Page

On this page, you can define the signals for the input and output modules of the PROFINET for the Edge Gateway. Signal names are required for the fieldbus node to address PROFINET data.

The table "IO item" provides information on the selected IO item for which you define signals on this page.

IO item

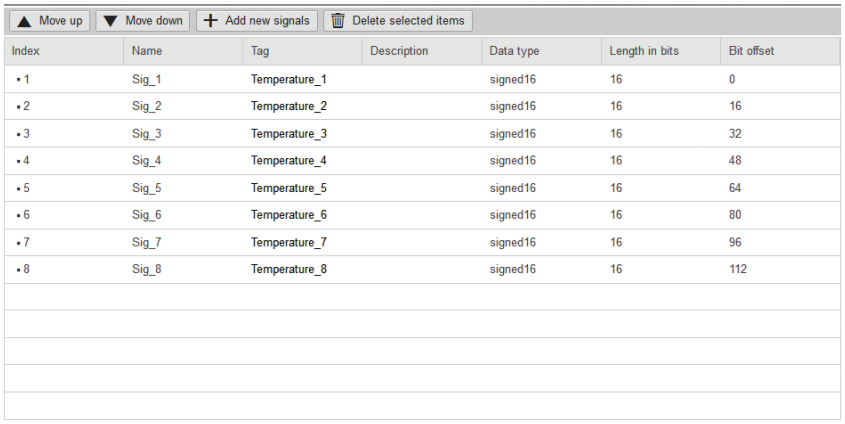
Index	Name	Tag	Length in bytes	Length in bits	Byte offset
001	16 send byte(inputs)	toController	16	128	0

Column name	Description
Index	Shows the module index.
Name	Shows the module name.
Tag	Shows the short name required for the signal name.
Length in bytes	Shows the module length in bytes.
Length in bits	Shows the module length in bits.
Byte offset	Shows the start address of the module (offset in bytes) in the input and output data memory. The offset is calculated automatically.

In the table Signals you can configure the signals for an IO item (module). The table shows you the current signal configuration and allows you to enter signal names, lengths and offset addresses.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Signals



Index	Name	Tag	Description	Data type	Length in bits	Bit offset
1	Sig_1	Temperature_1		signed16	16	0
2	Sig_2	Temperature_2		signed16	16	16
3	Sig_3	Temperature_3		signed16	16	32
4	Sig_4	Temperature_4		signed16	16	48
5	Sig_5	Temperature_5		signed16	16	64
6	Sig_6	Temperature_6		signed16	16	80
7	Sig_7	Temperature_7		signed16	16	96
8	Sig_8	Temperature_8		signed16	16	112

Column name	Description
Index	Shows the module index.
Name	Shows the module name.
Tag	Editable short name required for the signal name. The signal name addresses the data.
Description	Editable module description.
Data type	Data type of the signal. You can set the data type only when you add a new signal.
Length in bytes	Shows the module length in bytes.
Byte offset	Shows the start address of the module (offset in bytes) for a signal within a module.

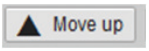
The column *Tag* is editable for each module. The *Tag* is an essential part of the signal name. The following characters are allowed only:

- Uppercase letters (AZ),
- lowercase letters (az),
- numbers (09), and
- underscore.

Do not enter two or more underscores in succession.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

The following table lists the controls of the table *Signals*.

Controls	Description
	Moves the marked signal upward by one line.
	Moves the marked signal downward by one line.
	Adds a new signal. A dialog box for selecting the data type is displayed. You can set the data type of the signal in this dialog box only.
	Deletes the marked signal.
	Each column offers you sort and search functions. To open the list to select from, click the column heading. To activate the filter function, enter the text in the filter input field and press the Enter key. You can use * as a wildcard for filtering. To deactivate the filter function again, delete any text you have entered in the filter input field and press the Enter key.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

5.3.3.2 Data Types for Signal Names

The following table lists the data types for signal names.

Name of the data type	Description	Bit length	Value range
Bit list	List of individual bits The number of bits in a list must be a multiple of 8.	1	0 (false), 1 (true)
OctetString	Character sequence	8*(n+1)	
Signed8	Short integer	8	-128 ...127
Signed 16	Integer	16	-32768 … 32767
Signed 32	Double integer	32	-2^{31}… $+2^{31}-1$
Signed 64	Long integer	64	-2^{53}… $+2^{53}-1$
Unsigned8	Unsigned short integer	8	0 … 255
Unsigned16	Unsigned integer/ Word	16	0 … 65535
Unsigned32	Unsigned double integer	32	0 … $+2^{32}-1$
Unsigned64	Unsigned long integer	64	0… $+2^{64}-1$
Real32	Floating point	32	≈ $\pm 10^{38}$
Real64	Long Float	64	≈ $\pm 10^{308}$

5.3.3.3 Structure of the Signal Names

Signal names address the fieldbus data.

General structure of the signal names

Signal names have the following structure:

- input~module_tag~signal_tag
- output~module_tag~signal_tag

For a bit list the signal names have the following structure:

- input~module_tag~bitlist_signal_tag
- output~module_tag~bitlist_signal_tag

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

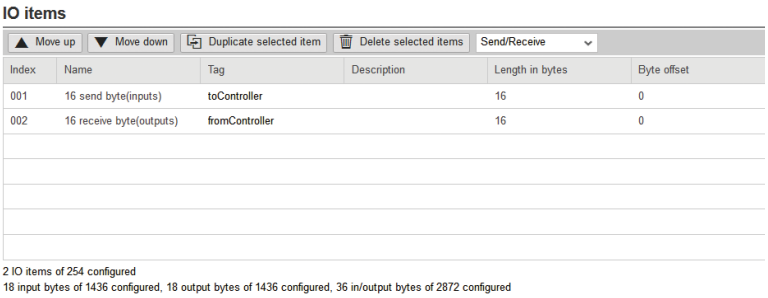
Parts of the signal name

1. The prefix input or output is produced automatically.
2. The tilde (~) separates the prefix and the module_tag.
3. You can enter the name for module_tag as described in 5.3.3.4 Configuring I/O.
4. The tilde (~) separates the module_tag and the signal_tag or the module_tag and the bitlist_signal_tag.
5. You can enter the name for signal_tag or bitlist_signal_tag as described in 5.3.3.5 Defining signals (procedure).

5.3.3.4 Configuring I/O

The upper table (*Available IO items*) displays the list of the possible (selectable) input and output modules. The lower table (*IO items*) displays the PROFINET configuration and is empty at the beginning.

1. Inserting the IO item (module):
   - ➢ To insert the required IO item into the "IO item" list, double click the IO item in the "Available IO item" list.
   - ✎ The respective IO item will be inserted at the end of the list, if no "IO item" line has been marked, or before a marked "IO item" line.



2. Entering the tag for the IO item:
   - ➢ Enter a name for each IO item in the column **Tag** using the characters A-Z, a-z, 0-9 and _ (underscore) only.
   - ✎ The name (**Tag**) is used for the signal name to address the data.

3. Entering the description of the IO item (optional):
   - ➢ In the column **Description**, you can enter a text (e.g. temperature sensor) which helps you describe the use of the data.
   - ✎ This description is used only if you print the signal list.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

4. Configuring PROFINET:
   - ➢ Repeat steps 1-3 until all required IO items (modules) are inserted into the "IO item" list.
   - ⇨ The PROFINET configuration is prepared.

5. Saving the configuration:
   - ➢ To save the PROFINET configuration in the Edge Gateway, click **Project** > **Save**.
   - ⇨ The PROFINET configuration in the Edge Gateway is saved, but not active yet.
   - ➢ To activate the PROFINET configuration, you have to click **Reconfigure** in the fieldbus node as soon as you have finished the PROFINET configuration.

5.3.3.5 Defining Signals (Procedure)

Each IO item (module) contains one or more bytes for input or output data. To enable the fieldbus node to access the input or output data, you have to select a signal at the fieldbus node. You can use the predefined signal name or define your own signal name.

The upper table (IO item) shows the IO item (module) for which you can define signals on this page. For defining signals, use the lower table (Signals).

Each IO item (module) already has a predefined signal. The predefined signal comprises and addresses the entire IO item (module). If you want to access a "part" of the IO item, you can define a new signal for this purpose.

Example 1: The IO item contains 16 bytes. If you want to access each single byte, define 16 signals of data type byte each.

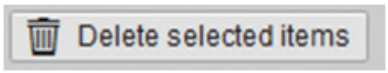
Example 2: The IO item contains 2 bytes that correspond to 16 digital input or outputs. If you want to access each single bit, define 16 signals of data type bit each.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Defining your own (new) signals

1. Selecting the IO item (module):
   - ➢ Select the IO item (module) for which you want to define signal names in the configuration tree under **IO and signal configuration**.
   - ↬ The selected IO item is displayed in the workspace.

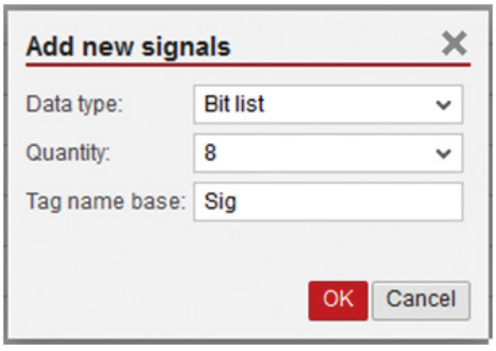
2. Deleting the predefined signal:
   - ➢ Mark the predefined signal in the table **Signals**.
   - ➢ Click **Delete selected items**.



   - ↬ The predefined signal is deleted and you can define your own signals.

3. Adding new signals:
   - ↬ Click **Add new signals**.

   - ➢ The dialog box **Add new Signals** for selecting the data type is displayed.



4. Configuring signals:
   Select the data type for the signal. The list of data types is described in0.
   - ➢ Data types for signal names.
   - ➢ Select the quantity. The possible values you can select depend on the selected data type and the number of input or output bytes to which no signal name has been assigned yet.
   - ➢ To have an identical beginning for all signal names, enter a text in the input field **Tag name base**. The extension „_1“, „_2“, etc. will be added automatically, if you have entered a value greater than 1 under "Quantity".
   - ➢ Click **OK**.
   - ⇨ The new signal/s is/are defined and will be displayed in the table Signals.

5. Modifying signals subsequently:
   - ➢ You can subsequently modify the signal names in the column **Tag** of the table Signals if required.
   - ➢ You can enter a signal description in the column **Description** of the table Signals if required.

6. Defining signals:
   - ➢ Repeat steps 1-5 until you have defined all required signals.
   - ⇨ The signal definition is prepared.

7. Saving the configuration and signal definition:
   - ➢ To save the configuration and signal definition in the Edge Gateway, click **Project** > **Save**.
   - ⇨ The configuration and signal definition in the Edge Gateway is saved, but not active yet.
   - ⇨ Click Update in the fieldbus node so that the fieldbus node reads the signal definition again.
   - ➢ To activate the configuration, click **Reconfigure** in the fieldbus node as soon as you have finished or changed the configuration.
   - ➢ The name in the column **Tag** is a part of the signal name and is described as **Signal_tag** or **Bitlist_signal_tag** in0
     Structure of the signal names.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

5.3.4 Signal Definitions Overview

This page displays the list of the defined signals in the table "Send and Receive signals". You can limit the display of the signals to send signals or receive signals. The signal names on these pages are not editable.

Send and Receive signals

Index	Name	Tag	Data type	Length in bits	Byte offset	Bit offset
001	16 send byte(inp...	toController	octetString	128	0	
1	Sig_1	Temperature_1	signed16	16		0
2	Sig_2	Temperature_2	signed16	16		16
3	Sig_3	Temperature_3	signed16	16		32
4	Sig_4	Temperature_4	signed16	16		48
5	Sig_5	Temperature_5	signed16	16		64
6	Sig_6	Temperature_6	signed16	16		80
7	Sig_7	Temperature_7	signed16	16		96
8	Sig_8	Temperature_8	signed16	16		112
002	16 receive byte(o...	fromController	octetString	128	0	
1	Sig_1	Set_temperature_1	signed16	16		0
2	Sig_2	Set_temperature_2	signed16	16		16
3	Sig_3	Set_temperature_3	signed16	16		32
4	Sig_4	Set_temperature_4	signed16	16		48
5	Sig_5	Set_temperature_5	signed16	16		64
6	Sig_6	Set_temperature_6	signed16	16		80
7	Sig_7	Set_temperature_7	signed16	16		96
8	Sig_8	Set_temperature_8	signed16	16		112

If a module contains one or more signals, these signals will be displayed under the respective module.

Column name	Description
Index	Shows the module index (3-digit) and signal (1 to 3-digit).
Name	Shows the module name or signal.
Tag	Editable short name required for the signal name. The signal name addresses the data.
Data type	Shows the data type of the module or signal.
Length in bits	Shows the length of the module or signal in bits.
Byte offset	Shows the start address (offset in bytes) for a module in the input or output data memory. The offset will be calculated automatically.
Bit offset	Shows the configured start address (bit offset) for a signal within a module.

You can print the table using **Print** > **Print configuration**.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

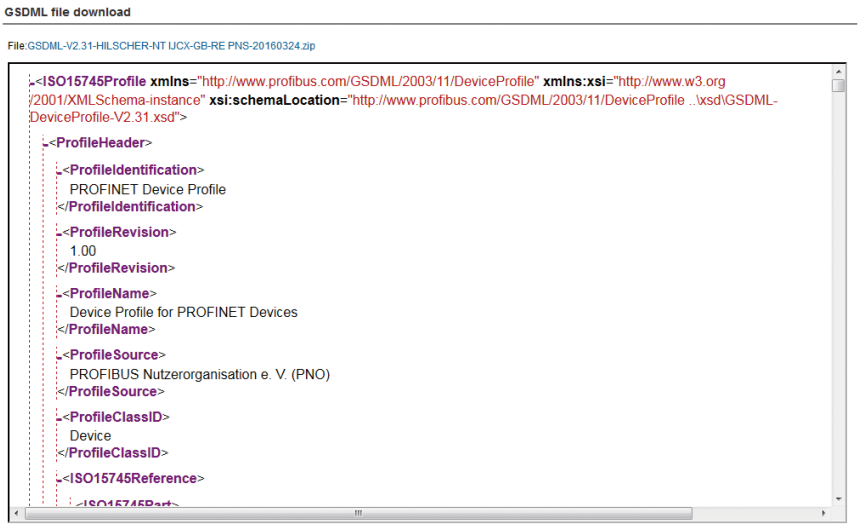
5.3.5 GSDML File Download

On this page you can

- display the contents of the GSDML file and
- save the GSDML file on your PC (download).

The GSDML file contains the PROFINET properties of the Edge Gateway and is required for configuring the PROFINET controller. Download the GSDML file from the Gateway so that you can use it in the configuration tool of the PROFINET controller.

Display of the GSDML file



The GSDML file is not editable in this window. The content of the GSDML file is represented in different colors:

- Violet: Names of elements.
- Black: Texts.
- Black and bold: Names of attributes.
- Red: Values of attributes.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Download of the GSDML file

➢ Click the file with the extension ZIP.

↬ Your PC displays a dialog for saving the GSDML file.

➢ Select a folder and click on **Save**.

➢ Use the saved GSDML file for configuring the PROFINET controller.

The job to build up your system is over. Your system should run very well if you have followed instructions in the previous chapters. The next following chapters depict each item for your reference only.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Chapter 6: System Menu

The system menu provides information of the CPS appliances and statuses, and it allows you to define the network setup as well as how you want to access. Once you have logged onto the IP address, you should see a page titled "Edge Gateway manager".

- Use the default **Username**: *admin* and **Password**: *admin* to login if you are logging for the first time.
- Remember to change username/password and keep it in a safe area to avoid hacking.

6.1 System

Click **Control Panel** to enter the system settings page.

6.1.1 Info Center

The page of **System | Info Center** will come into sight. Refer the table below for item descriptions.

Item	Description
Last update	The time of the last update from the system.
Serial number	The serial number of the system.
Model name	The model name of the system.
Firmware version	The firmware version of the system is loaded with.
System time	The status and time zone of the system is set to.
Processor name	The name of the processor equipped with the system.
CPU usage	CPU specification and consumed computing capability in percentage.
Memory utilization	Available memory size and consumed memory size in percentage.
Storage space	Available space in storage and consumed space in percentage.
CPU temperature	Current temperatures of each core in the CPU.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

6.1.2 Time

On the page of **Control Panel**, click **System** at the top of the page, and then click **Time** for time zone and NTP server settings. The page of **System | Time** will come into sight. Refer to the table below for descriptions of each item.

Item	Description
Save changes	Click to save changes to settings.
Timezone	Use the drop-down menu to select your time zone.
Manual	Tick to set time and date manually.
NTP synchronized	Tick to synchronize system time with the NTP server. Click **+ Add NTP server** or **Delete** to manage NTP server addresses.

6.2 Package Manager

6.2.1 Packages

On the page of **Control Panel**, click **Package Manager** at the top of the page, and then click **Packages** to manage packages. The page of **Package Manager | Packages** will come into sight. Refer to the table below for item descriptions.

Item	Description
Installed	Installed packages listed in the table. Click **Delete** if you wish to delete a package. Click **Refresh** to renew the package list.
Available	Click Explorer to launch your file explorer and import package files. Select available packages in the list and click **Install** to add packages to the system. Click **Clear** to remove packages from the list.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

6.3 Network

6.3.1 LAN

On the page of **Control Panel**, click **Network** at the top of the page, and then click **LAN** for information of local network settings. The page of **Network | LAN** will look like the table below.

Refresh	Save Changes		
Name	**MAC-Address**	**Settings**	**DNS**
LAN1(eth0)	xx.xx.xx.xx.xx.xx	• DHCP (default) • Fixed address	• DNS server • DNS server
LAN2(eth1)	xx.xx.xx.xx.xx.xx	• DHCP • Fixed address (default) • IP: 192.168.253.1 • Netmask: 255.255.255.0 • Gateway:	• DNS server • DNS server

Refer to the table below for item descriptions.

Item	Description
Refresh	Click to renew the status of local network settings.
Save changes	Click to save changes to local network settings.
Name	Title of the onboard network adapter, e.g. **eth0** denotes network adapter 1, **eth1** denotes Network adapter 2, and **cifx0** denotes PROFINET or EtherNet/IP. • You can check the IP address assigned to eth0 by DHCP server if you are connected with 1. • You can change settings of **cifix0** once PROFINET or EtherNet/IP is available in your CPS 200 or CPS 100.
MAC address	Physical address of the network adapter.
Settings	The IP address assigned to the network adapter by either DHCP or manual input.
Domain Name System	The domain name server list the network adapter applies to resolve addresses.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

6.3.2 Wi-Fi

On the page of **Control Panel**, click **Network** at the top of the page, and then click **Wi-Fi** for wireless network settings. The page of **Network | Wi-Fi** will come into sight. Refer to the table below for item descriptions. (This is an optional module).

⇨ An error message will pop up if wireless network is not available in the system.

Item	Description
Save changes	Click to save changes to settings.
Operating mode	Use the drop-down menu below to change modes of the wireless network. Select **Access point** to set up wireless networking. Select **Client** to manage client connections.
Name	Name of the wireless network adapter.
MAC address	Physical address of the wireless network adapter.

6.3.3 Hostname

On the page of **Control Panel**, click **Network** at the top of the page, and then click **Hostname** to set up system hostname. The page of **Network | Hostname** will come into sight. Refer to the table below for item descriptions.

Item	Description
Refresh	Click to refresh the hostname from the system.
Save changes	You can input a desired name in the field of **Hostname**, and press **Save changes** to keep it.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

6.4 Services

6.4.1 Service List

On the page of **Control Panel**, click **Services** at the top of the page, and then click **Service List** to manage network protocol settings. The page of **Services | Service List** contains two services, Node-Red and MQTT Broker now. To change settings, click on the item in the service list at left of the page. Refer to the table below for item descriptions.

Item	Description
Operating status	The status of the service You can click **Stop** or **Start** to disable or enable the service.
Autostart	Tick **enabled** or **disabled** to set the service to start automatically as the system starts or not. Click **Apply** to keep the setting.
Current flow	Current flow can be downloaded, uploaded deleted or undo deploy.

6.5 User Management

6.5.1 User Accounts

On the page of **Control Panel**, click **User Management** at the top of the page, and then click **Accounts** to manage accounts of the system. The page of **User Management | Accounts** will come into sight. Refer to the table below for item descriptions.

Item	Description
Create new user account	Click to create a new user account.
Edit user account	Click to edit the user account of the selected username in the list.
Delete user account	Click to delete the user account of the selected username in the list.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

6.5.2 Roles

On the page of **Control Panel**, click **User Management** at the top of the page, and then click **Roles** to control privileges of accounts in the system. The page of **User Management | Roles** will come into sight. Refer to the table below for item descriptions.

Item	Description
Create new role	Click to create a new privilege role.
Delete role	Click to delete a privilege role.
Save changes	You can select a role in the role list, and change privileges of the role by ticking **None**, **Read**, or **Read & Write** in each resource. Click **Save changes** to keep settings.

6.6 Security

6.6.1 SSL Certificate

On the page of **Control Panel**, click **Security** at the top of the page, and then click **SSL Certificate** to update certificates of the system. The page of **Security | SSL Certificate** will come into sight. Refer to the table below for item descriptions.

Item	Description
Certificate	Click Browse to select your certificate file.
Private Key	Click Browse to select your private key file.
Upload and install certificates	Click to upload and install the selected files to your system.
Certificate information	Details of your certificate.
Refresh	Click to refresh certificate information.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

6.7 Help

6.7.1 Info

On the page of **Control Panel**, click **Help** at the top of the page, and then click **Info** to view the version number of the system. The page of **Help | Info** will come into sight. Refer to the table below for item descriptions.

Item	Description
Version	The version of the system

6.8 Session

6.8.1 User profile

On the page of **Control Panel**, click **Session** at the top of the page, and then click **User Profile** to view the information of the current logged in user. The page of **Session | User Profile** will come into sight. Click Edit user account to change the password of the current logged in user.

6.8.2 Logout

On the page of **Control Panel**, click **Session** at the top of the page, and then click **Logout** to exit the system.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Chapter 7: IoT Studio Menu

7.1 Input Nodes

7.1.1 inject

Press the button to the left of the node allows a message on a topic to be injected into the flow.

Payload	**flow**	The flow name
	global	The global variable
	string	The string (character type)
	number	The number
	boolean	false/true
	JSON	The json format
	timestamp	The current time in milliseconds since 1970
Topic		The name of the node
Repeat	**Interval/Interval between times/ at a specific time**	The repeat function allows the payload to be sent on the required schedule.
	Inject once at start?	The inject once at start option actually waits a short interval before firing to give other nodes a chance to instantiate properly.
Name		The name of node

Note:

- "Interval between times" and "at a specific time" uses cron job(crontab). This means that 20 minutes will be at the next hour, 20 minutes past and 40 minutes past, not in 20 minutes time. If you want every 20 minutes from now, use the "interval" option.
- All string input is escaped. To add a carriage return to a string you should use a following function.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.1.2 catch

The catch node catches errors thrown by nodes on the same tab. If a node throws an error whilst handling a message, the flow will typically halt. This node can be used to catch those errors and handle them with a dedicated flow. The node will catch errors thrown by and node on the same tab. If there are multiple catch nodes on a tab, all will get triggered.

If an error is thrown within a subflow, any catch nodes within the sub flow will handle the error. If none exists, the error is propagated up to the tab the subflow instance is on.

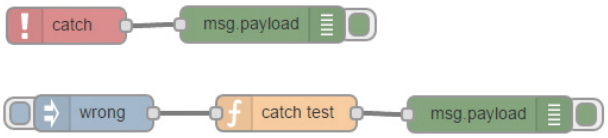
The message sent by this node will be the original message if the node that threw the error provided it. The message will have an error property with the following attributes:

message: the error message
source.id: the id of the node that threw the error
source.type: the type of the node that threw the error
source.name: the name, if set, of the node that threw the error

If the message already had an error property, it is copied to error.

Example: Use catch node to problematic nodes.

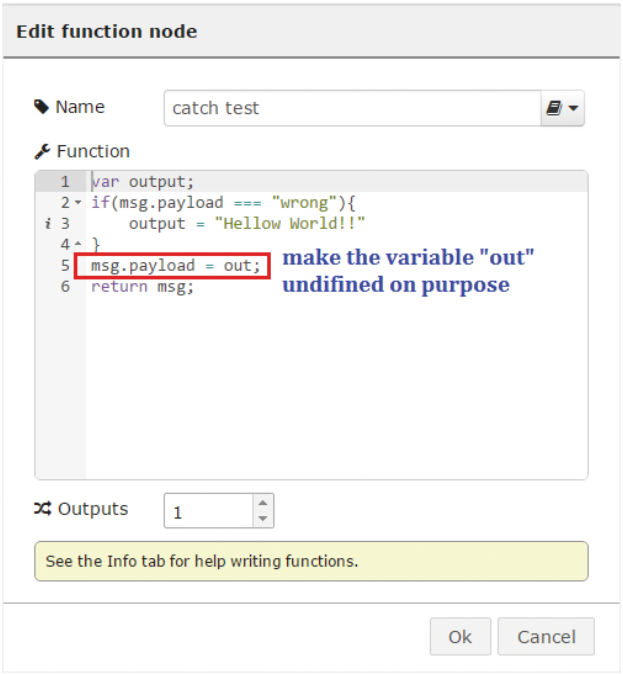
1. Add an **inject** node, a **function** node, a **catch** node, and two **debug** nodes to the workspace as shown.



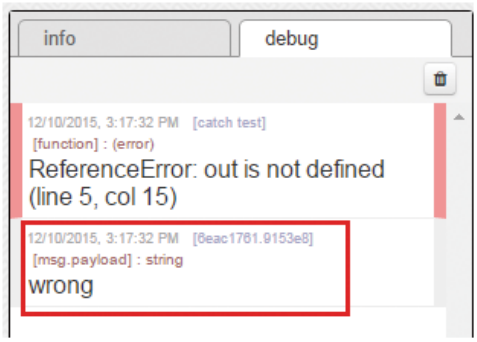
2. Edit the **inject** node by setting msg.payload to be a string "wrong" and click **Done**.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

3. Edit function node as below and click **Ok**. Notice that there is an error in the code on purpose.



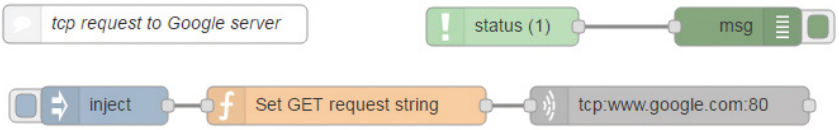
4. Deploy your flow and click the button to the left of the inject node, and user shall see the debug as below. The message in red is the message that the catch node throws to indicate there is a problem with "wrong" node.



7.1.3 status

The status node sends status message from other nodes on the same tab.

Example: Get the URL www.google.com status.

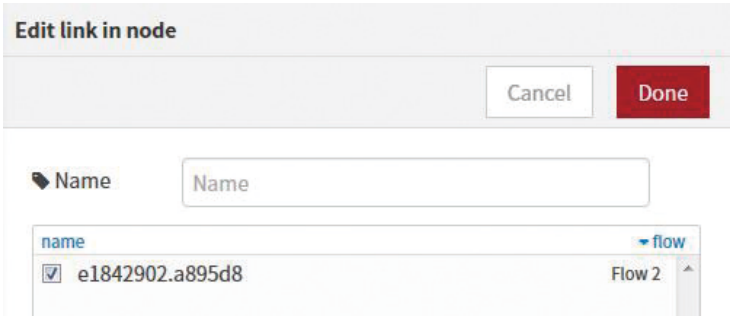


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.1.4 link

The link input node can be connected to any link out node that exists on any tab. Once connected, they behave as if they were wired together. The wires between link nodes are only displayed when a link node is selected. If there are any wires to other tabs, a virtual node is shown that can be clicked on to jump to the appropriate tab. Links cannot be created going into, or out of, a subflow.

Click on the node to select which link out node it will connect to. Check the box to select, and click **Done** when finished.



7.1.5 mqtt

The mqttinput node connects to a broker and subscribes to the specified topic. The topic may contain MQTT wildcards. Outputs and object called msg containing:

msg.topic
msg.payload
msg.qos
msg.retain

msg.payload is usually a string, but can be a binary buffer.

Server	**Server**	The URL or the IP address of MQTT broker.
	Port	The network port listening to publish/ subscribe requests with the default value of 1883.
	Client ID	With the unique Client ID the broker can recognize when a client reconnects and close an old potentially half-open TCP connection for the client.
	Username/ Password	The broker refuses the anonymous connection by default setting “allow_ anonymous false”.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Topic		The string used by the broker to filter messages. A topic consists of one or more topic levels. Each topic level is separated by a forward slash (topic level separator).
QoS		The Quality of Service (QoS) level is an agreement between sender and receiver of a message regarding the guarantees of delivering a message.
	0 (default)	The message is delivered at most once, or it is not delivered at all. Its delivery across the network is not acknowledged. The message is not stored. The message might be lost if the client is disconnected, or if the MQTT broker fails.
	1	The message is always delivered at least once. If the sender does not receive an acknowledgement, the message is sent again with the DUP flag set until an acknowledgement is received. As a result, the receiver can be sent the same message multiple times, and might process it multiple times.
	2	The message is always delivered exactly once. The message must be stored locally at the sender and receiver until it is processed.
Name		The name of node.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.1.6 http

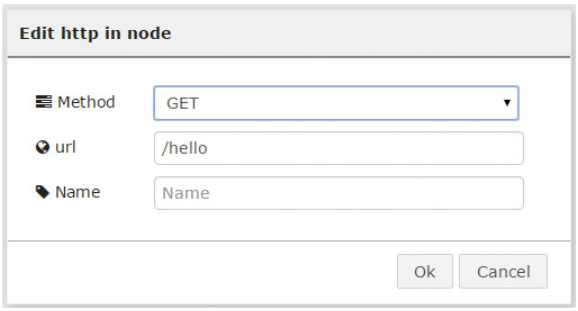
The http input node allows the creation of simple web services. This node does not send any response to the http request. This should be done with a subsequent HTTP Response node.

Method	**GET/POST/PUT/DELETE/PATCH**
URL	

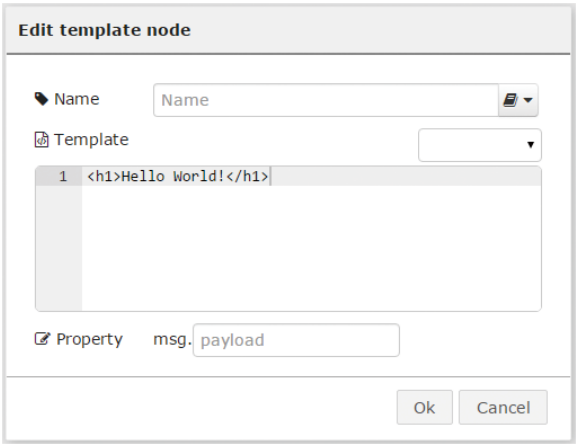
Example: Create a HTTP request and return a page with "Hello World!"

1. Add http, template and http response nodes to the workspace and link them as shown.

2. Edit http node, set **url** to "/hello" and click **Ok**.



3. Edit template node, add "<h1>Hello World!</h1>" in template node and click **Ok**.



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

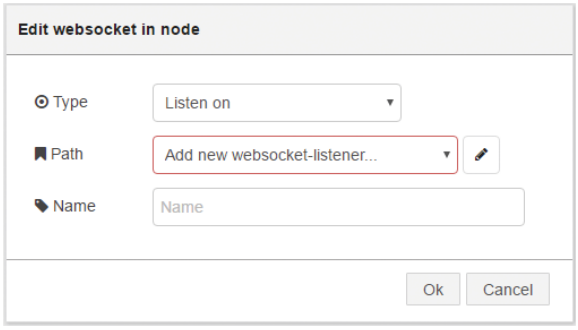
4. Confirm the changes on the nodes shown at right. Now, deploy the flow.

5. Open a new tab on local web browser with the address http://device IP/iiotStudio/hello, and result will be like this:

7.1.7 websocket

The websocket input node provides a duplex TCP connection and where designed to allow web browsers and servers to maintain a "backchannel" that could be used to augment traditional HTTP interactions, allowing servers to update web pages without the client making a new pull request.

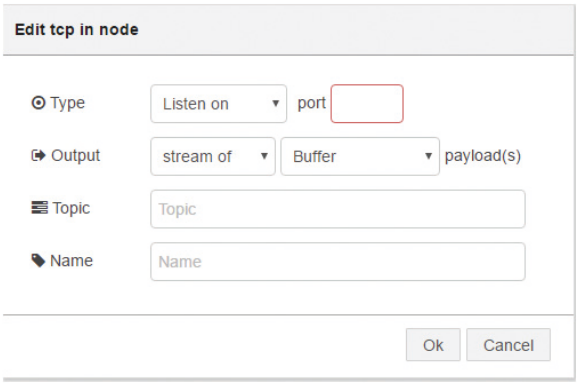
It comes in two flavors, input and output, allowing user to listen for incoming data or to send on a websocket. The output version is designed to check to see if the output payload originated at a websocket in a node, in which case it responds to the original sender. Otherwise, it will broadcast the payload to all connected websockets. Furthermore, both input and output websocket nodes can be configured as either server listening on a URL or client connecting to a specified IP address.



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.1.8 tcp

The tcpinput node is used to accept incoming TCP requests on a specified port or to connect to a remote TCP port.

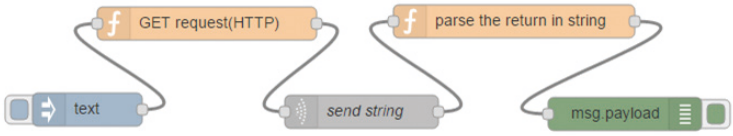


TYPE	**Listen on/ Connect to**	The heart of publish/subscribe protocol.
	Port	The service port number.
Output	**Stream of/ Single**	The consecutive data structure/ single output.
	Buffer/String/ Based64 String	The custom data structure/the string/compressed data string.

The example below shows you how to send TCP requests using the tcp node. In this case, you will make an HTTP request following the specifications in (http://tools.ietf.org/html/rfc2616#section-5.1.2).

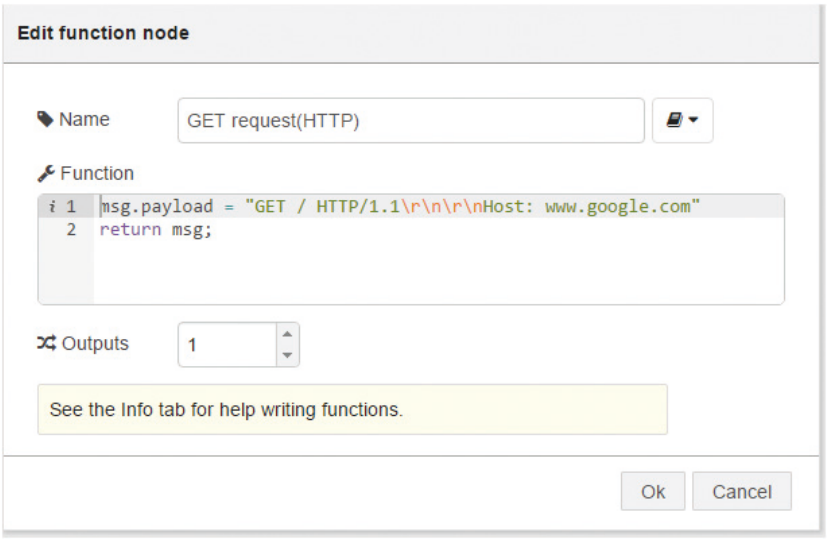
Example: Send TCP requests

1. Connect an **inject** node, a **function** node, a **tcp request** node, and a **debug** node as shown.

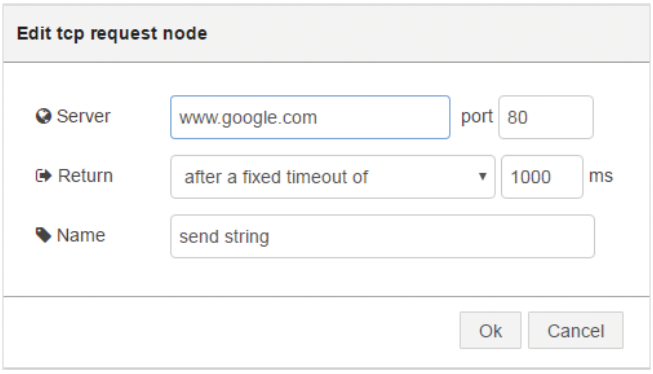


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

2. Edit the first function node to add a function that sets the string "GET / HTTP/1.1\r\n\r\nHost: www.google.com" as payload. This string is a standard HTTP request, indicating that it is a GET request, the protocol is HTTP 1.1, and the host is www.google.com. The \r\n\r\n is two pairs of return/newline required in the HTTP protocol.

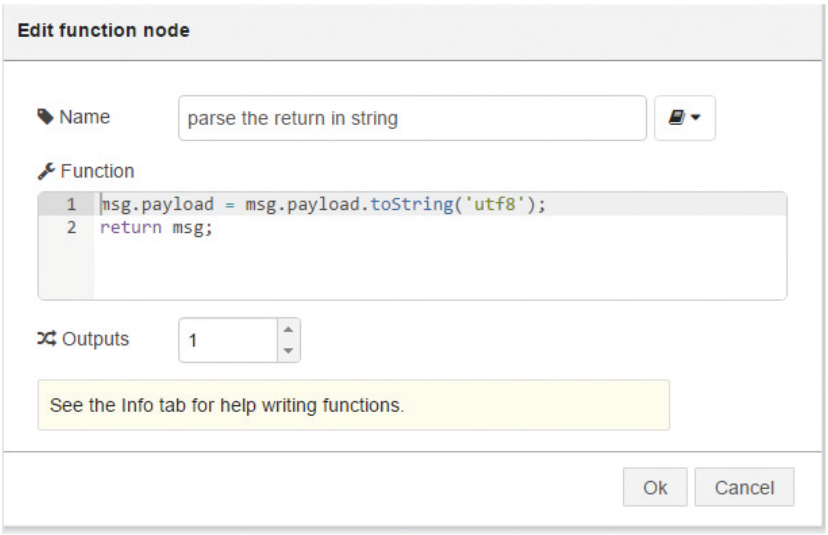


3. Configure the tcp request node to connect to the www.google.com server, on port 80. Configure it to close the connection after 1 second (1000 ms) as shown.

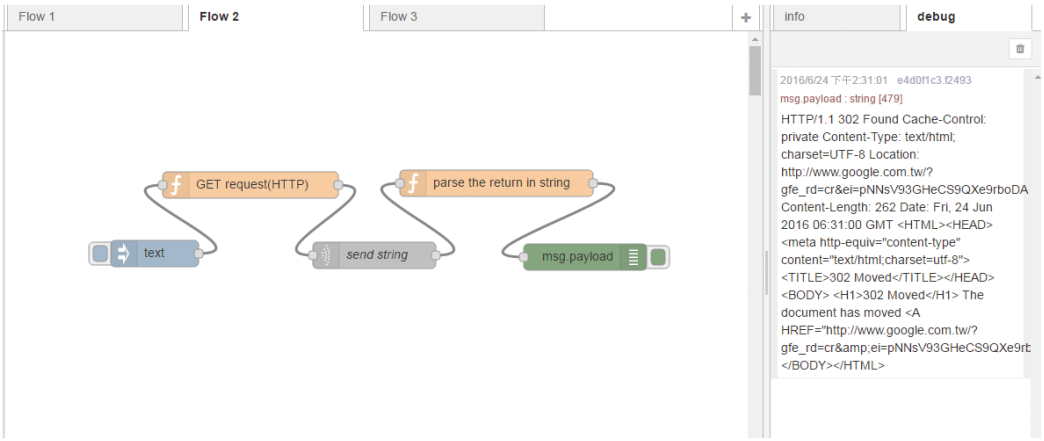


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

4. The tcp request node response is a buffer and needs to be parsed. Configure the second function node to parse the tcp request node response as shown.



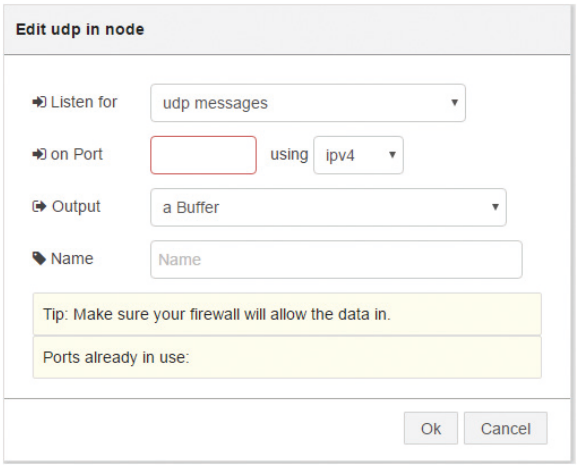
5. If you deploy the flow and click on inject, you will make a request to Google and will get a TCP response. The debug node will print the response as a string as shown.



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.1.9 udp

The udp input node is used to accept incoming UDP packets (or multicast packets) on a specified port.



Listen for	**UDP messages/ multicast messages**	The heart of publish/subscribe protocol.
	Port	The service port number.
	Using ipv4/ipv6	
Output	**Buffer/String/ Based64 String**	The custom data structure/the string/compressed data string.

7.1.10 fieldbus

Refer to Chapter 5 Fieldbus Configuration for details.

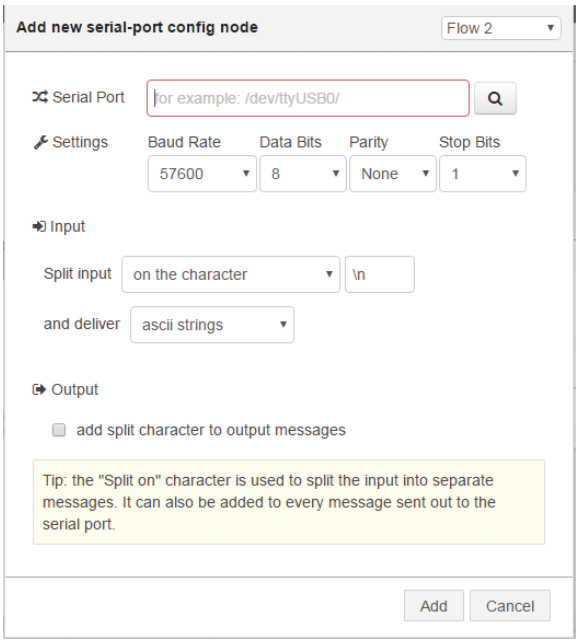
7.1.11 opc.ua

Use this node to read items from an OPC UA Server. Configure your endpoint and select a specific variable over the integrated browse function. The value is written as json in msg.payload.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.1.12 serial

The serial input node reads from a serial port on the local device. It can wait for a "split" character (default \n). Also accepts hex notation (0x0a), wait for a timeout in milliseconds for the first character received, or wait fill a fixed sized buffer.

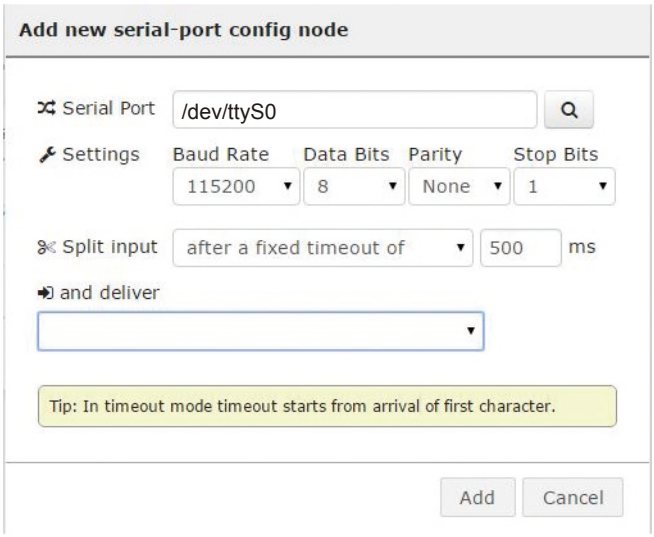


Next, it outputs **msg.payload** as either a UTF8 ASCII string or a binary Buffer object. If no split character is specified, or a timeout or buffer size of 0, then a stream of single characters is sent, again either as ASCII chars or size 1 binary buffers.

Example: Use RS232 COM port to connect with a device or a PC

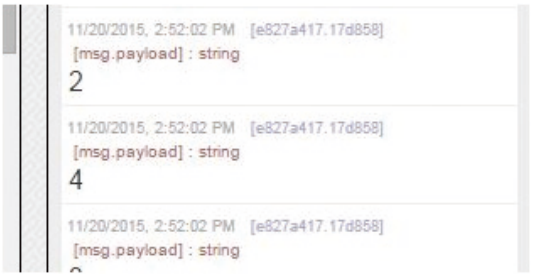
1. Add and connect a **serial** input node and a debug node to the workspace and configure the **serial** input node as shown, then click **Ok**.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.



Note: COM1 is using /dev/ttyS0

2. **Deploy** your flow and any message received will show up in the **debug** tab.



7.2 Output Nodes

7.2.1 debug

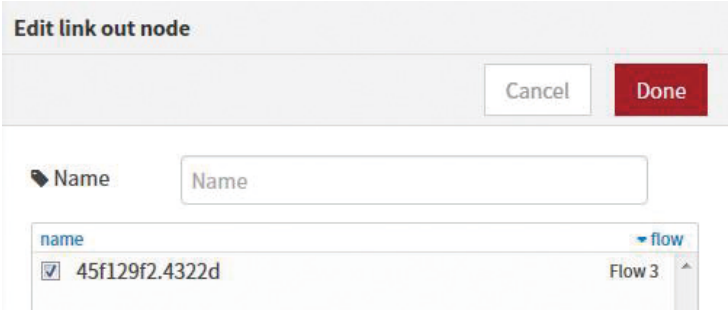
The debug node can connect to the output of any node displaying any message property in the debug tab of the sidebar. The default is to display msg.payload. Each message will also display the timestamp, msg.topic, and the property chosen to output. Access the sidebar under the options drop-down in the top right corner. The button to the right of the node will toggle its output on and off so you can de-clutter the debug window. If the payload is an object or buffer, it will be stringified first for display and indicate that by saying "(Object)" or "(Buffer)". Selecting any particular message will highlight (in red) the debug node that reported it, which is useful if you wire up multiple debug nodes. Other than optionally show the complete msg object, any calls to node. warn or node.error will appear here in the debug node.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.2.2 link

The link output node can be connected to any link in node that exists on any tab. Once connected, they behave as if they were wired together. The wires between link nodes are only displayed when a link node is selected. If there are any wires to other tabs, a virtual node is shown that can be clicked on to jump to the appropriate tab. Links cannot be created going into, or out of, a subflow.

Click on the node to select which link out node it will connect to. Check the box to select, and click **Done** when finished.



7.2.3 mqtt

The mqtt output node connects to a MQTT broker and publishes msg.payload either to the msg.topic or to the topic specified in the edit window. The value in the edit window has precedence. Likewise QoS and/or retain values in the edit panel will overwrite msg.qos and msg.retain properties. If nothing is set, the default value is 0 and false respectively. If msg.payload contains an object, it will be converted into a string before being sent.

Server	**Server**	The URL or the IP address of MQTT broker.
	Port	The network port listening to publish/ subscribe requests with the default value of 1883.
	Client ID	With the unique Client ID the broker can recognize when a client reconnects and close an old potentially half-open TCP connection for the client.
	Username/ Password	The broker refuses the anonymous connection by default setting “allow_ anonymous false”.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Topic		The string used by the broker to filter messages. A topic consists of one or more topic levels. Each topic level is separated by a forward slash (topic level separator).
QoS		The Quality of Service (QoS) level is an agreement between sender and receiver of a message regarding the guarantees of delivering a message.
	0 (default)	The message is delivered at most once, or it is not delivered at all. Its delivery across the network is not acknowledged. The message is not stored. The message might be lost if the client is disconnected, or if the MQTT broker fails.
	1	The message is always delivered at least once. If the sender does not receive an acknowledgement, the message is sent again with the DUP flag set until an acknowledgement is received. As a result, the receiver can be sent the same message multiple times, and might process it multiple times.
	2	The message is always delivered exactly once. The message must be stored locally at the sender and receiver until it is processed.
Retain		A retained message is a normal MQTT message with the retained flag set to true. The broker will store the last retained message and the corresponding QoS for that topic. Each client that subscribes to a topic pattern, which matches the topic of the retained message, will receive the message immediately after subscribing. The broker will store one retained message only for each topic.
Name		The Name of the node.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.2.4 http response

The http response output node sends responses back to http requests received from an http input node.

7.2.5 websocket

By default, msg.payload will be sent over the websocket. The socket can be configured to encode the entire msg object as a JSON string and send that over the websocket. If the message arriving at this node started at a websocket input node, the message will be sent back to the client that triggered the flow. Otherwise, the message will be broadcasted to all connected clients. If you want to broadcast a message that started at a websocket input node, you should delete the msg._session property within the flow.

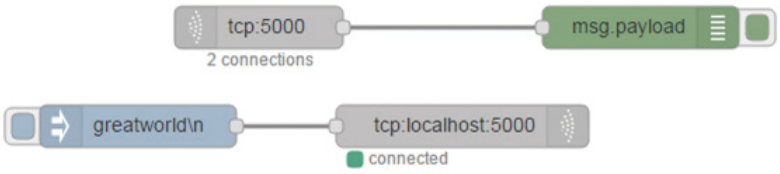
7.2.6 tcp

The tcp output node provides a choice of TCP outputs to connect to a remote TCP port, accept incoming connections, or reply to message received from the tcp input node.

The example blow shows you how to send TCP requests using the tcp node. In this case, user can make a TCP connection to target unit/port for sending out the data, structure of data and not expect the response data sent back by target unit.

Example: Send TCP requests

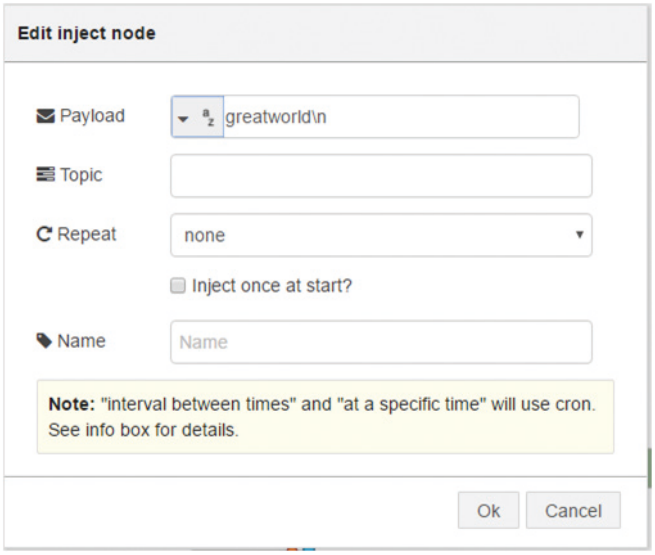
1. Connect an **inject** node, a **tcp** output node, a **tcp** input node, and a **debug** node as shown.

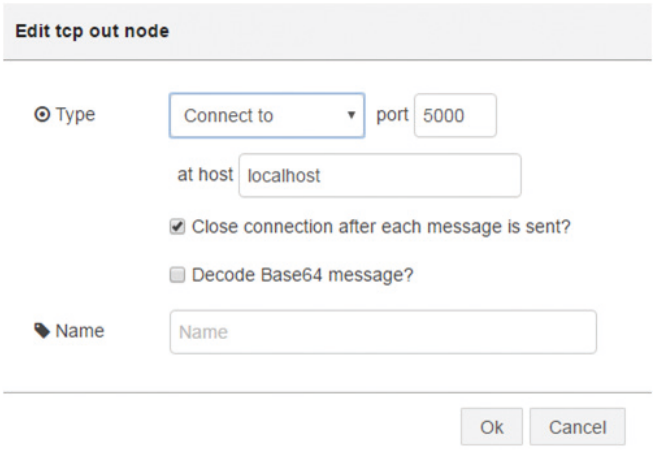


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

2. Edit the **inject** node to add the string, called "greatworld\n".

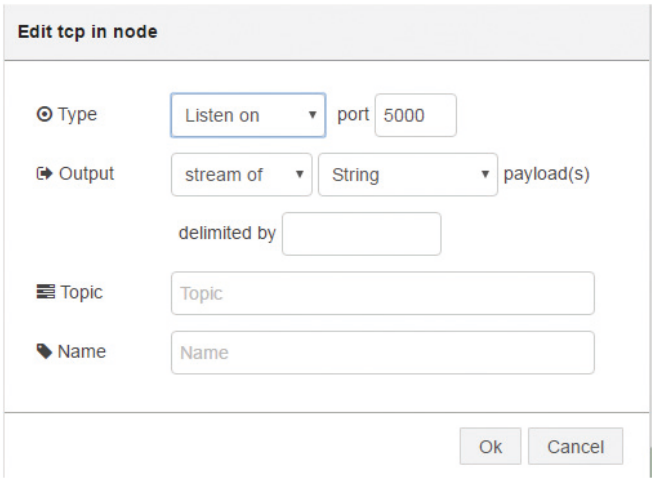
   Edit the **tcp** output node to set type to "connect to", port to "5000", and at host to "localhost". Tick the box before "Close connection after each message is sent?"





Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

3. Edit the **tcp** input node and select "listen on" and set the port to "5000".



4. Check the result "greatworld\n" on debug tab.

7.2.7 udp

The udp output node sends msg.payload to the designated UDP host and port and supports multicast.You may also use msg.ip and msg.port to set the destination values, but the statically configured values have precedence. Either set the address to the local broadcast IP address or 255.255.255.255, the global broadcast address, for broadcast.

Note: On some systems, you may need to be root to use ports below 1024 and/or broadcast.

7.2.8 fieldbus

Refer to Chapter 5 Fieldbus Configuration for details.

7.2.9 opc ua

Use this node to write items to an OPC UA Server. Configure your endpoint and select a specific variable over the integrated browse function. The value is written as json in msg.payload.

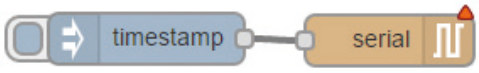
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

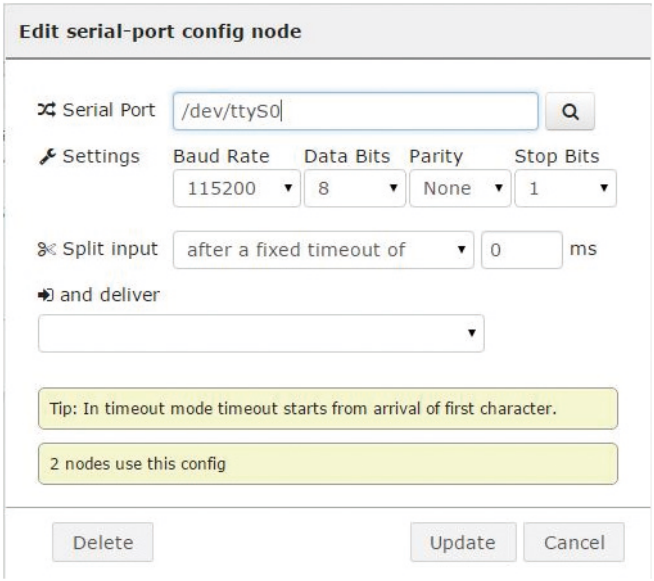
7.2.10 serial

The serial output port provides a connection to an outbound serial port. Only the **msg.payload** is sent. The new line character used to split the input can be appended to every message sent out to the serial port optionally.

Example: Use RS232 COM port to connect with a device or a PC

1. Add **serial (out)** node to the workspace and configure it as shown below, and then click **OK**.





2. **Deploy** your flow, and it will send messages to the target device.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.3 Function Nodes

7.3.1 function

The function node is a versatile utility that you can use when there is no existing node dedicated to the task. It is great for doing specialized data processing or formatting. As the name implies, a function node exposes a single JavaScript function. Using the function node, your JavaScript code can run against the messages passed in the returns zero or more messages to downstream nodes for further processing.

7.3.2 template

The template node is able to create a new message based on the provided template.

Format	**Mustache template**	Mustache is a simple web template system, described as a "logic-less" system because it lacks any explicit control flow statements, like "if and else" conditions or "for loops"; however, both looping and conditional evaluation can be achieved using section tags processing lists and lambdas. Furthermore, it is named "Mustache" because of heavy use of curly braces, { }, that resemble a sideways moustache.
	Plain text	The plain text shows the raw content in char.

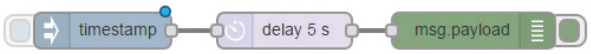
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.3.3 delay

The delay node introduces a delay into a flow or rate limits messages. Default delay time is 5 seconds and rate limit of 1msg/second, but both can be configured.

Example: Write a message "*Hello World!!*" and make it to delay for 5 seconds

1. Connect an **inject** node, a **delay** node, and a **debug** node to the workspace as shown.



2. Edit the inject node to set the payload to string "Hello World!!" and click **Ok**.

3. Deploy the flow and click the button to the left of the inject node, and the message will come after 5 seconds in debug window.

7.3.4 trigger

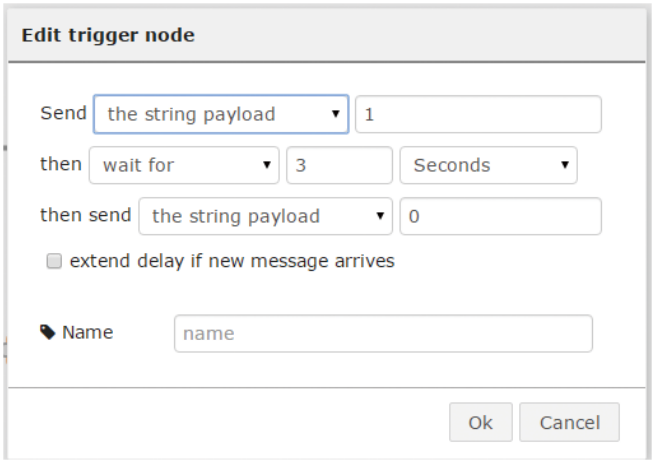
The triggernode creates two messages on the output separated by a timeout whenever ANY message arrives on the input. The two output states can be specified as the duration of the timer, and either one can be set to a value or a template from the inbound message using mustache syntax, "the payload is {{payload}}".

Example: Write a trigger

1. Connect an **inject** node, a **trigger** node, and a **debug** nodes to the workspace as shown.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

2. Double click the trigger node and set the time to 3 seconds.



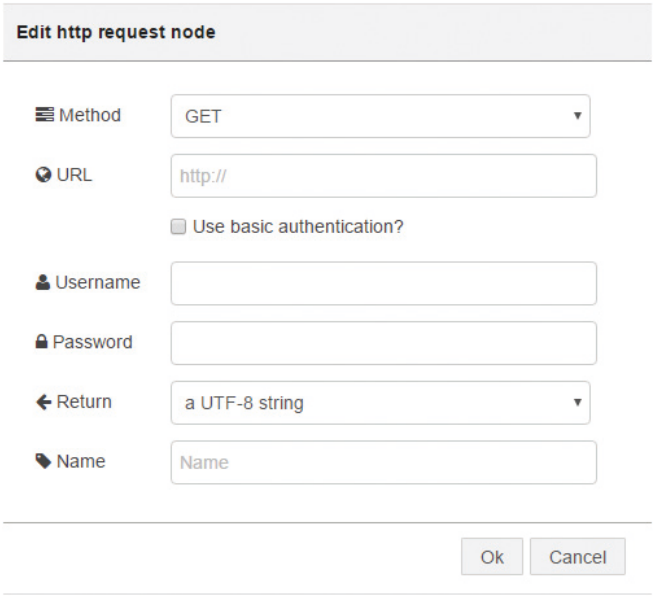
3. Deploy the flow and click the button to the left of the inject node, and two messages shows up in debug window, the value 1 shows up and value 0 comes after in 3 seconds.

7.3.5 comment

The comment adds simple description or documentation about nodes or flow. Anything you write in the **Body** will be rendered in the info tab.

7.3.6 http request

The http request provides a node for making http request.



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

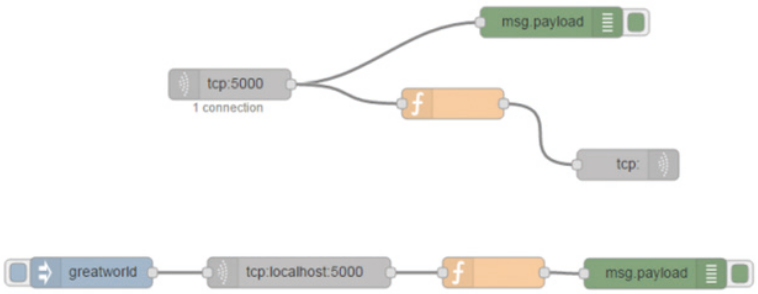
7.3.7 tcp request

The **tcp request** node sends the msg.payload to a server tcp port and expects a response. Connect, sends the "request", and reads the "response". It can either count a number of returned characters into a fixed buffer, match a specified character before returning, wait a fixed timeout from first reply and then return, or just sit and wait for data.

The example below shows you how to send TCP requests using the tcp node. In this case, user can make a TCP connection to target unit/port for sending out the data and the structure of data as well as expect the response data sent back by target unit.

Example: Send TCP requests and expects

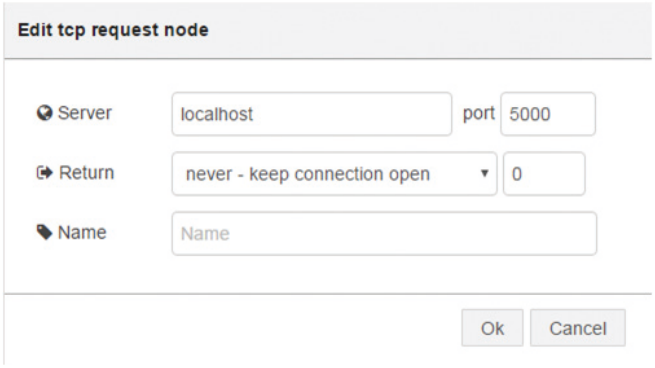
1. Connect an **inject** node, a **tcp request** node, a **tcp** input node, a **tcp** output node, a **function** node, and a **debug** node as shown.



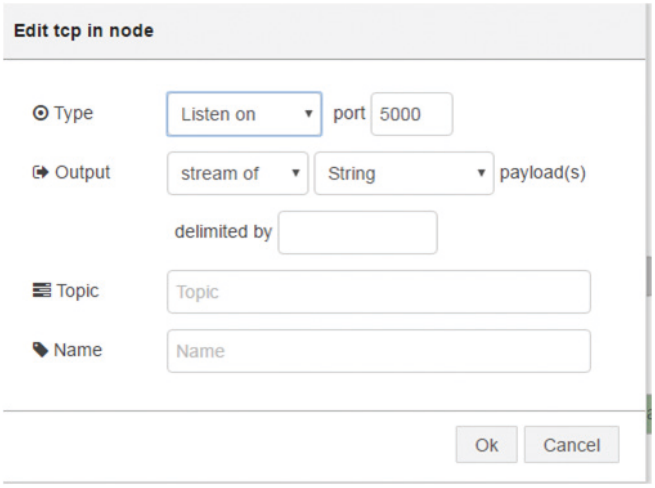
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

2. Edit the **inject** node to add the string "greatworld\n".

   Edit the **tcp request** node to connect to "localhost" at port "5000" and choose "never – keep connection open".



3. Edit the **input.tcp** node to choose "Listen on" at port "5000" and "stream of" with "String".



7.3.8 switch

The switch routes messages based on its properties. When a message arrives, the selected property is evaluated against each of the pre-defined rules. The message is then sent to the output of all rules that pass.

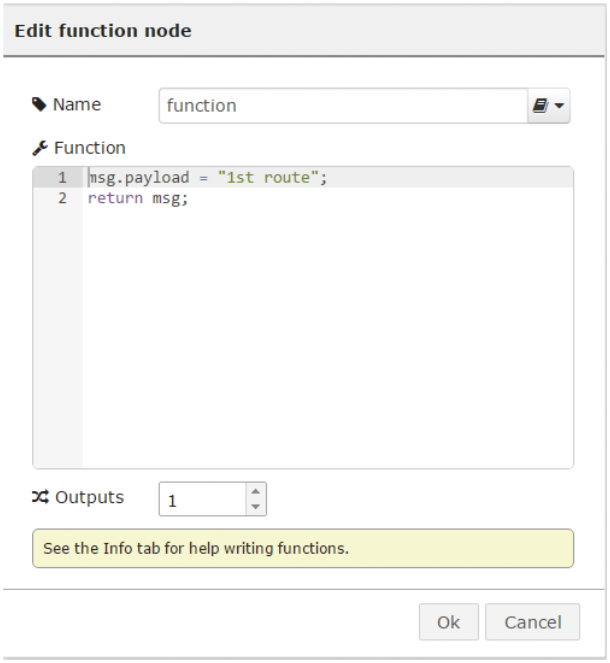
Note: the otherwise rule applies as a "not any of" the rules preceding it.

Example: Make a switch to determine if the message received matches with the rule, go first route, otherwise go second route.

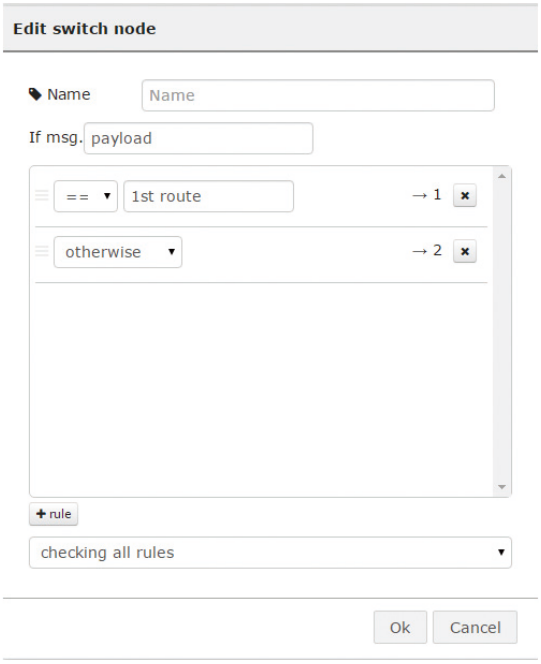
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

1. Connect an **inject** node, a **function** node, a **switch** node, and a **debug** node to the workspace as shown.

2. Edit the **function** node as shown and click **Ok**.



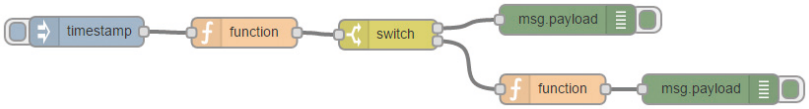
3. Edit the **switch** node as shown, and click **Ok**.



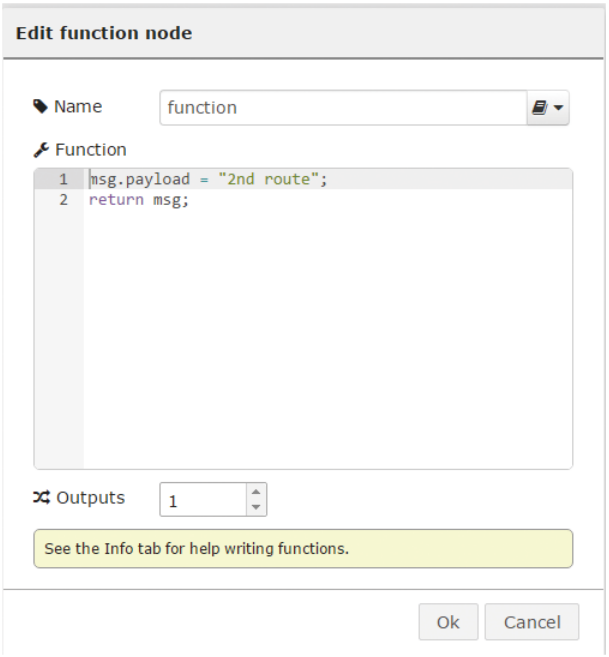
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Note: If the msg.payload is equal to "1st route", go first route; otherwise, go second route. Notice here, after you click **Ok**, the **switch** node now has two ports for you to connect two routes.

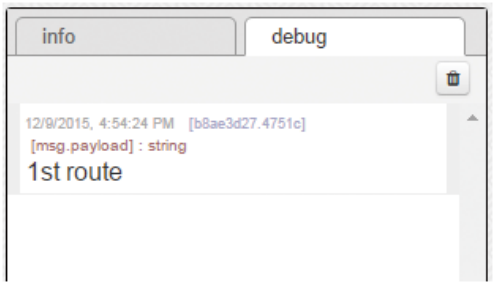
4. Add a **function** node and a **debug** node to the workspace and connect them as shown.



5. Edit the second **function** node as shown and click **Ok**.



6. Deploy the flow and click the button to the left of the **inject** node, and the result will be shown in debug tab.



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

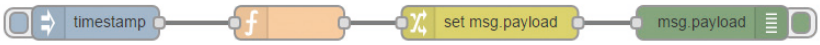
7.3.9 change

The change node sets, changes, or deletes properties of a message. It can specify multiple rules that apply to the message in turn.

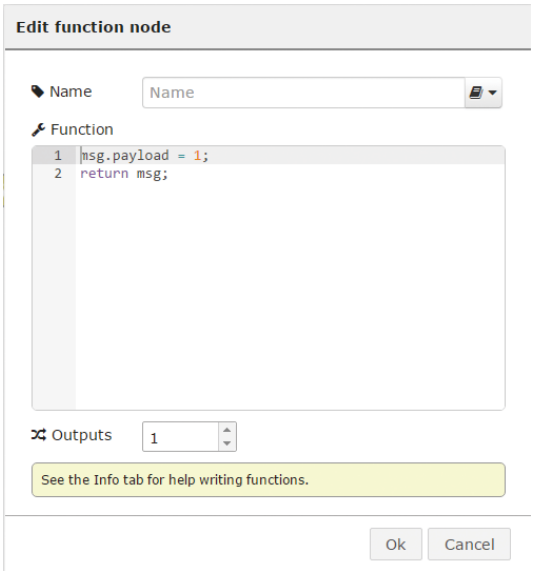
Rule	**Set**	Set a property. The target property can either be a string value or reference another message property by name, for example: **msg.topic**
	Change	Search and replace parts of the property. If regular expressions are enabled, the replace with property can include capture groups, for example $1.
	Delete	Delete a property
	Move	Move or rename a property.

Example: Change a value message from 1 to 2

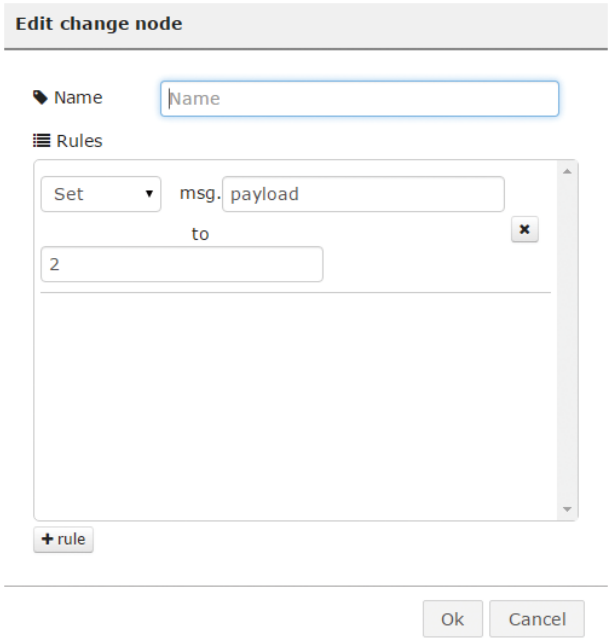
1. Connect an **inject** node, a **function** node, a **change** node, and a **debug** node to the workspace as shown.



2. Edit the **function** node as below and click **Ok**.



3. Edit the **change** node as shown and click **Ok**.



4. Deploy the flow and click the button to the left of the **inject** node, and the result will be shown in the debug tab.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.3.10 range

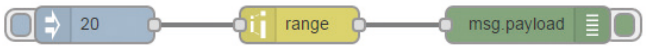
The **range** node remaps numeric input values to another scale linearly.

Note: This only operates on numbers. Anything else will be converted into a number and rejected if fails.

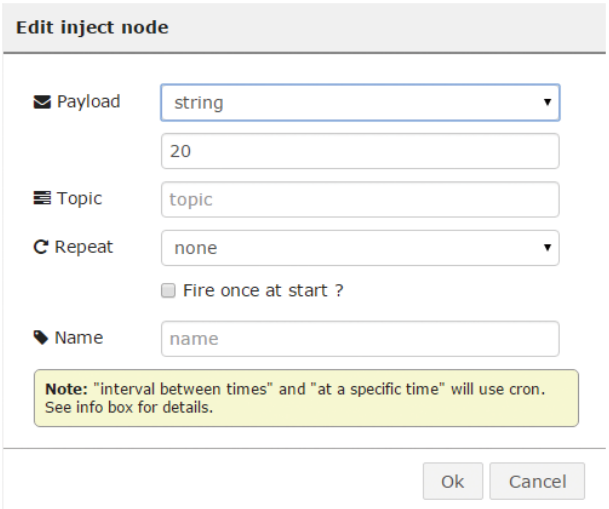
Action	**Scale and limit to target range**	The result will never be outside the range specified within the result range.
	Scale and wrap within the target range	The result will essentially be a "modulo-style" wrap-around within the result range.

Example: Map value 20 in the range of 10 to 50 to another value in the scale of 1 to 5.

1. Connect an **inject** node, a **range** node, and a **debug** node to the workspace as shown.

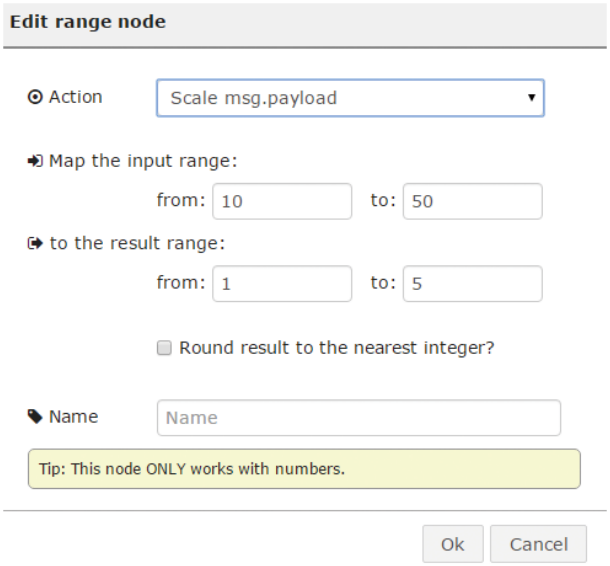


2. Edit the **inject** node as shown and click **Ok**.



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

3. Edit the **range** node as shown and click **Ok**.



4. Deploy the flow and click the button to the left of the inject node, and the result will be shown in the debug tab.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.3.11 split

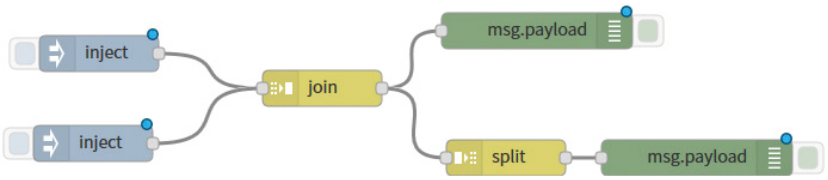
The split node splits an input into multiple outputs based on the provided configuration.

7.3.12 join

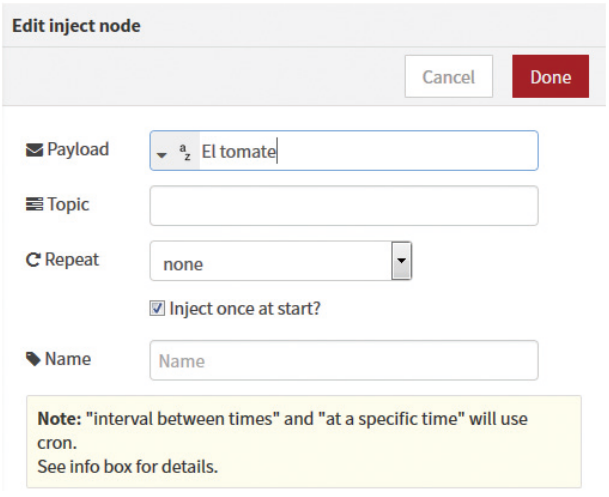
The join node joins a sequence of message into a single message.

Example: Join a sequence of message into a single message and split it into multiple outputs.

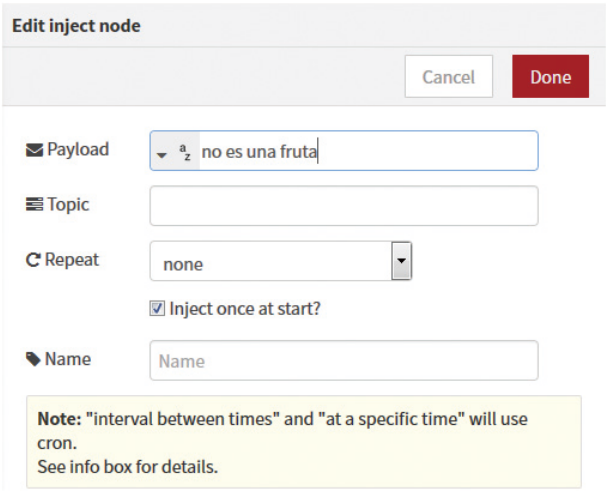
1. Add and connect 1 **inject** node, 1 **join** node, 1 **split** node, and 2 **debug** nodes to the workspace as shown.



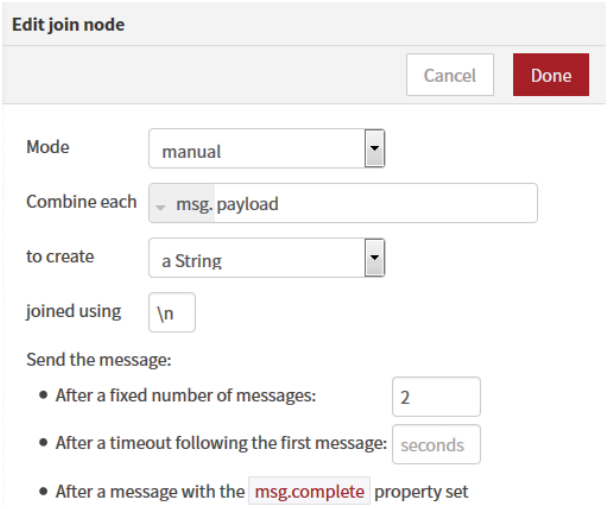
2. Edit the first inject node as shown and click **Done**.



3. Edit the second inject node as shown and click **Done**.



4. Edit the join node and set **After a fixed number of messages:** to 2 as shown and click **Done**.



5. Deploy the flow and click the button to the left of the first inject node and the second node in turn, and the result will be shown in the debug tab.

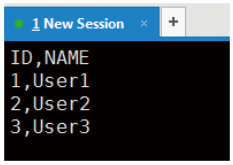
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.3.13 csv

The csv node parses the msg.payload to convert csv to/from a JavaScript object and places the result in the payload. The source may be a string, a file, a buffer, or a readable stream. The columns template should contain an ordered list of column headers. For csv input, these become the property names. For csv output, these specify the properties to extract from the object and the order for the csv.

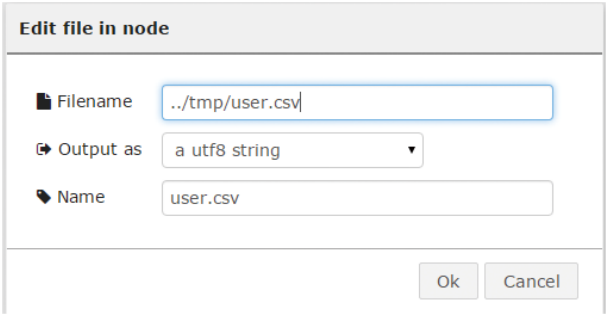
Example: Output the content of "user.csv" file as string messages

1. Create a "user.csv"(under folder ../tmp/) file on the device as shown.



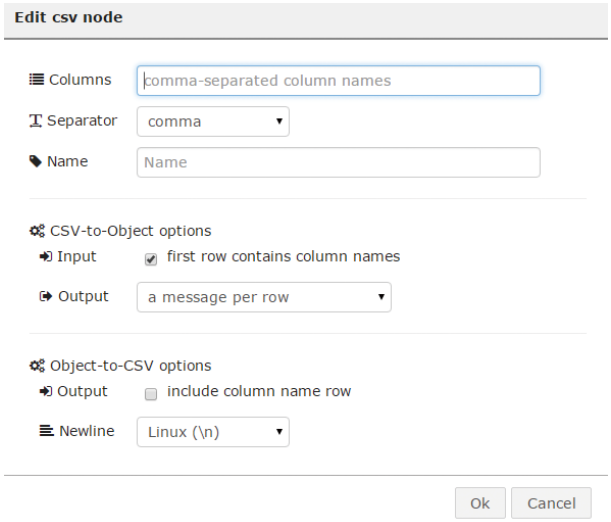
2. Connect an **inject** node, a **file in** node, a **csv** node, and a **debug** node to the workspace as shown.

3. Edit the **file in** node to assign the file called "user.csv" under (../tmp/), and give the node name "user.csv" then click **Ok**.



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

4. Edit the **csv** node as shown and click **Ok**.

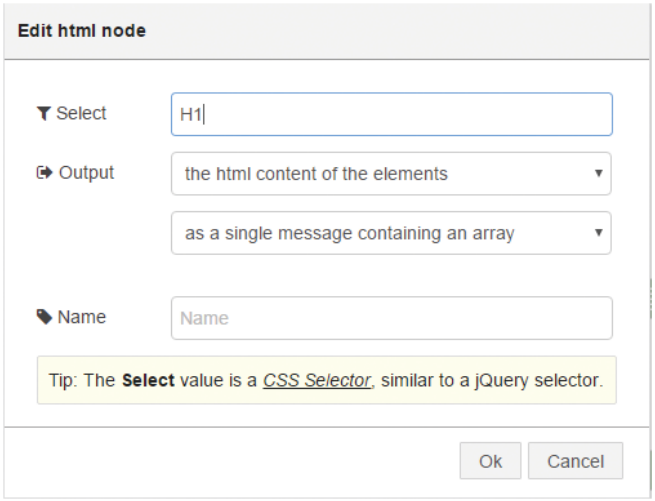


5. Deploy the flow and click the button to the left of the inject node, and the result will be shown in the debug tab.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.3.14 html

The html node extracts elements from an html document held in msg.payload using a selector that uses Cheerio with the CSS selector syntax. The result can be either a single message with a payload containing an array of the matched elements, or multiple messages that each contains a matched element.



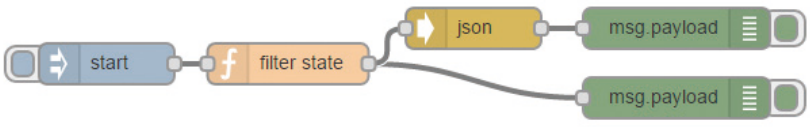
Select The name of header of element

7.3.15 json

The json node parses the msg.payload to convert a json string to/from a JavaScript object and place the result back into the payload.

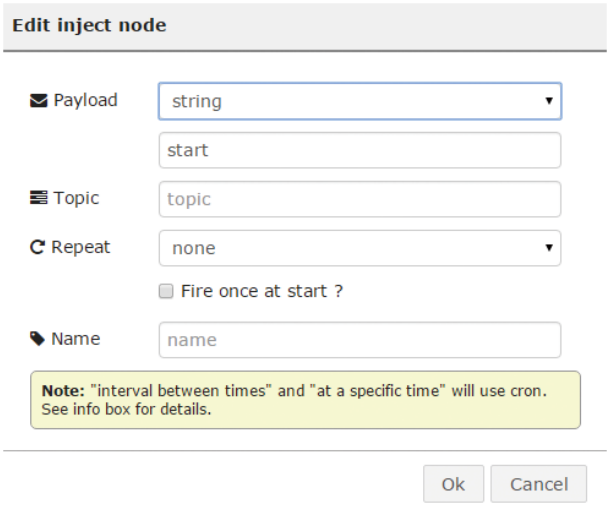
Example: Output a string "start" and the json string

1. Connect an **inject** node, a **function** node, a **json** node, and a **debug** node to the workspace as shown.



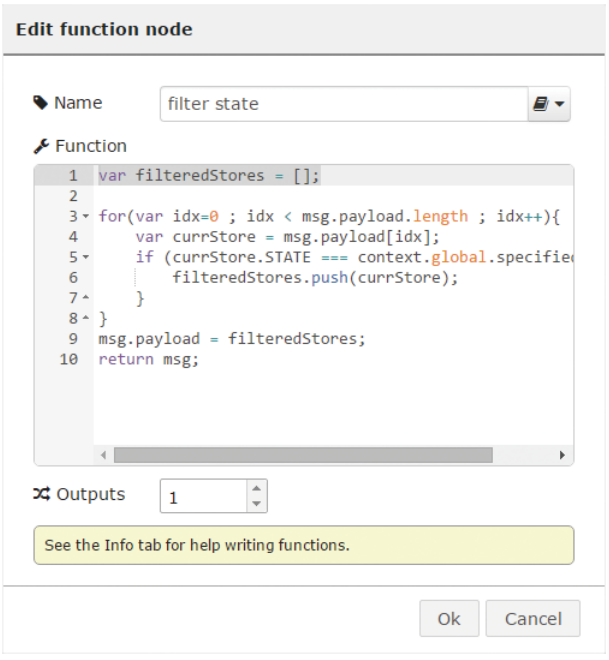
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

2. Edit the **inject** node to insert "start" string.



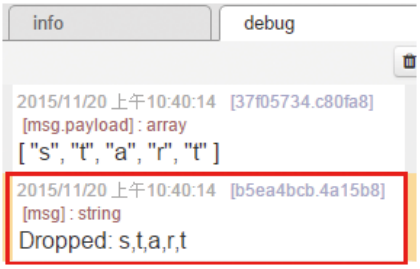
3. Edit the **function** node to add a JavaScript language as shown.

```
var filteredStores = [];
for(var idx=0 ; idx < msg.payload.length ; idx++){
    var currStore = msg.payload[idx];
    if (currStore.STATE === con-text.global.
specifiedState){
        filteredStores.push(currStore);
    }
}
msg.payload = filteredStores;
```



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

4. Deploy the flow and click the button to the left of the inject node and the result will be shown in the debug tab.



7.3.16 xml

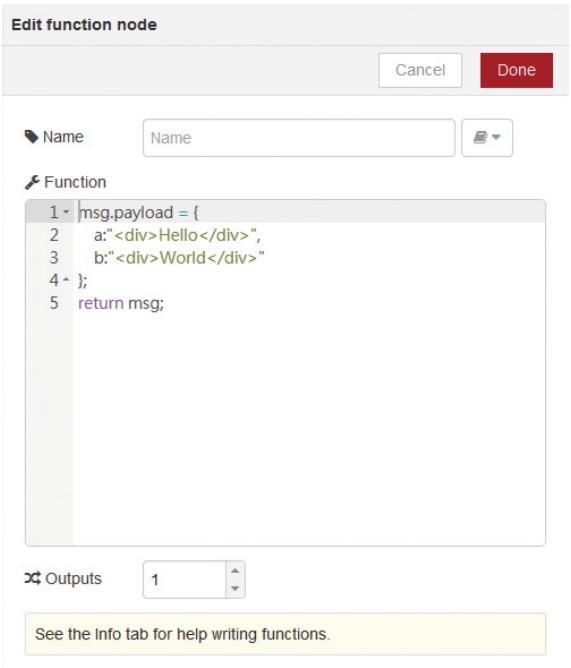
The xml node parses the msg.payload to convert xml to/from a JavaScript object, and place the result in the payload.

Example: Generate an xml file for an object

1. Connect an **inject** node, a **function** node, an **xml** node, and a **debug** node to the workspace as shown.

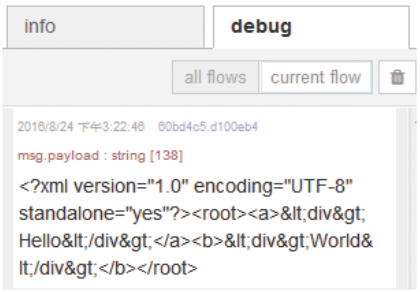
2. Edit the function node and add codes below as shown.

```
msg.payload = {
    a:"<div>Hello</div>",
    b:"<div>World</div>"
};
return msg;
```



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

3. Deploy the flow and click the button to the left of the inject node and the result will be shown in the debug tab.

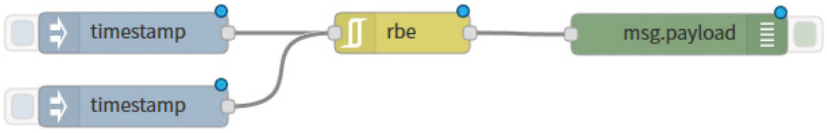


7.3.17 rbe

The rbe node passes or blocks until the msg.payload has changed. This works on number, strings and simple objects. On the other hand, it can block until the value changes by a specified amount – deadband mode. In deadband mode the incoming payload must contain a parsable number and is output only if greater than + or – the band gap away from previous value. Deadband also supports % - only sends if the input differs by more than a certain percentage of the original value.

Example: Check if the input value has changed by more than 5

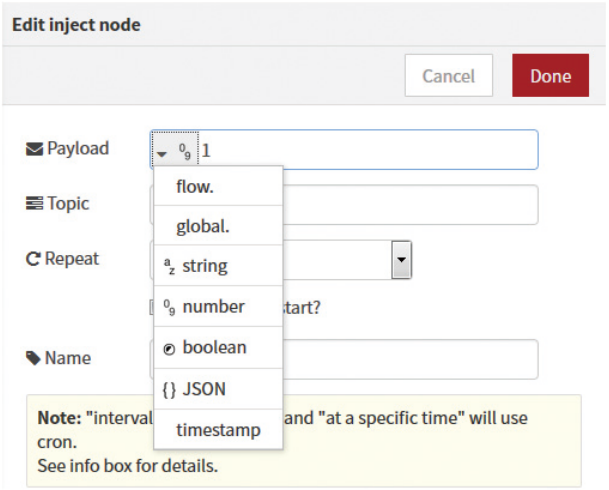
1. Add and connect 2 **inject** nodes, 1 **rbe** node, and 1 **debug** node to the workspace as shown.



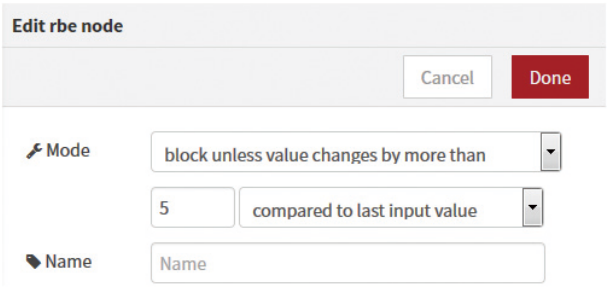
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

2. Click on the first inject node. Set the data type of **Payload** to **number** and fill **1** in the **Payload** field. Click **Done** when finished.

   Click on the second inject node. Set the data type of **Payload** to **number** and fill **9** in the **Payload** field. Click **Done** when finished.



3. Click on the rbe node and configure it to block unless value changes by more than 5 compared to last input value.



4. Deploy the flow and click the button to the left of the first inject node and the result will be shown in the debug tab.

   You can try clicking the button of the first inject node again. There will be no addition result showing in the debug tab.

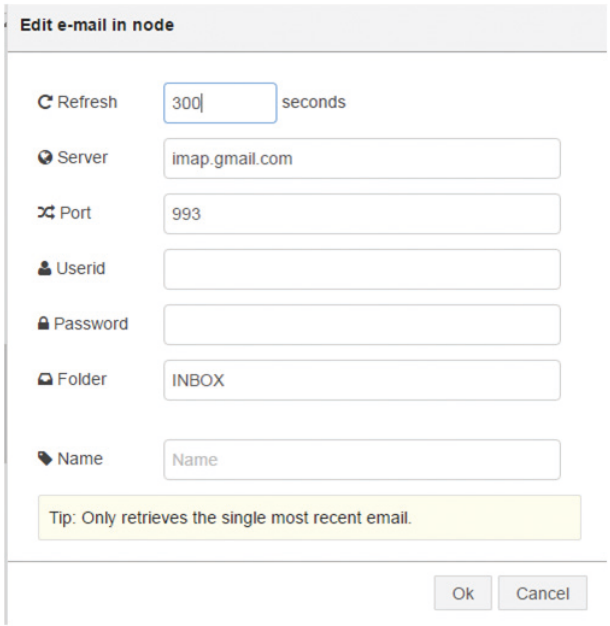
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

5. Now try clicking the button of the second inject node. The result will be shown in the debug tab.

7.4 Social Nodes

7.4.1 email in

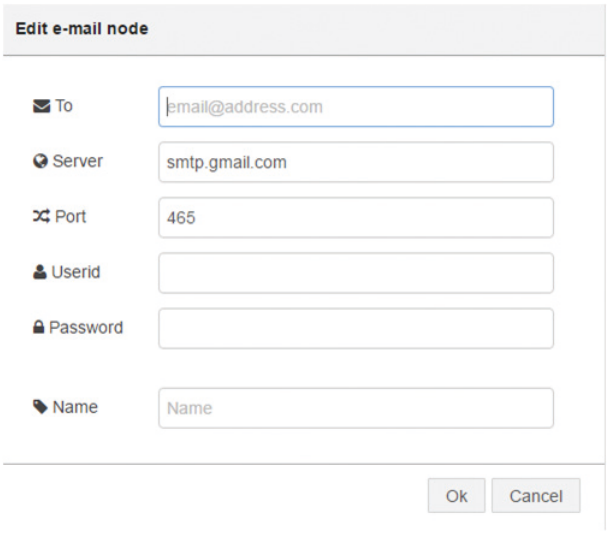
The email in node retrieves emails from email server. For security concern, the default setting is enabled the SSL over IMAP on 993 port.



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

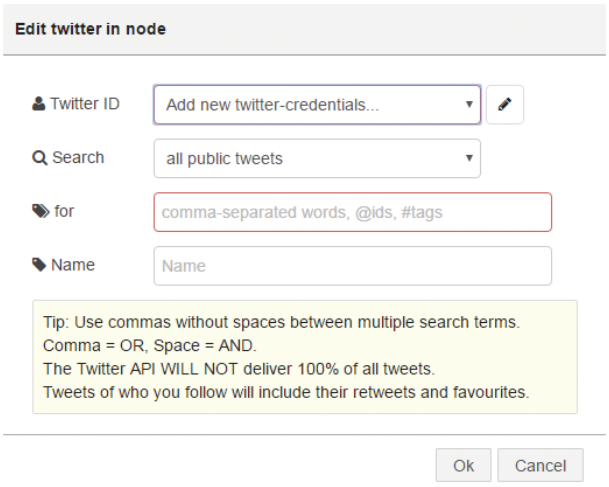
7.4.2 email out

The email out node sends emails through email server. For security concern, the default setting is enabled the SSL over SMTP on 465 port.



7.4.3 twitter in

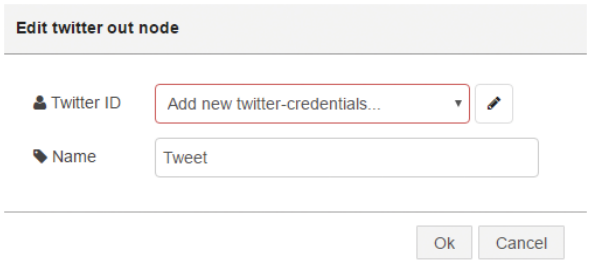
The twitter in node searches either the public or a user's stream for tweets containing the configured search term, all tweets by specific users, or direct messages received by the authenticated user.



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.4.4 twitter out

The twitter out node tweets the msg.payload to send a direct message using a payload like "D{username}{message}". If msg.media exists and is a buffer object, this node will treat it as an image and attach it to the tweet. If msg.params exists and is an object of name:value pairs, this node will treat it as parameters for the update request.



7.5 Storage Nodes

7.5.1 tail

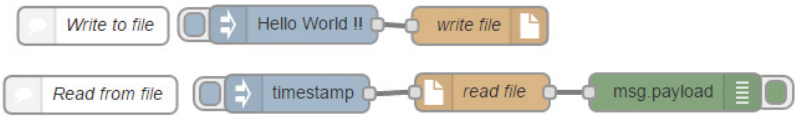
The tail node displays the last information appended into the file.

7.5.2 file in

The file in node reads the content of specific file in a UTF8 string/a buffer.

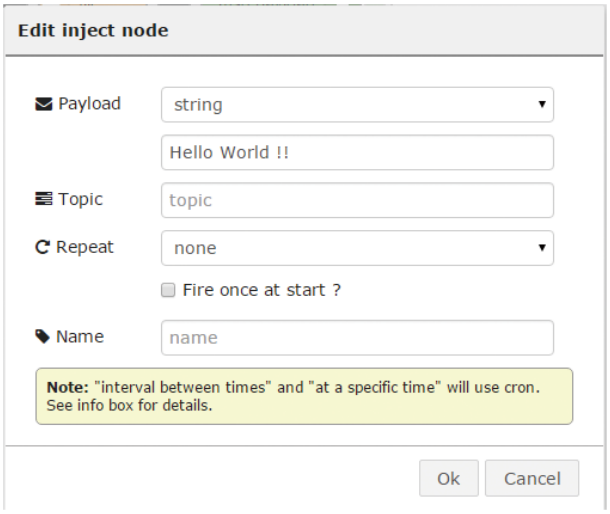
Example: Save a message to a file name "hello" on the gateway device and read the message from the file just saved

1. Connects 2 **inject** nodes, a **file in** node, a **file out** node and a **debug** node to the workspace as shown.

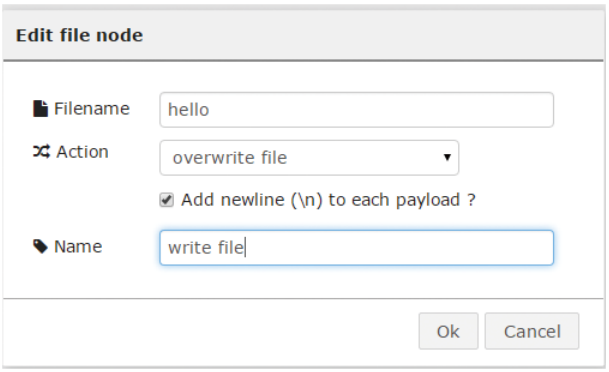


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

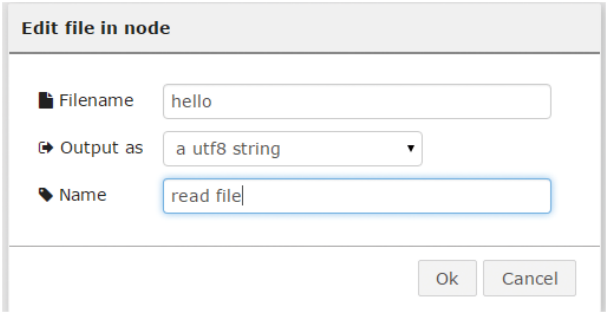
2. Edit the **inject** node. Set the payload to string "Hello World!!" which is the message saved to the file, and click **Ok**.



3. Edit the **file out** node. In **Action**, select **overwrite file** and tick **Add newline (\n) to each payload?**

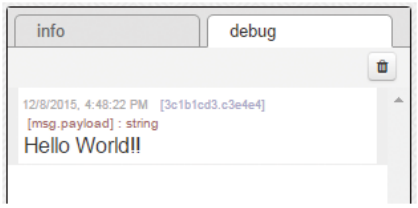


4. Edit the **file in** node to read the content from targeted file.



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

5. Deploy the flow and click the button to the left of the inject node and the result will be shown in the debug tab.



7.5.3 file out

The file out node writes msg.payload to the file specified, for example to create a log. You can configure the filename in the node. The filename shall be set in an incoming message on msg.filename if blank. A newline is added to every message, but this can be turned off if required, for example, to allow binary files to be written. The default behavior of the node is to append to the file, and this can be changed to overwrite the file each time, for example if you want to output a "static" web page or report, or to delete a file if required.

7.5.4 SQLite

The sqlite node allows basic access to a sqlite database. It uses the db.all operation against the configured database. Furthermore, it allows INSERT, UPDATE and DELETE in SQL language.

Example: Use sqlite command to create a database, create a table “user” in “testdb.db” on the device, and retrieve the data from “user” table.

1. Create a database, “testdb.db”, and create a table “*user*” in the database on the device with 3 records in it.

```
sqlite> select * from user;
ID          NAME
----------  ----------
1           User1
2           User2
3           User3
```

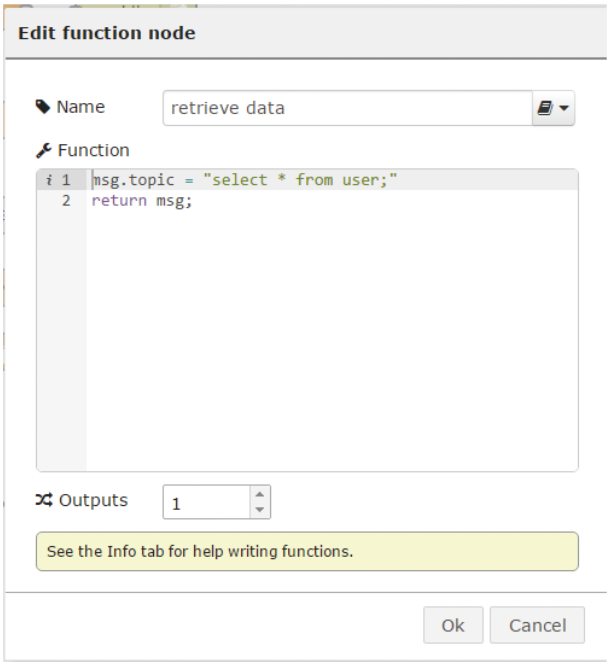
SQL command for reference:

```
#user:sqlite3 /home/root/testdb.db
sqlite>create table user(ID char(10),NAME char(20));
sqlite>insert into user values (‘1’,’User1’);
```

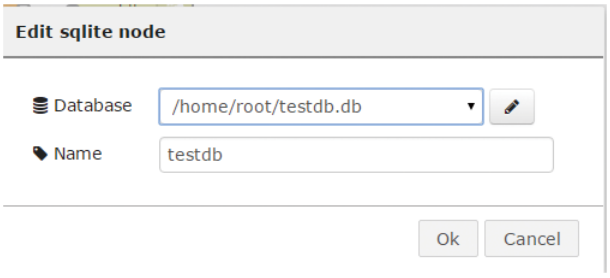

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

2. Connect an **inject** node, a **function** node, a **sqlite** node, and a **debug** node to the workspace as shown.

3. Edit the **function** node.
   Add SQL command into msg.topic. (msg.topic="select * from user;") in **Function** field.

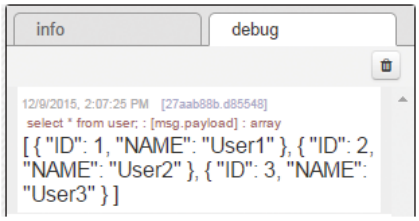


4. Edit the **sqlite** node.
   In **Database** field, point to the database "testdb.db", and give this node a name "testdb", then click **Ok**.



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

5. Deploy the flow and click the button to the left of the inject node and the result will be shown in the debug tab.



7.6 Analysis Nodes

7.6.1 sentiment

The sentiment node analyses the msg.payload and adds a msg.sentiment object that contains the resulting AFINN-111 sentiment score as msg.sentiment.score. The score is positive if greater than zero, and is negative, if less than zero. The range of the score is typically between 5 to +5, but can go higher and lower.

7.7 Advanced Nodes

7.7.1 watch

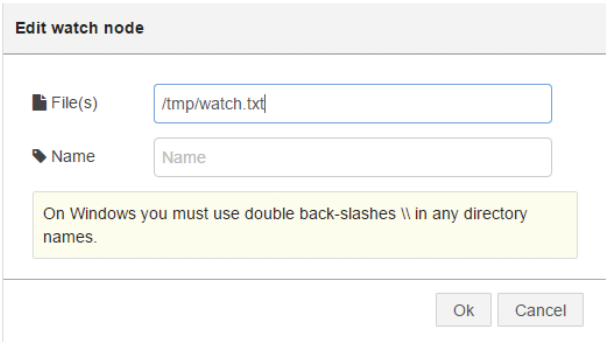
The watch node monitors the changes of a directory or a file. Multiple targets are allowed by entering a list of comma separated directories and/or files. Putting quotes "…" around any that have spaces in is required.

Example: Continue to monitor the changes of a file called / tmp/watch.txt

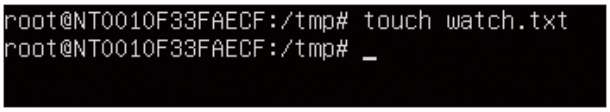
1. Connect a **watch** node and a **debug** node to the workspace as shown.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

2. Input the target file name with full path for monitoring.



3. Deploy the flow and turn to the CLI mode. Execute the command touch on the target file, then the result will be displayed in the debug tab.



7.7.2 feedparse

The feedparse node monitors an RSS/atom feed for new entries. The msg.topic contains the original article link, the msg.payload contains the description, and the msg.article contains the complete article object, which has properties such as .title, .summary, .date and so on.

7.7.3 exec

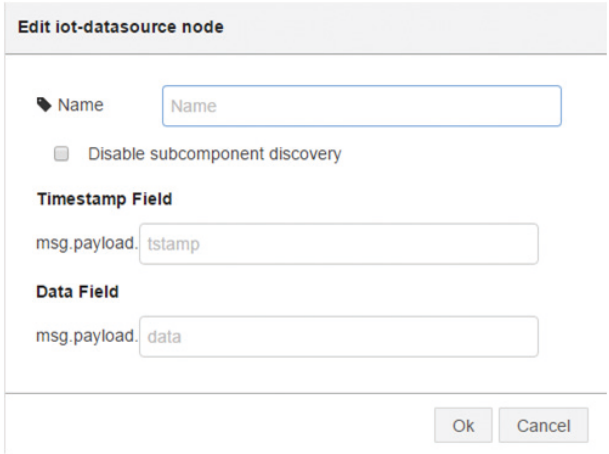
The exec node calls out to a system command and gets a callback on completion, returning the complete result in one message, along with any errors. The optional append gets added to the command after msg.payload, so you can do things like pipe the result to another command. Parameters with spaces should be enclosed in quotes.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.8 Gateway Kit Nodes

7.8.1 iot-datasource

The iot datasource node provides the data as Dashboard data inputs.



Disable subcomponent discovery

If checked, the **iot-datasource** node does not attempt to look inside the data field and split it into subfields. For example, discover enabled and data format shown as below.

```
msg.payload = {
  tstamp: 1438637044000,
  data: {
    x: 3.14,
    y: 1.41,
    z: 6.02
  }
}
```

iot-datasource node goes inside msg. payload.data and find the fields x, y, and z, and present them to the Dashboard as separate data points. Once disabled, the Dashboard will receive the entire JSON object msg.payload.data as one data point.

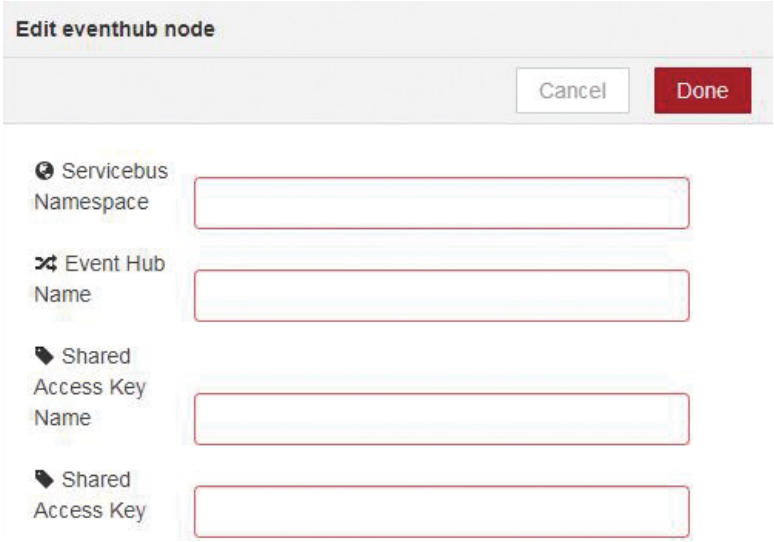
A line chart might need them split up so it can chart the data points separately, but a 3D scattered plot would need the data intact, since the entire object would represent just one data point on the plot.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.9 Cloud Nodes

7.9.1 eventhub

The eventhub node sends events to Azure™ Event Hub for live monitoring, analytics, statistics, and even business intelligence. Before enabling this node, you need to apply the services in Azure™ portal and get the related information.

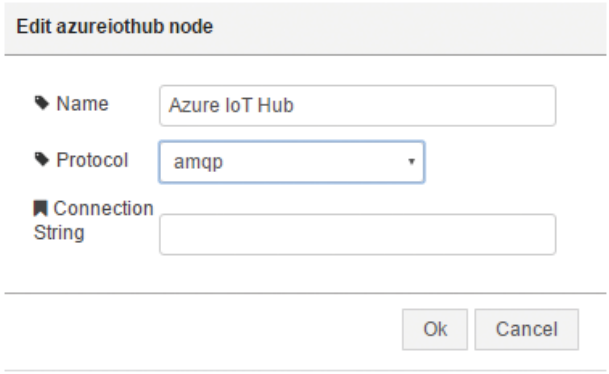


Servicebus Namespace	The title of the name space you termed in Azure™.
Event Hub Name	The name of the hub you created in Azure™.
Shared Access Key Name	The title of the policy in Shared access policies of your Azure™ account.
Shared Access Key	The connection string-primary key of the policy above in your Azure™ account.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.9.2 azureiothub

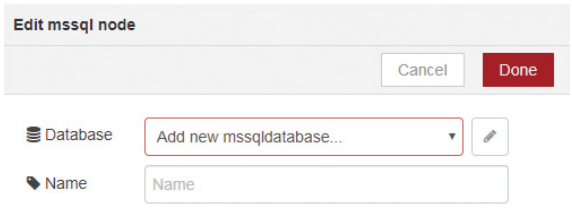
The azureiothub node sends field data to the Azure™ IoT Hub for live monitoring, field data extraction, device management, and so on. It supports multiple communication protocols including HTTP, MQTT, AMQP and AMQPWS.



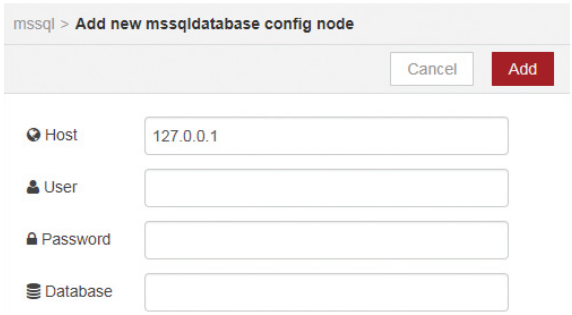
Name		The name of the node.
Protocol	**http/amqp/ mqtt/amqpWs**	The dedicated protocol of your selected device.
Connection String		The connection string of your selected device.

7.9.3 mssql

The mssql node executes SQL command in remote database like Azure™ Database Service via network. You can create a database (MSSQL) either at the remote or at the local for data processing.



Database	The list of mssql servers. Click ✎ to add a new mssql server.
Name	The name of the node.



Host	The IP address/hostname of the database.
User	The name of the authorized identity to the database.
Password	The credential of the authorized identity.
Database	The name of the database.

The following examples assume that you have built up a database "TESTDB" in Azure™, created a table "usertable" with fields "userid" and "username", and filled with data as the table below.

userid	username
00001	ericchiu
00002	superman
00003	spiderman
00004	hitman

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Example: Select a table from a MSSQL database in Azure™ and delete a piece of record in a table from a MSSQL database in Azure™.

1. Add and connect 1 **inject** node, 1 **function** node, 1 **mssql** node, and 1 **debug** node to the workspace as shown.

2. Click on the **mssql** node. Fill in the required information to connect your database in Azure™.

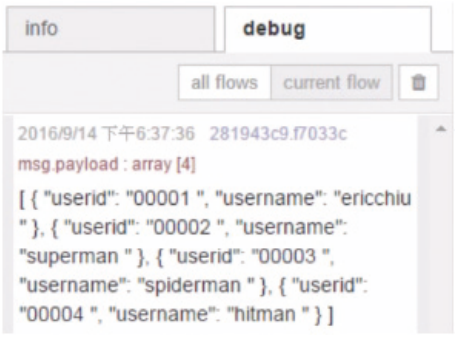
3. Click on the function node, and put the code below into **Function** field as shown.

```
msg.qurey  = "SELECT * FROM usertable";
```

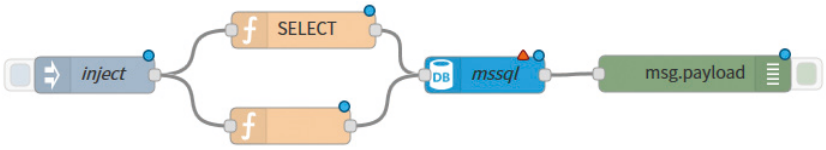
Click **Done** when finished.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

4. Deploy the flow and click the button to the left of the inject node, and the result will be shown in the debug tab.



5. Add another **function** node and connect it to the other nodes as shown.



6. Click on the newly added function node, and put the code below into **Function** field as shown.

```
msg.qurey = "DELETE FROM usertable WHERE
userid = '00003';
```

Click **Done** when finished.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7. Deploy the flow and click the button to the left of the inject node, and the result will be shown in the debug tab.

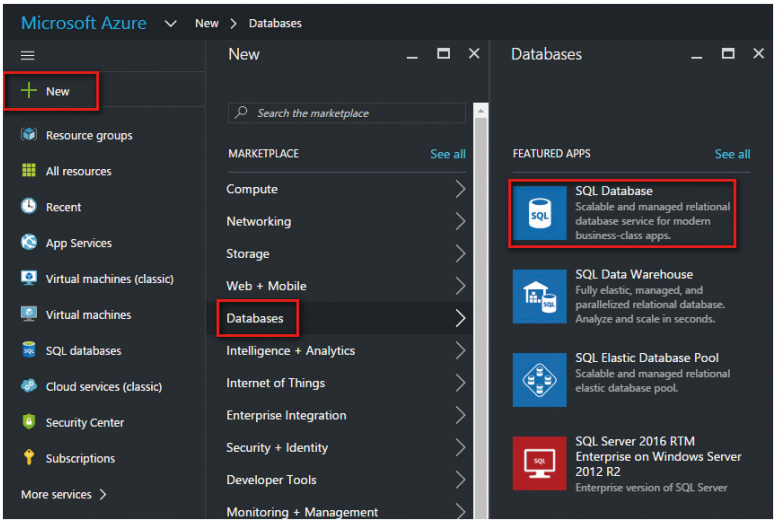
8. You can use your SQL management tool to check the records of the table “usertable”. Both of “00003” and “spiderman” are gone.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

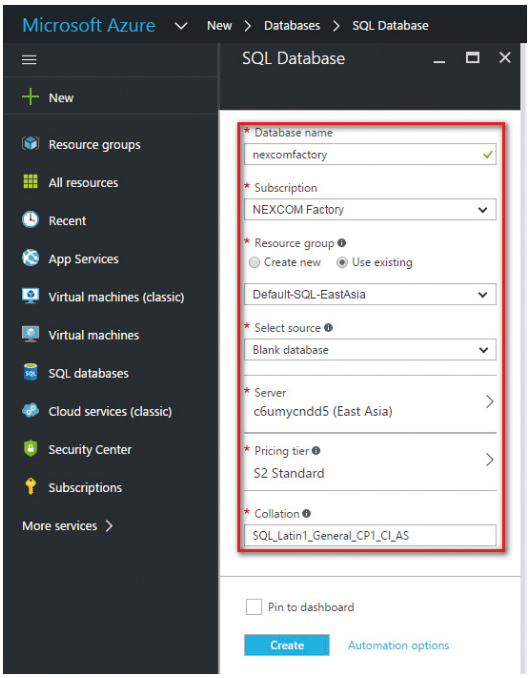
Example: Establish a connection to a MSSQL database in Azure

1. Add and connect 1 **inject** node, 1 **mssql** node, and 1 **debug** node to the workspace as shown.

2. Logon to Microsoft Azure and create a **SQL Database** as shown.

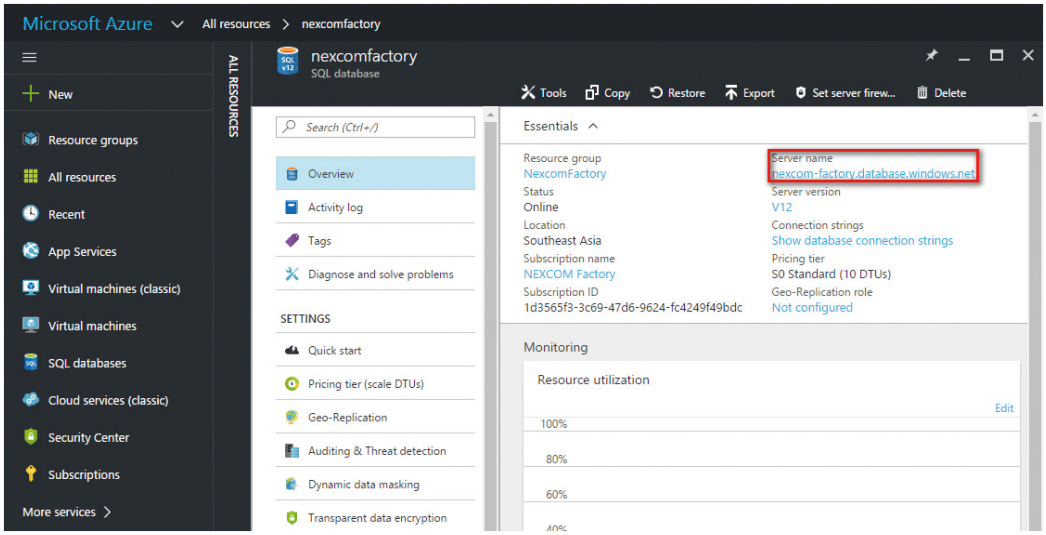


3. Fill out the required information for your database and click **Create** as shown.

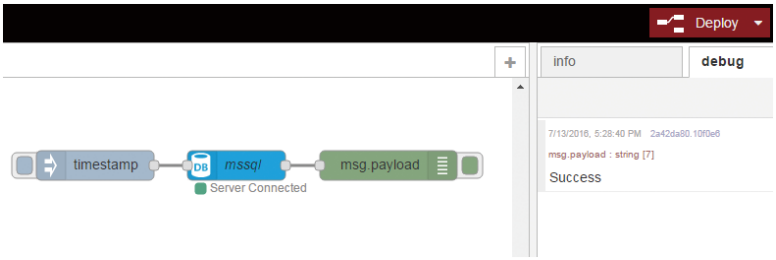


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

4. Along with the configuration, copy and paste **ServerName** to the mssql node.



5. Deploy the flow and click the button to the left of the inject node, and the result will be shown in the debug tab.



The following examples assume that you have built up a database "TESTDB" in Azure, created a table "usertable" with fields "userid" and "username", and filled with data as the table below.

userid	username
00001	ericchiu
00002	superman
00003	spiderman
00004	hitman

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Example: Select a table from a MSSQL database in Azure and delete a piece of record in a table from a MSSQL database in Azure

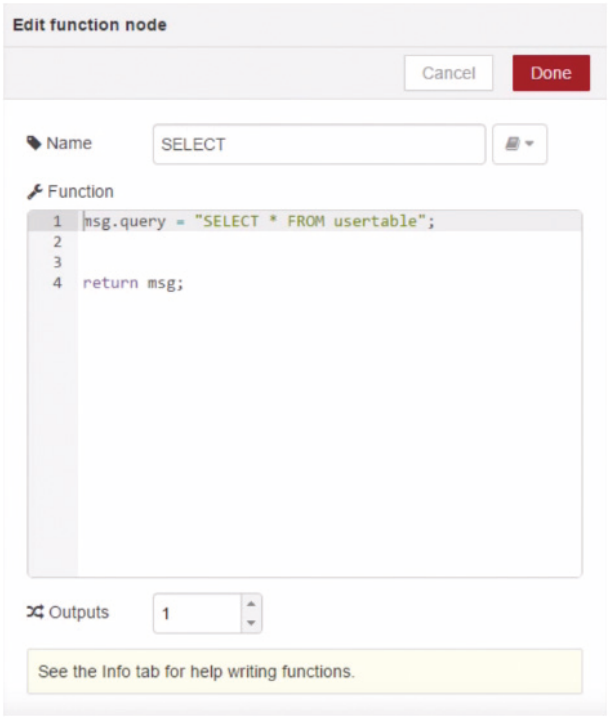
1. Add and connect 1 **inject** node, 1 **function** node, 1 **mssql** node, and 1 **debug** node to the workspace as shown.

2. Click on the mssql node. Fill in the required information to connect your database in Azure as described in the previous example.

3. Click on the function node, and put code below into **Function** field as show.

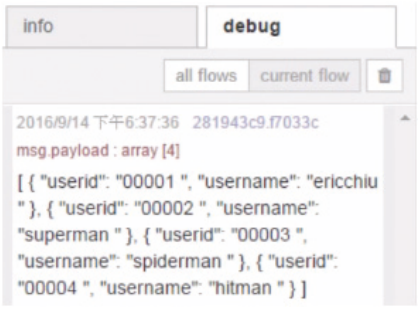
```
msg.qurey  = "SELECT * FROM usertable";
```

Click **Done** when finished.

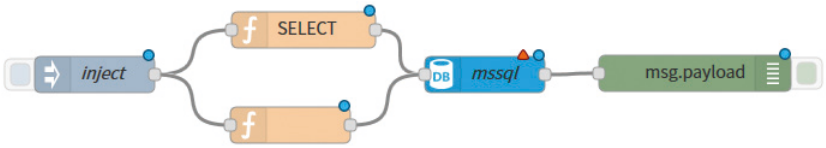


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

4. Deploy the flow and click the button to the left of the inject node, and the result will be shown in the debug tab.



5. Add another **function** node and connect it to the other nodes as shown.



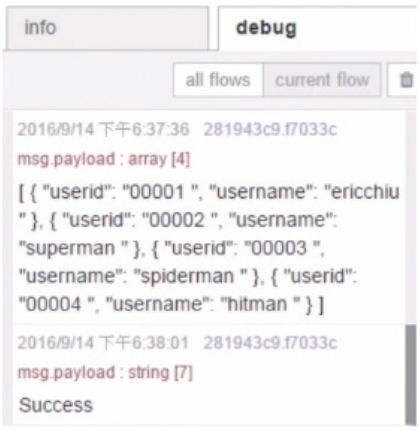
6. Click on the newly added function node, and put the code below into **Function** field as show.

```
msg.qurey  = "DELETE FROM usertable WHERE userid =
'00003';
```

Click **Done** when finished.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7. Deploy the flow and click the button to the left of the inject node, and the result will be shown in the debug tab.



8. You can use your SQL management tool to check the records of the table "usertable". Both of "00003" and "spiderman" are gone.

7.10 Data Process Nodes

7.10.1 boundary

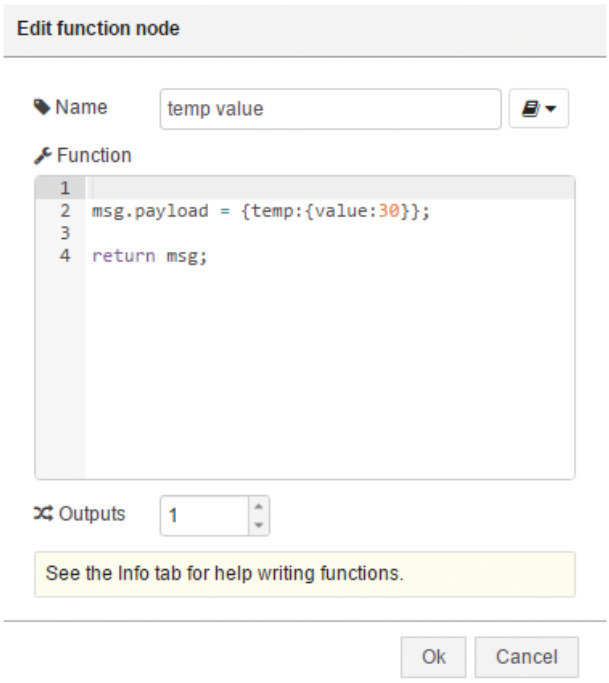
The boundary node triggers the alarm since the value of object is out of the range of setting.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Example: Monitors a value in a preset range and triggers alarms if not in the range

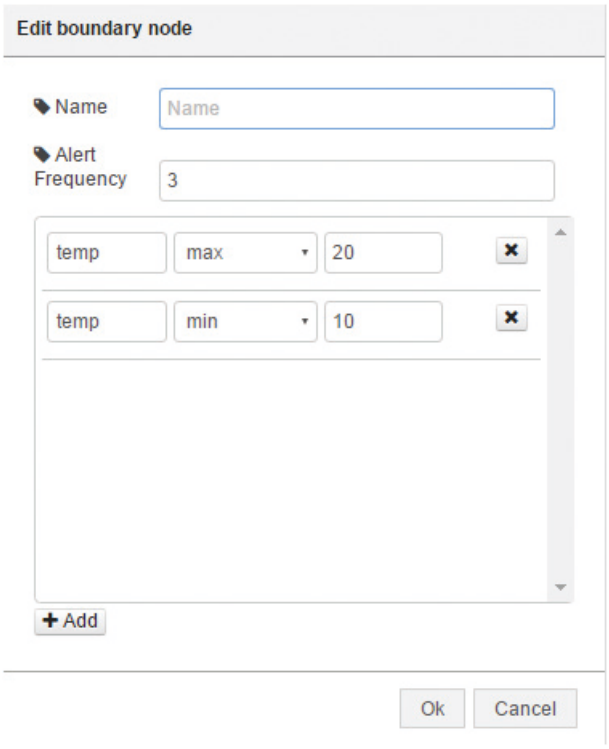
1. Connect an **inject** node, a **function** node, a **boundary** node, and a **debug** node to the workspace as shown.

2. Edit the **function** node.
   Create an object msg.payload = {temp : {value:30}}.



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

3. Edit the boundary node.
   Set the values as shown, so if the value is greater than 20 or lesser than 10, then the node triggers the alarm.



4. Deploy the flow and click the button to the left of the inject node, and the result will be shown in the debug tab.

You can notice the boundary node in the workspace prompts warnings.

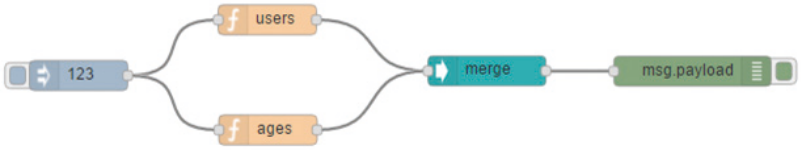
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.10.2 merge

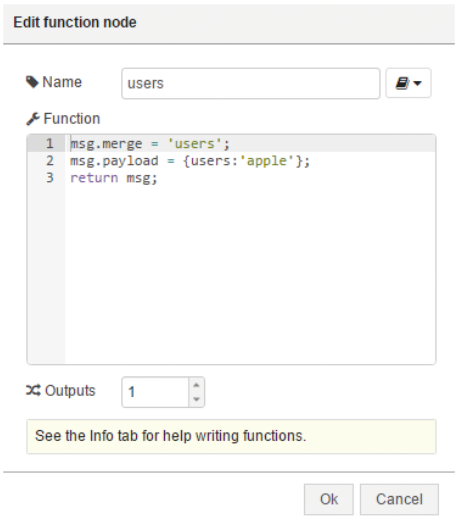
The merge node merges two objects into one object for specific data processing.

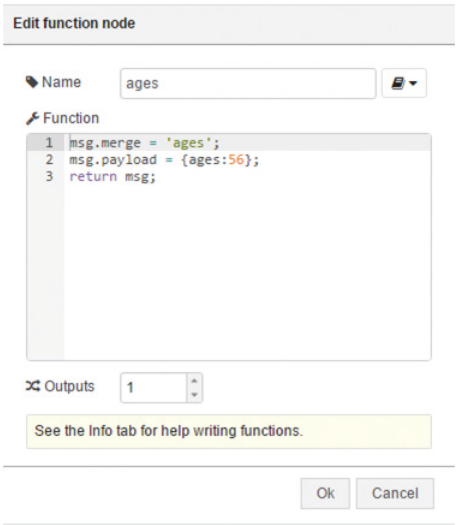
Example: Merge two columns, users and ages, into one object for data processing

1. Connect an **inject** node, a **function** node, a **merge** node, and a **debug** node to the workspace as shown.



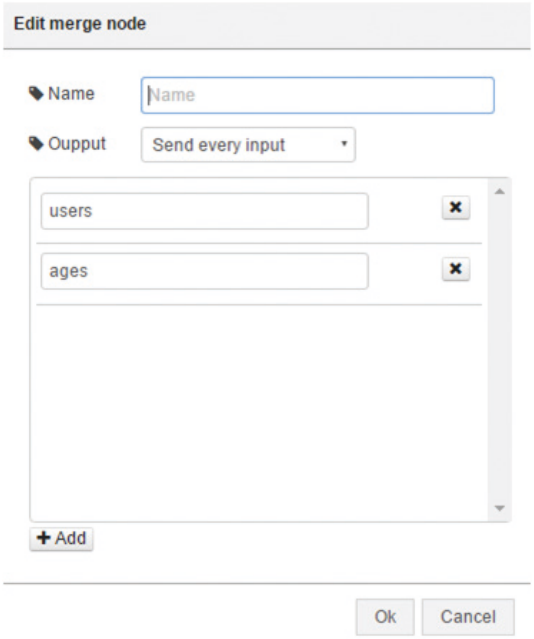
2. Edit the **function** nodes and create two objects named "users" and "ages" individually as shown.



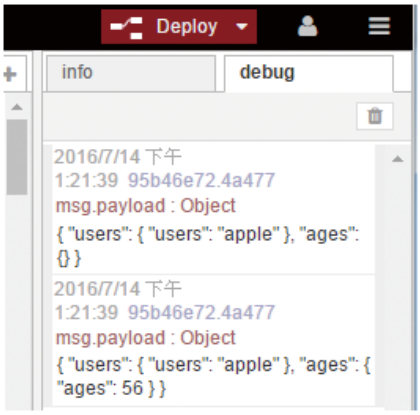


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

3. Edit the **merge** node.
   Input the subject of each object. With “user” and “ages”.

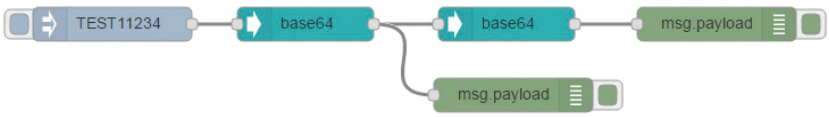


4. Deploy the flow and click the button to the left of the inject node, and the result will be shown in the debug tab.



7.10.3 cypher

The cypher node encrypts or decrypts the data stream to and from input source based on base64 or 3DES algorithm.



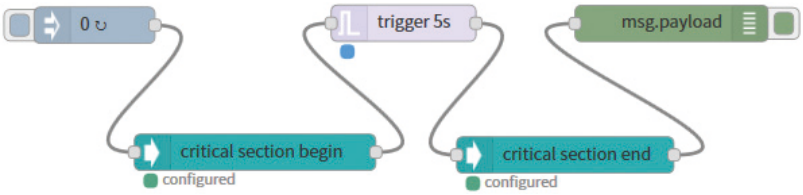
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.10.4 critical section

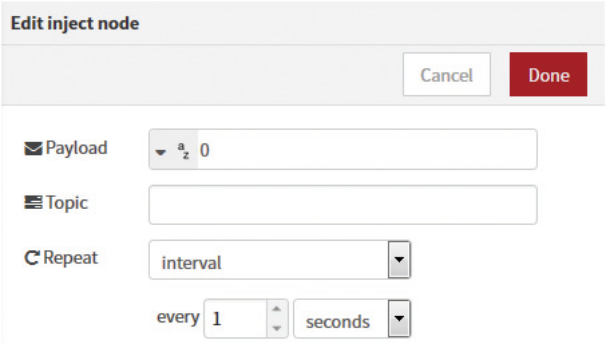
The critical section nodes guarantees the first-in & first-out in a task pipeline. Concurrently apply both nodes of critical section begin and critical section end to your flow to ensure the job gets done in sync without hassles. You can have more than one pair of critical section in your flow. Make sure the corresponding pair shares the same mutex.

Example: Set up a critical section to force the one second interval request to wait for 5 seconds for output.

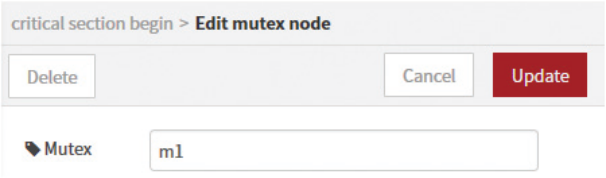
1. Connect 1 **inject** node, 1 **critical section begin** node, 1 **trigger** node, 1 **critical section end** node, and 1 **debug** node to the workspace as shown.



2. Edit the inject node as shown and click **Done**.

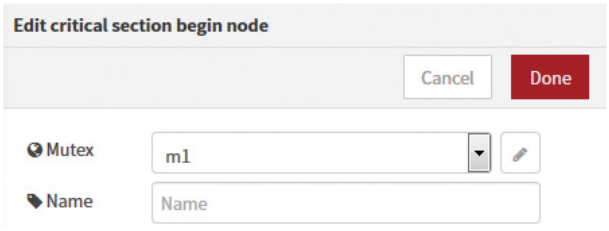


3. Click on the critical section begin node and click . Input "m1" in the field of **Mutex**. Click **Update** when done.

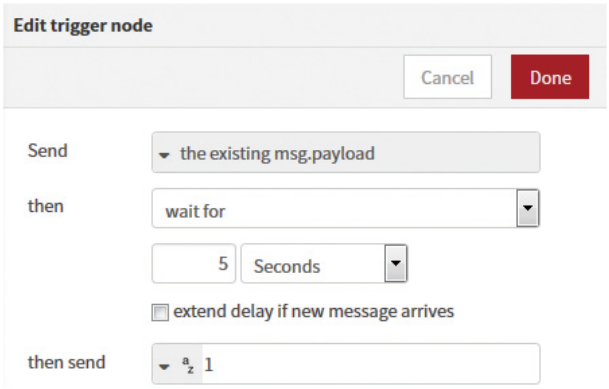


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

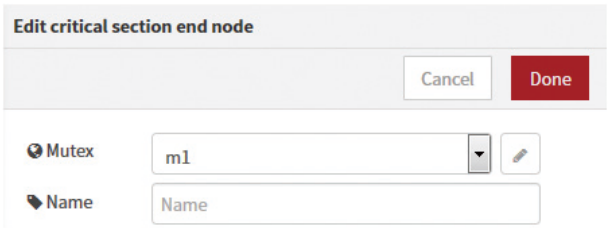
Click **Done** to exit editing node.



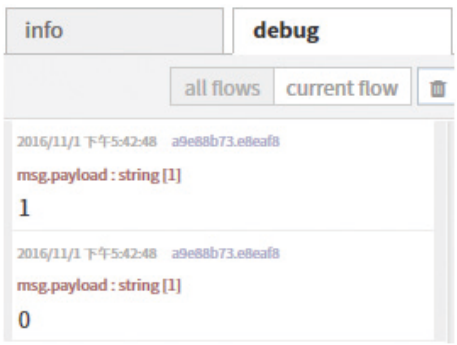
4. Edit the trigger node as shown and click **Done**.



5. Click on the critical section end node and select "m1" from the dropdown list of **Mutex**. Click **Done** when finished.



6. Deploy the flow and click the button to the left of the inject node, and the result will be shown in the debug tab.



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

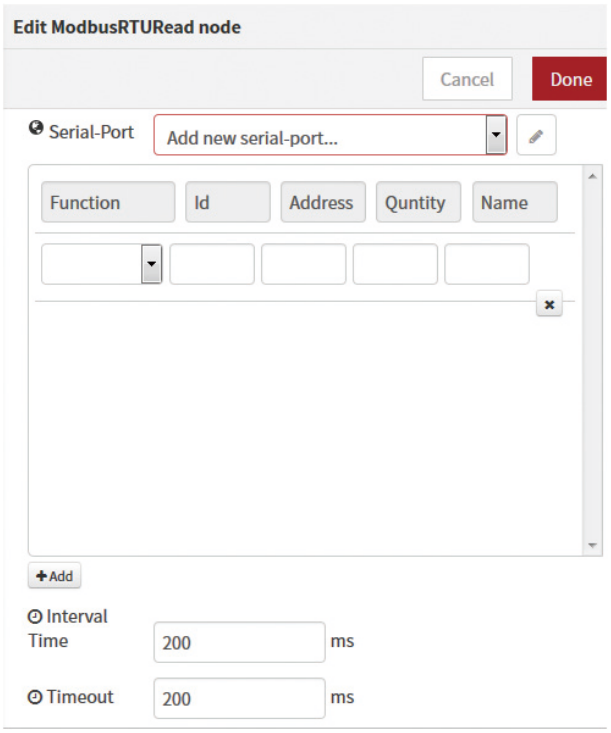
7.10.5 HWInfo

The HWInfo node reveals hardware information of the host installed with Xcare.

7.11 Modbus Nodes

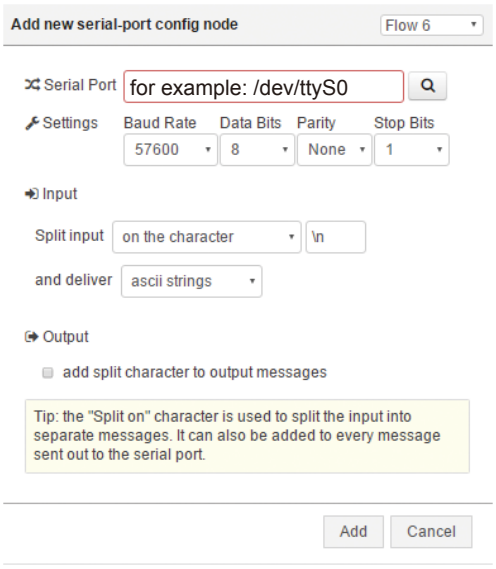
7.11.1 Modbus RTU

The Modbus RTU node sends the msg.payload to a serial interface and expects a response. The response will be output in msg.payload as a buffer, so user may need to use ".toString()" to convert.



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Serial Port	**Settings**	The local interface of serial input. Baud Rate, Data Bits, Parity, Stop Bits: 57600, 8, None, 1
FC (Function Code)	**Read Coils**	Read single bit. This command reads the ON/OFF status of coils (0x reference address) in the slave/server.
	Read Discrete Inputs	Read single bit. This command reads the ON/OFF status of discrete inputs in the slave/server.
	Read Holding Registers	Read 16-bit word. This command reads the binary contents of holding registers (4x reference address) in the slave device.
	Read Input Registers	Read 16-bit word. This command reads the binary contents of input registers (3x reference addresses) in the slave device.
	Write Coil(s)	Write single bit or a 16-bit word. Simultaneously forces a series of coils (0x reference address) either ON/OFF.



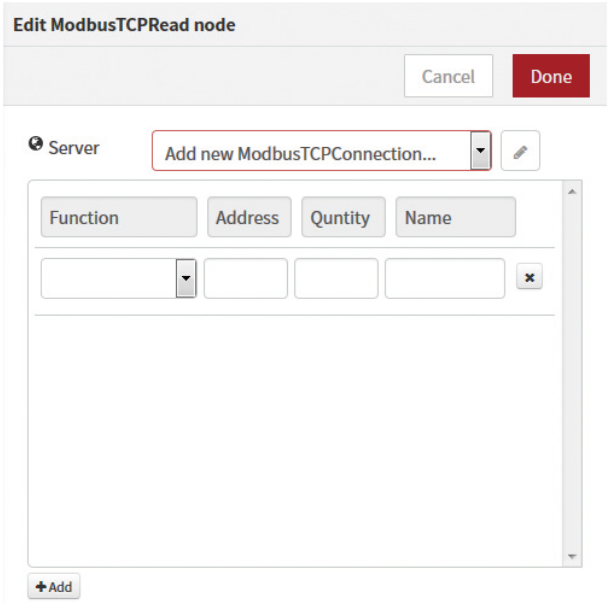
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

FC (Function Code)	**Write Register(s)**	Write single bit or a 16-bit word. This command presets a single holding register (4x reference address) to a specific value. The Preset Multiple Registers normal response message returns the slave address, function code, starting register reference, and the number of registers preset, after the register contents have been preset.
Device ID		The value from 1 to 247
Address		The value from 1 to 65536
Quantity		The value from 1 to 65536
Name		The tag of the connection callable by other nodes **Note:** leave this field blank will keep other nodes from transporting values.
Interval Time		The interval of time in milliseconds to poll the slave/server
Timeout		The interval of time in milliseconds to stop

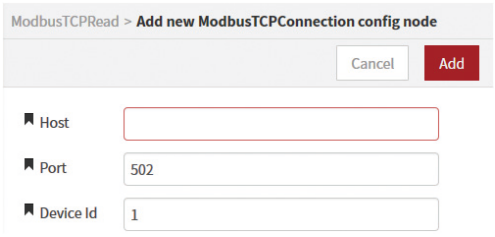
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.11.2 ModbusTCP

The Modbus TCP Read node sends the msg.payload to a Modbus TCP portand expects a response. The response will be output in msg.payload as a buffer, so you may need to use ".toString()" to convert.



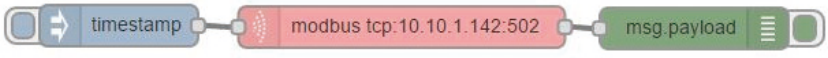
Server	**Host**	The IP address to access.
	Port	The port number of the IP address to access.
	Device ID	Device ID
FC (Function Code)	**Read Coil**	Read single bit. This command reads the ON/OFF status of coils (0x reference address) in the slave/ server.



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

FC (Function Code)	**Read Discrete Input**	Read single bit. This command reads the ON/OFF status of discrete inputs in the slave/server.
	Read Holding Registers	Read 16-bit word. This command reads the binary contents of holding registers (4x reference address) in the slave device.
	Read Input Registers	Read 16-bit word. This command reads the binary contents of input registers (3x reference addresses) in the slave device.
	Write Coil(s)	Write single bit or a 16-bit word. Simultaneously forces a series of coils (0x reference address) either ON/OFF.
	Write Register(s)	Write single bit or a 16-bit word. This command presets a single holding register (4x reference address) to a specific value. The Preset Multiple Registers normal response message returns the slave address, function code, starting register reference, and the number of registers preset, after the register contents have been preset.
Address		The value from 1 to 65536.
Quantity		The value from 1 to 65536.
Name		The tag of the connection callable by other nodes. **Note:** leave this field blank will keep other nodes from transporting values.

Example: Build a Modbus TCP server on PC with Modbus TCP request node.



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.12 OPCUA Nodes

7.12.1 OpcUA Item

The OpcUA Item node defines OPC UA item, type, and value to represent the connection to data sources within the server.

Item		Use a valid OPC UA address such as ns=2;i=4 OR ns=3;s=MyVariable
Type	**Integer**	Available data types for the operations and the way values can be stores
	Float	
	Double	
	Unsigned Int16	
	Boolean	
	String	
Value		Required if writing to the OPC UA server.
Name		The name of the node

7.12.2 OpcUA Client

Use this node to interact with an OpcUa Server. The value to write should be injected by an OpcUA Item. The value is written as json in msg.payload.

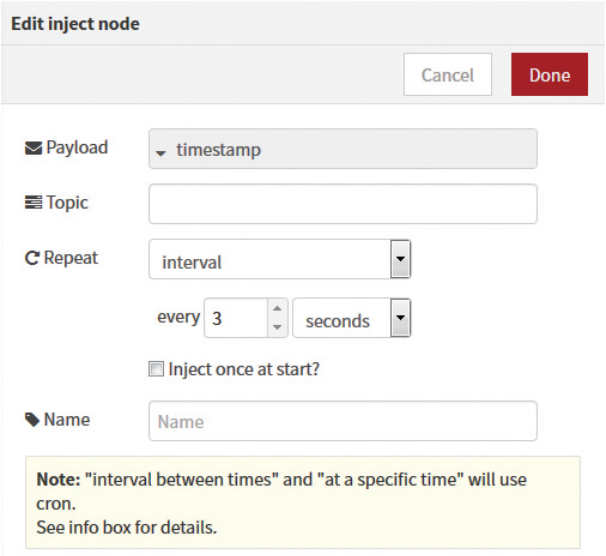
Endpoint		Connection to an endpoint such as opc.tcp://host:port/UA/EndpointName.
Action	**Read**	Get data from the server.
	Write	Put data to the server.
	Browse	Browse items on the server.
	Subscribe	Get the monitored item from the server.
	Event	Get events or alarms from the server.
Name		The name of the node.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

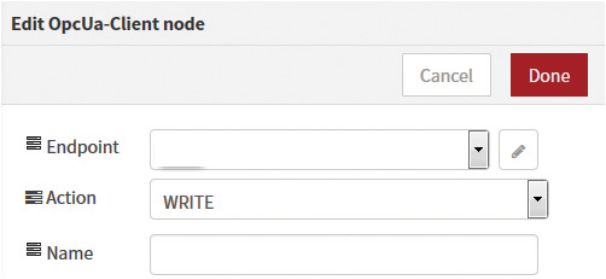
Example: Establish a connection and write to an OPC UA server

1. Add and connect 1 **inject** node, 1 **OPC UA Item** node, 1 **OPC UA Client** node, and 1 **debug** node to the workspace as shown.

2. Click the **inject** node, set **Repeat** to interval for **every 3 seconds**, and then click **Done**.

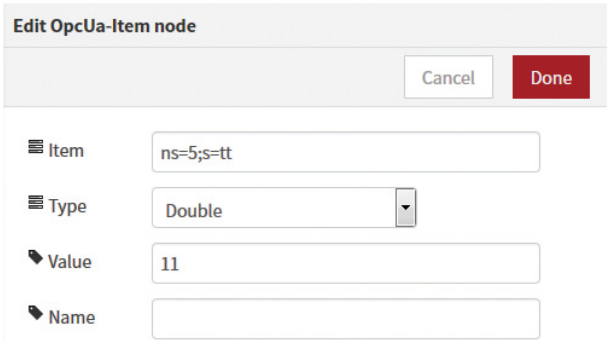


3. Click on the **OPC UA Client** node. Add or select an endpoint in **End-point** field and set **Action** to WRITE. Click **Done** when finished.

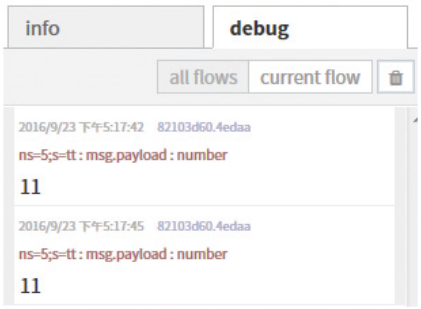


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

4. Click on the **OPC UA Item** node.
   Refer to your OPC UA server and fill the value of signal name/NodeId such as ns=2;i=4 in the **Item** field. Set **Type** to Double and put 11 to the **Value** field.
   Click **Done** when finished.



5. Deploy the flow and click the button to the left of the **inject** node, and the result will be shown in the debug tab.



6. You should notice that the value of signal name/NodeId on your OPC UA server fluctuates to "11" every 3 second as shown.

7.12.3 OpcUA Browser

Use with an inject node of a timestamp or fill topic of msg object to browse the OPC UA item.

Endpoint	Connection to an endpoint such as opc.tcp://host:port/UA/EndpointName.
Topic	Definition of one browsable OPC UA Item address such as ns=0;i=85

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

7.13 Siemens S7

7.13.1 read

The Siemens S7 read node reads a signal from a SPS via the SiemensS7 Communication protocol and injects it into the flow.

Connection	The IP address and parameters of the Siemens S7.
Signal	Use the connection selector and the connection configurator button underneath to configure signals.
Repeat	This allows the payload to be sent on the required schedule. If the repeat function is disabled, the node can be triggered from outside such as an inject node.

The contents are stored in msg.payload as a JSON object with the following format:

payload.Signal: Name of an SPS Address e.g. machine#1
payload.Path: Absolut addressing of an SPS Address e.g. DB10,BYTE1-4,
payload.Error: Error value: 0=noError, -1=Error,
payload.Value: Returned Data within an Array. The returned data can be:
null: if returned data is invalid;
single value: if the Quantity is 1;
array of values: if the Quantity is > 1;

7.13.2 write

The SiemensS7 write node writes a value to a signal of a SPS via the SiemensS7 Communication protocol.

Connection	The IP address and parameters of the Siemens S7.
Signal	Use the connection selector and the connection configurator button underneath to configure signals.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

The value is injected into the node and then sent to the selected signal. If an inject node is used, the writing value has to type in as a JSON within an Array of the payload such as {"Value":[1]}.

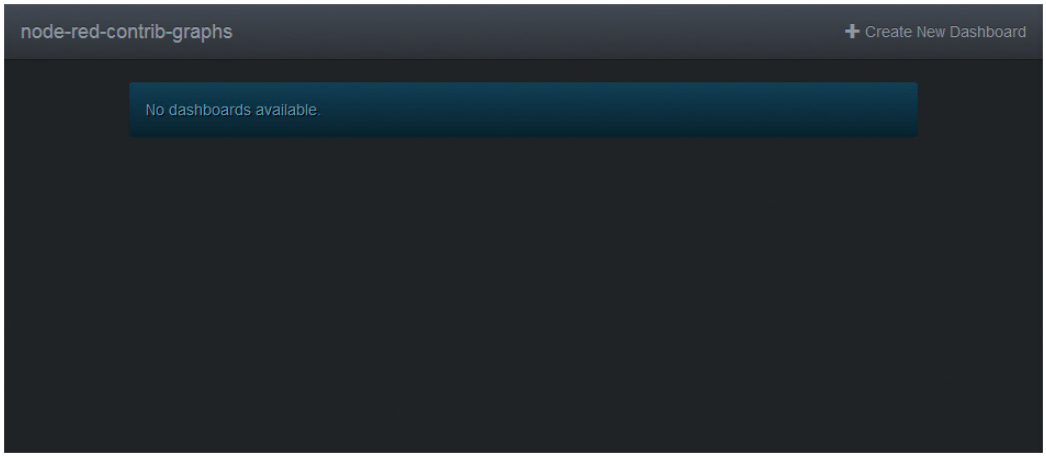
The SiemensS7 write node confirms each writing-job and the contents are stored in msg.payload as a JSON object with the following format:

payload.Signal: Name of an SPS Address e.g. machine#1
payload.Path: Absolut addressing of an SPS Address e.g. OB1,
payload.Error: Error value: 0=noError, -1=Error,
payload.Value: Writing Value (decimal) or Array of Values (decimal)

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Chapter 8: Dashboard

This chapter introduces the user interface and the basic operation of NEXCOM IoT Studio Dashboard. Once you log onto your CPS 200/100's IP address/dash, such as http://192.168.253.1/dash, with your browser, you will see the page as shown below.

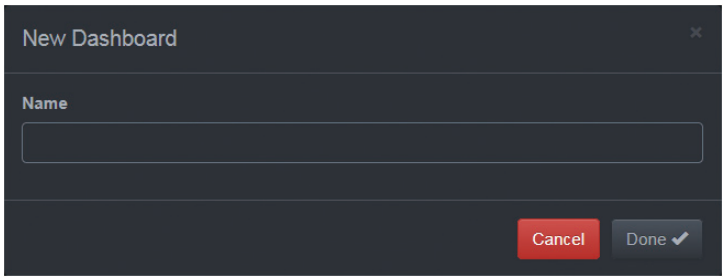


Prerequisite

You have created at least one IoT data source node in your flow, or you will not see any of data reflections on the graphs or charts.

8.1 Create Your Dashboard

1. Click + Create New Dashboard at the top right of the page.
2. Fill a desired name in the **Name** field, and click **Done**.

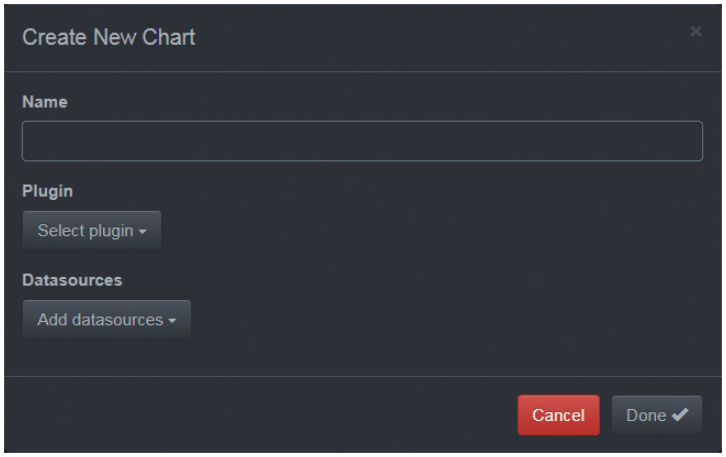


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

8.2 Create Your Chart

In your dashboard, click Create New Chart at the top right of the page. As the window pops up,

1. Fill a desired name in the **Name** field.
2. Select a plugin from the drop-down menu of **Plugin**.
3. Select a datasource from the drop-down menu of **Datasources**. You should see the name of the IoT datasource nodes you created in your flow.
4. Click **Done** and you should see the chart that reflects with your datasource.



Available Plugin	Description
Alert	Showing an alert sign based on the data source.
Circle Gauge	Measuring data off in a circle.
Line/Area Chart	Displaying data as a series of points connected by line segments.
Gauge	Measuring data off in a semicircular arch.
NVD3 Bar	Presenting data in rectangular bars with the lengths analogous to the values and choices to line up or pile up if come with multiple data sources.
NVD3 Pie	Showing data in a circle divided into slices to illustrate numerical proportion.
NVD3 Scatter Plot	Showing data as a collection of points with variables to determine the coordinates on the horizontal axis and the vertical axis.

Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

8.3 Work with Modbus RTU Climate Sensors

Items to prepare:

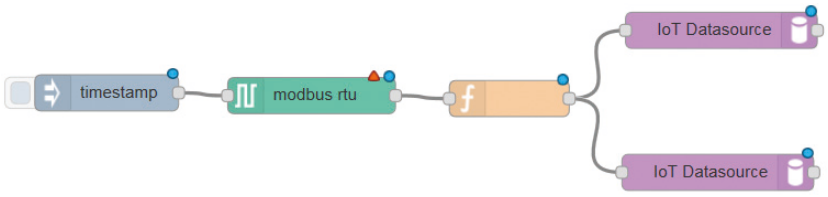
1. A set of Modbus RTU climate sensors.
2. A set of CPS 200 or CPS 100.
3. Required cables such as the power cable and the serial cable.

Prerequisite:

Every item above is well connected and turned on.

8.3.1 Plan Your Flow

1. Log onto your CPS 200/100's IP address for the IoT Studio page.
2. Add and connect 1 **inject** node, 1 **modbusrtu** node, 1 **function** node, and 2 **IoTDatasource** nodes to the workspace as shown.

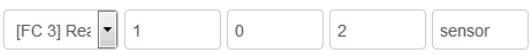


3. Double click the inject node.
   Select 

   and then click **Done**.

4. Double click the modbus rtu node, and click  .
   Click  to select a port connected to your sensor.
   In Settings, adjust each value according to the specification of your sensor such as 

   .

   Click **Update**.
   Make sure you have an FC input looks like 

   and then click **Ok**.

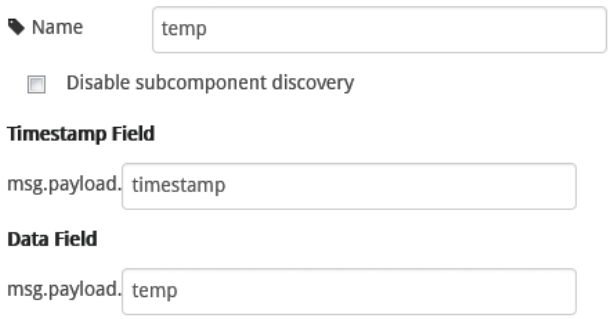
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

5. Double click the function node.
   Copy and paste the codes at right into **Function** field.

```
msg.payload.temp=msg.payload.sensor.results[0]/100;
msg.payload.humi=msg.payload.sensor.results[1]/100;
msg.payload.timestamp=new Date().getTime();
return msg;
```

   Click **Done**.

6. Double click an IoTDatasource node.
   Fill temp in the **Name** field.
   Fill timestamp in the **TimestampField**.
   Fill temp in the **DataField**.
   Click **Done**.



   Double click the other IoT Datasource node, and repeat steps above but fill humi both in the **Name** field and in the **DataField**.

   *Both of the names in the **Name** field will apply to dashboard configuration.

7. Click **Deploy** at the top right, and your flow should start running.

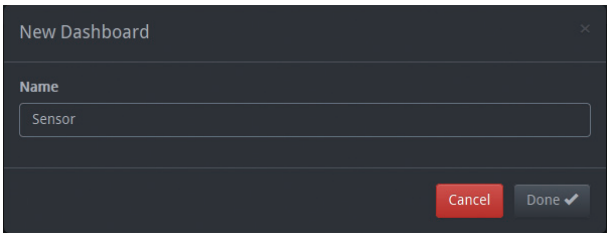
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

8.3.2 Configure Your Dashboard

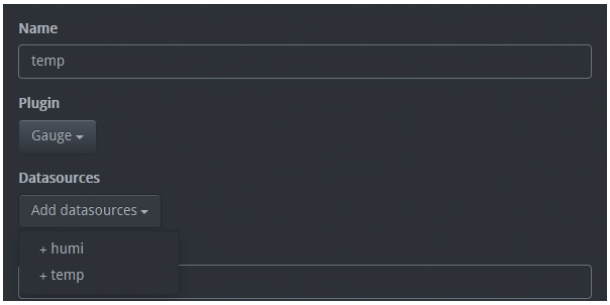
1. Log onto your CPS 200/100's IP address/dash for the dashboard page.

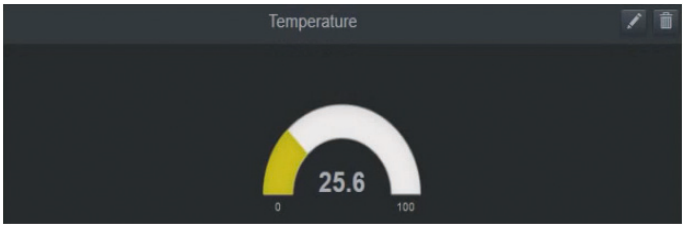
2. Click + Create New Dashboard at the top right of the page.

3. Fill Sensor in the Name field. Click **Done**.



4. Click + Create New Chart at the top right of the page. Fill temp in the **Name** field. Select **Gauge** from the drop-down menu of **Plugin**. Select **+temp** in **Datasources**. Click **Done** and you should see the chart that reflects with your datasource.

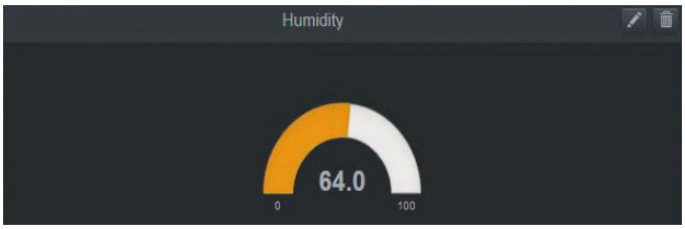




Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

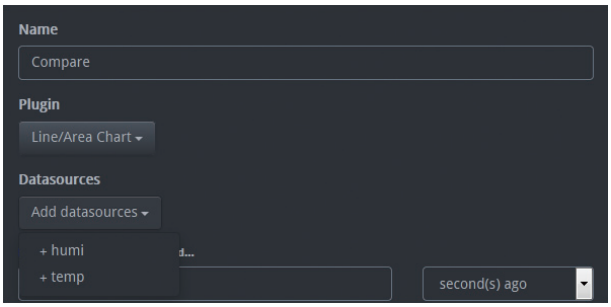
5. Repeat step 4but fill humi in the **Name** field. Select +humi in **Datasources**.

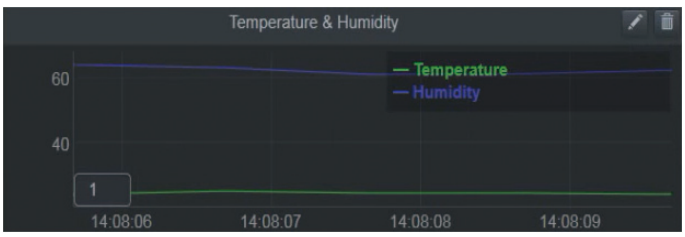
   Click **Done** and you should see another chart that reflects with your datasource.



6. Click + Create New Chart at the top right of the page. Fill Compare in the Name field. Select **Line/Area Chart** from the drop-down menu of **Plugin**. Select both of the names in **Datasources**.

   Click **Done** and you should see the chart that reflects with both of your datasources.





Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

Chapter 9: Case Studies

9.1 Modbus TCP

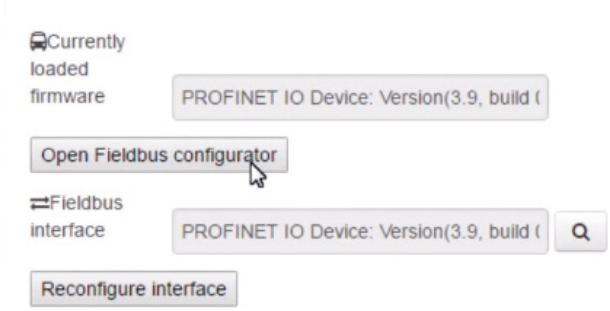
Please refer to 8.3 Work with Modbus RTU climate sensors.

9.2 Work with PROFINET Configuration

Below descriptions will guide you through the establishment of both the master and the slave in a PROFINET configuration.

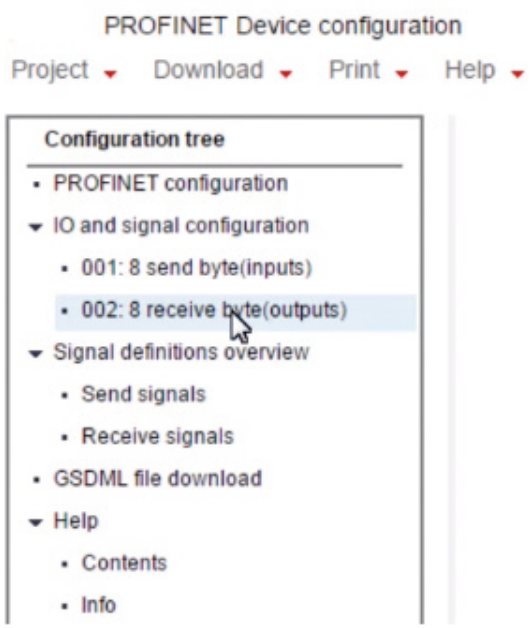
1. Log onto your CPS 200/100's IP address for the IoT Studio page.
2. Add and connect 1 **fieldbus input** node and 1 **debug** node to the workspace as shown.

3. Double click the fieldbus input node, and click  .
4. Click on **Open Fieldbus configurator** to open the main menu of **PROFINET Device configuration**.

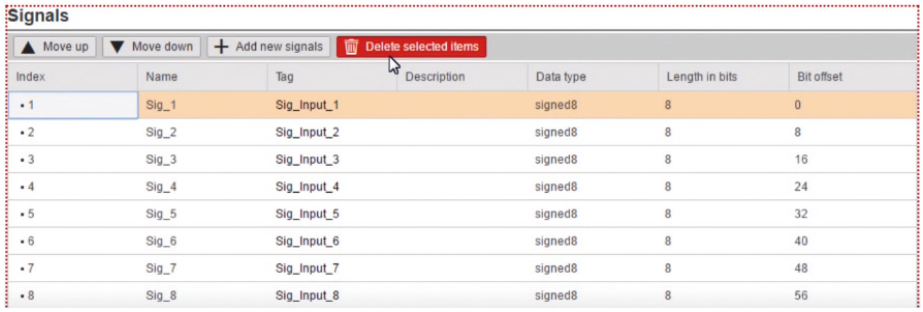


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

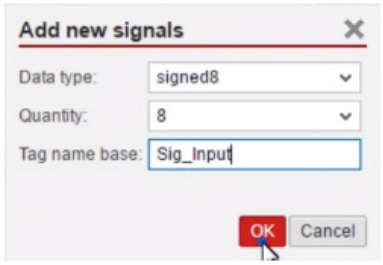
5. Click **002: 8 receive byte(outputs)** in the configuration tree to bring out **Signal configuration**.



6. Select all items in the table of **Signals** and click **Delete selected items** to clear the table.

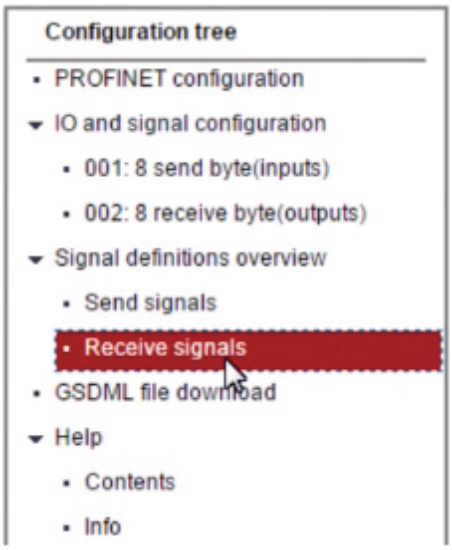


7. Click **Add new signals**. In the prompt of **Add new signals**, select signed8 in **Data type**, 8 in **Quantity**, fill Sig_Input in the field of **Tag name base**, and click **OK**.

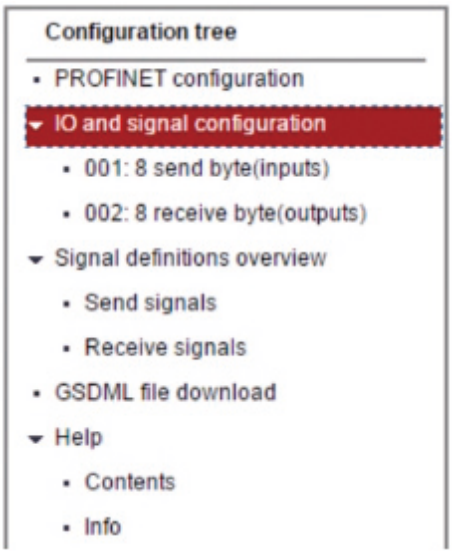


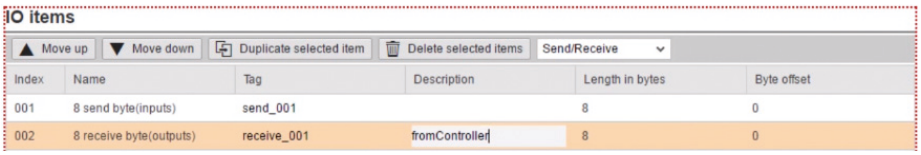
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

8. Click **Receive signals** in the configuration tree to bring out the table of **Receive signals**.



9. Click **IO and signal configuration** in the configuration tree. In the table of IO items, click the field of **Description** in the row of **Index 002**, and fill fromController.

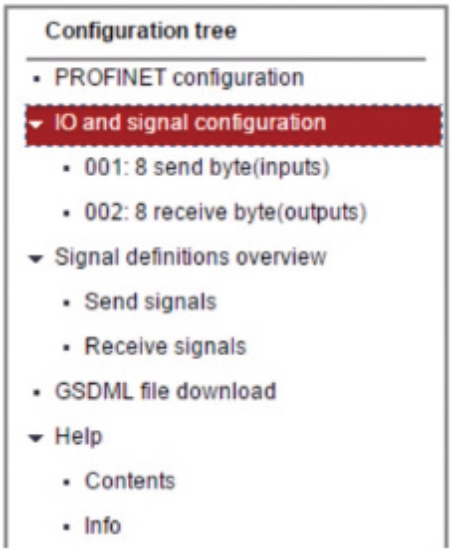


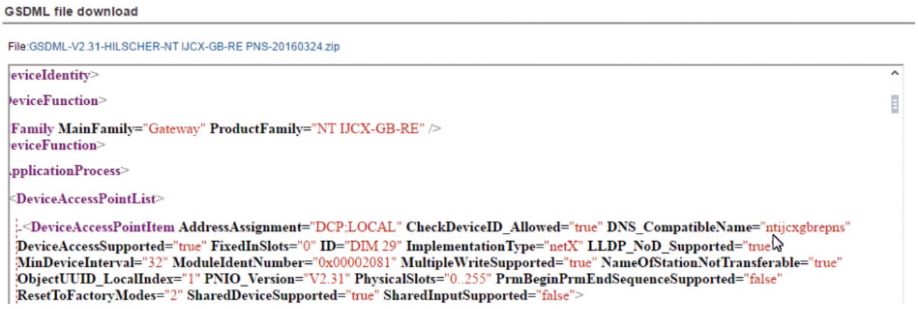


10. At the top left of the page, click **Project** > **Save** and confirm the configuration.

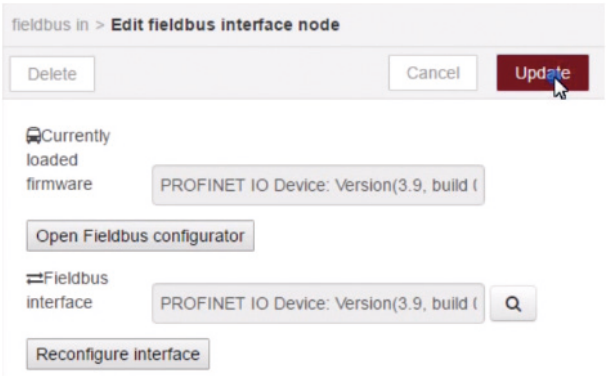
Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

11. Click **GSDML file download** in the configuration tree. Search the content of the GSDML file for **DNS_CompatibleName**, and save its value, "ntijcxgbrepns".



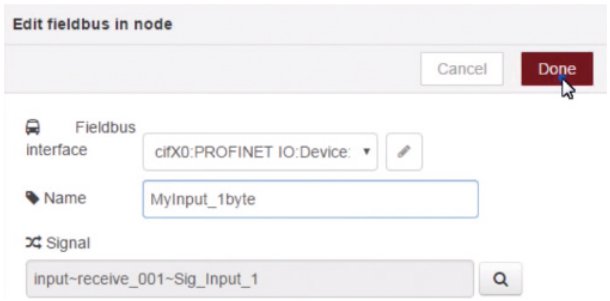


12. At the top left of the page, click **Project** > **Save** and confirm the configuration.
13. Close the page of **PROFINET Device configuration**. Go back to **Edit fieldbus interface node** and click **Update**.



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

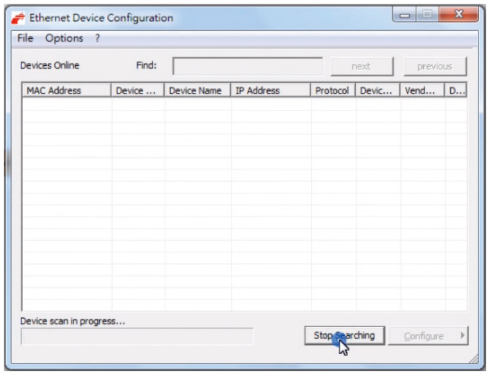
14. Click 🔍 to select **in-put-receive_001-Sig_Input_1**.
    In the field of **Name**, fill MyIn-put_1byte.
    Click **Done**.

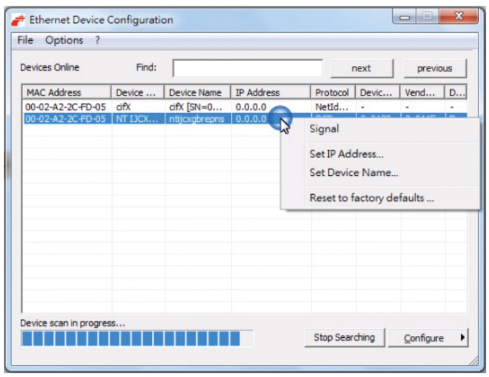


15. Connect the nodes as shown and deploy the flow.

    Until step 15, you have built up the slave in the PROFINET configuration. Continue steps below for building up the master with CODESYS.

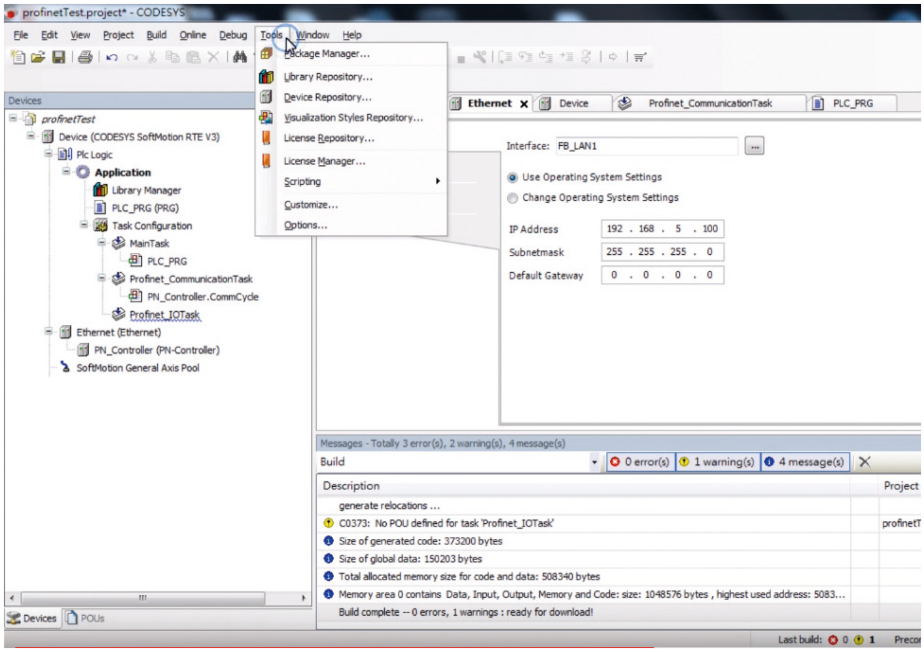
16. Launch **Ethernet Device Setup** and click **Search Devices**.
    Check devices in the list and make sure there is one device name the same as the value of **DNS_CompatibleName**.



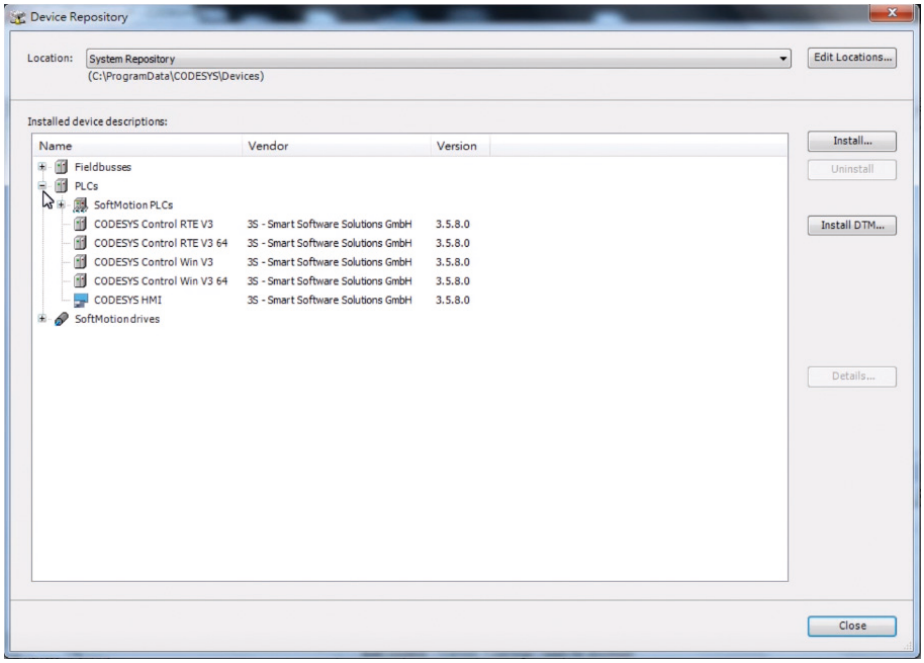


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

17. Launch CODESYS.
    Click **Tools** > **Device Repository**.

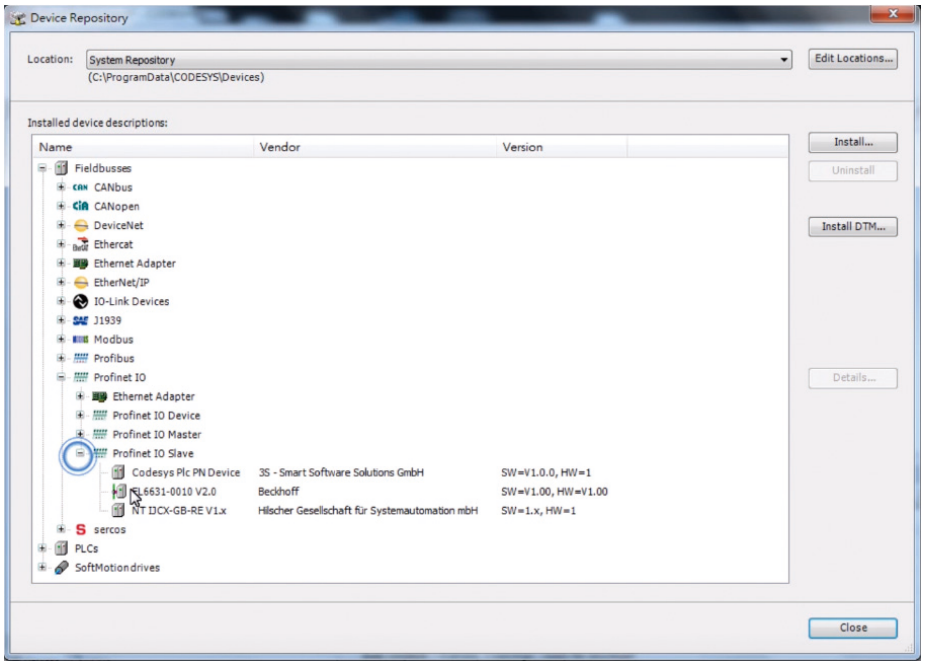


18. Expand and check PLCs.

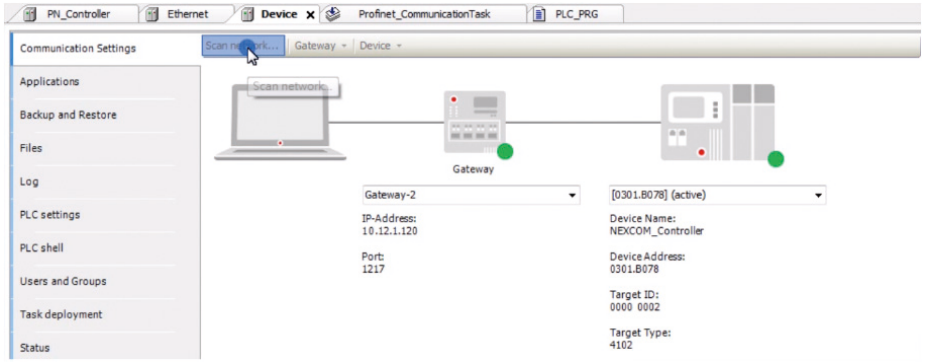


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

19. Expand **Fieldbuses**, **Profinet IO**, and click **Profinet IO Slave**.

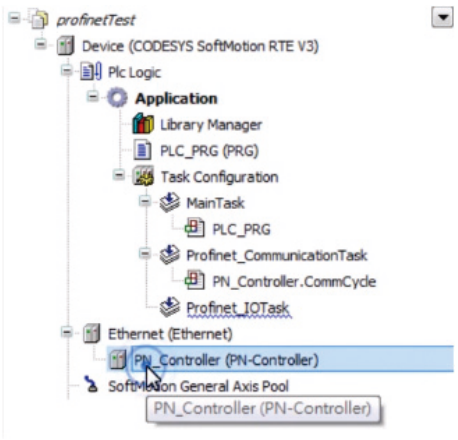


20. Close **Device Repository**.
21. Click **Scan Network** in **Device** tab and select the controller in the network path.

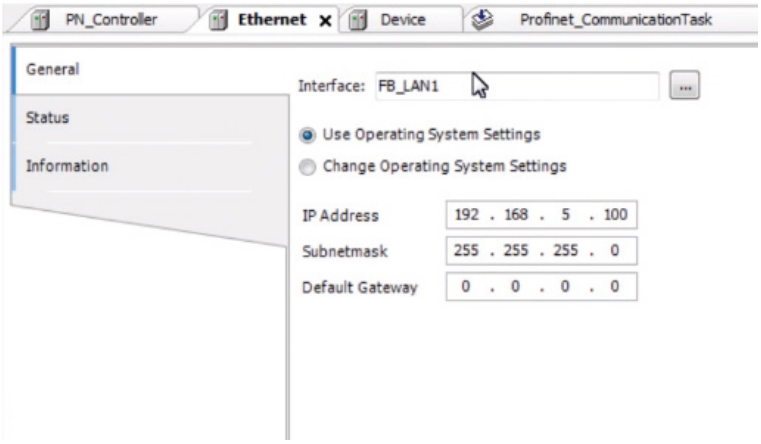


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

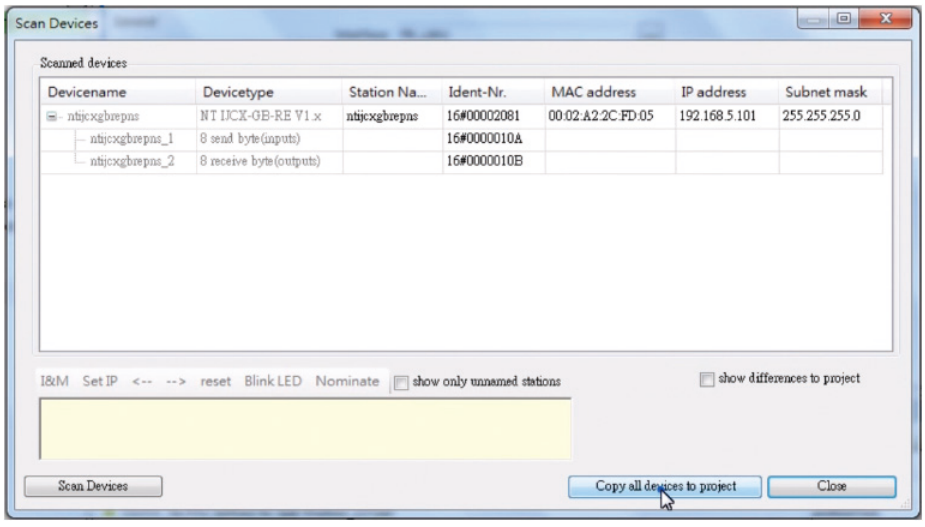
22. Right click on **PN-Controller** under **Ethernet** and click **Scan for Devices**. Make sure there is no item in the list.



23. Click the **Ethernet** tab. Make sure the name of the interface.

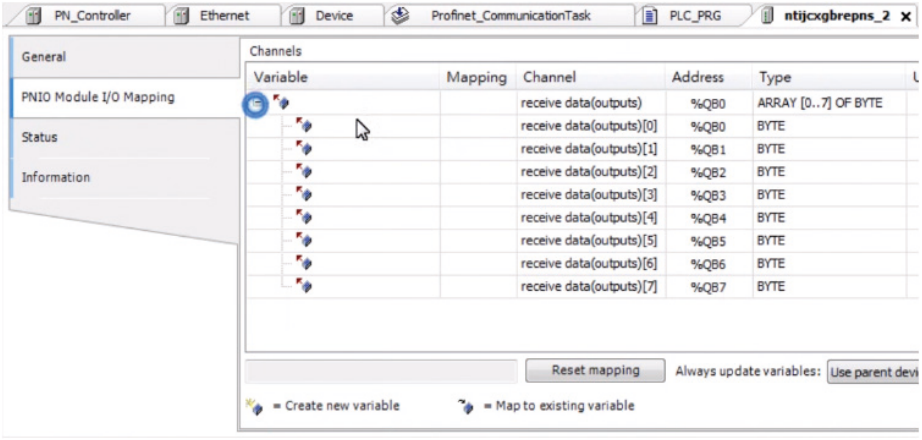


24. Right click on **PN-Controller** under **Ethernet** and click **Scan for Devices**.

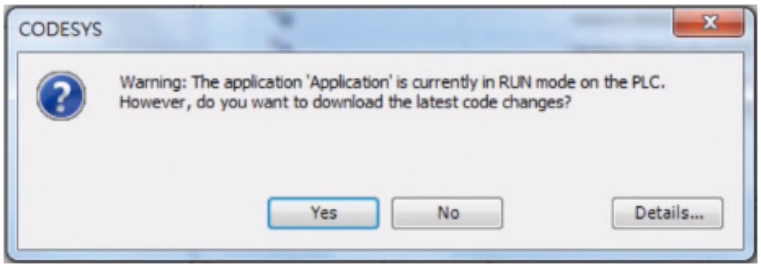


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

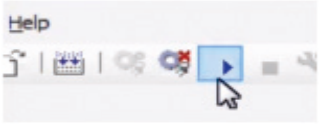
25. Click **Copy all devices to project**. "**ntijcxgbrepns**" should be added under **PN-Controller**.
26. Click **ntijcxgbrepns_2** and click **PNIO Module I/O Mapping**. Expand **Variable**.



27. Click **Build** > **Build**.
28. Click the run icon and click **Yes**.

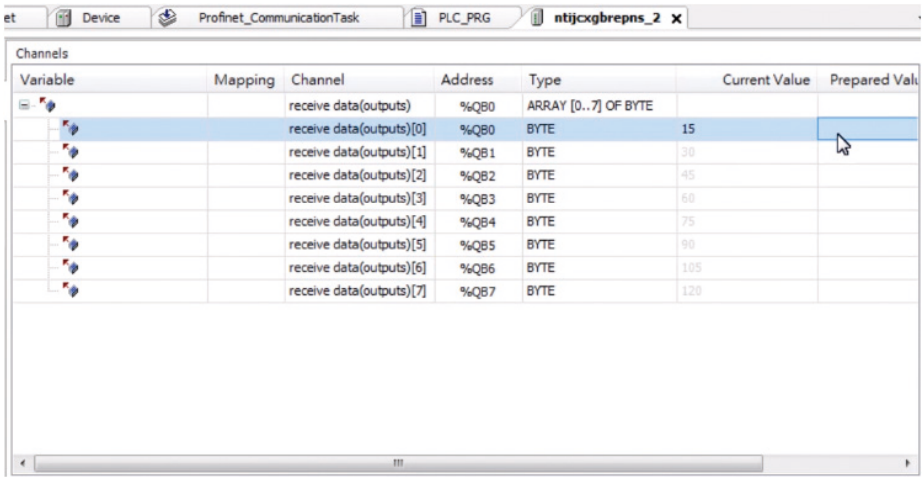


29. Click the play icon.

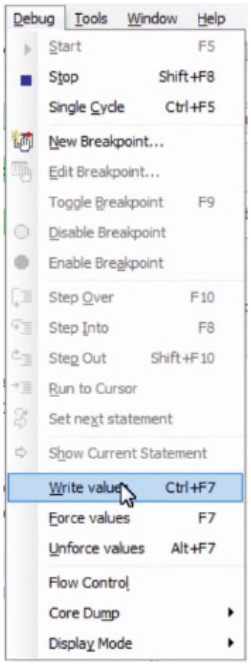


Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.

30. Click **ntijcxgbrepns_2** tab. Click **Prepared Valued** cell in the row of **receive data(outputs)[0]**.



31. Go back to Node-RED page and the debug window should keep updated with messages.
32. Go back to CODESYS.
    Click the stop icon.
    Click **ntijcxgbrepns_2**. Click **Prepared Valued** cell in the row of **receive data(outputs)[0]**. Enter a desired number.
    Click the play icon.
33. Click **Debug** > **Write** value.
    Go back to Node-RED page. The debug window should keep updated with the value starting with the desired number.



Copyright © 2017 NEXCOM International Co., Ltd. All Rights Reserved.