Connect Anybus CompactCom to IIoT Applications Using Node-Red

APPLICATION NOTE

SCM-1300–014 1.1 ENGLISH

Important User Information

Liability

Every care has been taken in the preparation of this document. Please inform HMS Industrial Networks AB of any inaccuracies or omissions. The data and illustrations found in this document are not binding. We, HMS Industrial Networks AB, reserve the right to modify our products in line with our policy of continuous product development. The information in this document is subject to change without notice and should not be considered as a commitment by HMS Industrial Networks AB. HMS Industrial Networks AB assumes no responsibility for any errors that may appear in this document.

There are many applications of this product. Those responsible for the use of this device must ensure that all the necessary steps have been taken to verify that the applications meet all performance and safety requirements including any applicable laws, regulations, codes, and standards.

HMS Industrial Networks AB will under no circumstances assume liability or responsibility for any problems that may arise as a result from the use of undocumented features, timing, or functional side effects found outside the documented scope of this product. The effects caused by any direct or indirect use of such aspects of the product are undefined, and may include e.g. compatibility issues and stability issues.

The examples and illustrations in this document are included solely for illustrative purposes. Because of the many variables and requirements associated with any particular implementation, HMS Industrial Networks AB cannot assume responsibility for actual use based on these examples and illustrations.

Intellectual Property Rights

HMS Industrial Networks AB has intellectual property rights relating to technology embodied in the product described in this document. These intellectual property rights may include patents and pending patent applications in the USA and other countries.

Trademark Acknowledgements

Anybus® is a registered trademark of HMS Industrial Networks AB. All other trademarks are the property of their respective holders.

Connect Anybus CompactCom to IIoT Applications Using Node-Red Application Note
SCM-1300–014 1.1

Table of Contents

Page

This page intentionally left blank

1 History

Revision	Date	Description	Responsible
1.00	2016-09-23	First version	MaH
1.1	2016–10–03	Moved to DOX	KeL

2 Solution Overview

Node-RED is a tool created by IBM Emerging Technologies. It is used to wire together hardware devices, APIs, and online services in new and interesting ways.

This application note will guide you through the steps how to easily integrate data from your Anybus CompactCom enabled device into Node-RED.

In Node-RED you can process the data further and connect it to different Internet of Things (IoT) services.

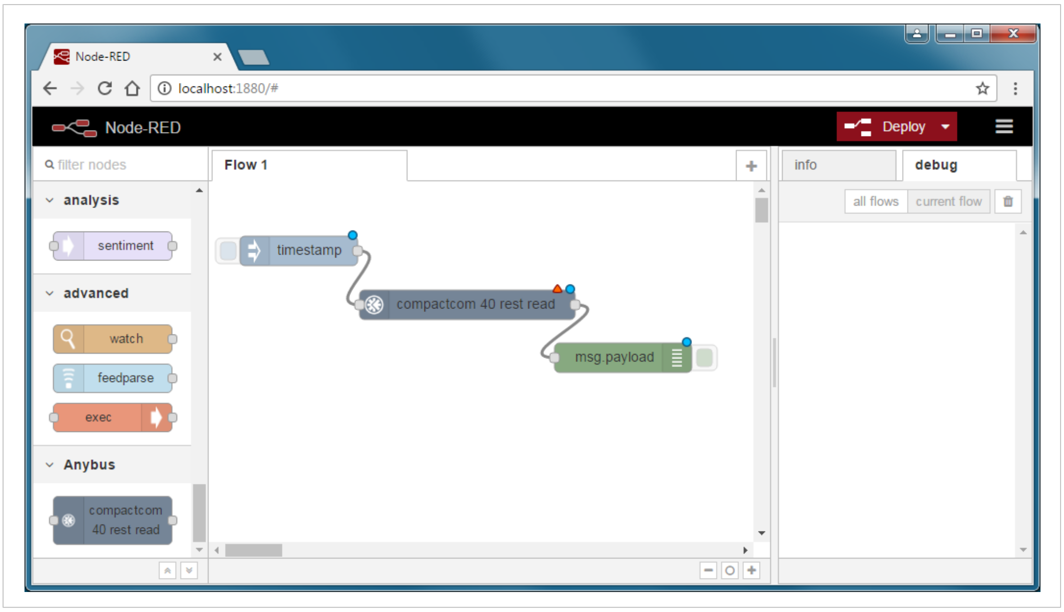


Fig. 1

3 Anybus CompactCom and Node-RED

3.1 Prerequisites

- A server running the Node-RED environment. You can install it on several platforms e.g., Linux, Mac, and Windows. See the Node-RED Getting Started guide for installation instructions. http://nodered.org/docs/getting-started/
- An Anybus CompactCom 40 series Ethernet module integrated into your application or an IXXAT INpact PCIe or PCIe mini PC interface card. The Anybus web server has to be enabled and the data of the device must be mapped into Application Data Instances (ADIs). We recommend you to set up a username and password for increased security (see the Web Server section in the network guide for your Anybus module).

3.2 Install the Anybus Nodes into Node-RED

After you have installed Node-RED, you need to install the Anybus Node-RED node. The Anybus node will access data in the Anybus CompactCom module using the Anybus JSON interface over HTTP.

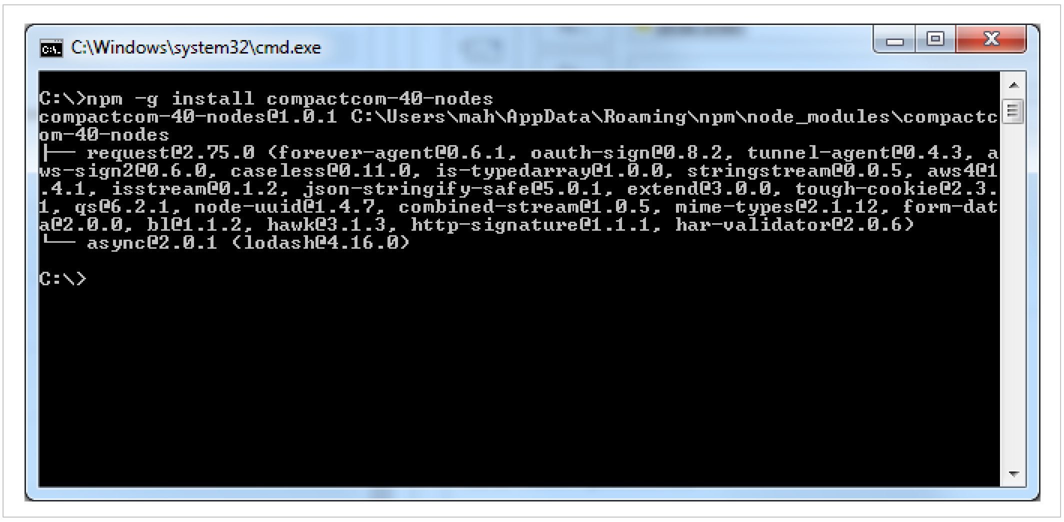
Steps to install the node:

1. Open a Command Prompt on the computer running Node-RED and enter the following command to download and install the Anybus node:

   ```
   npm -g install compactcom-40-nodes
   ```

   On Linux you need to be root or prepend the command line with `sudo`.

   The output should be similar to this screenshot:



```
C:\>npm -g install compactcom-40-nodes
compactcom-40-nodes@1.0.1 C:\Users\mah\AppData\Roaming\npm\node_modules\compactc
om-40-nodes
├── request@2.75.0 (forever-agent@0.6.1, oauth-sign@0.8.2, tunnel-agent@0.4.3, a
ws-sign2@0.6.0, caseless@0.11.0, is-typedarray@1.0.0, stringstream@0.0.5, aws4@1
.4.1, isstream@0.1.2, json-stringify-safe@5.0.1, extend@3.0.0, tough-cookie@2.3.
1, qs@6.2.1, node-uuid@1.4.7, combined-stream@1.0.5, mime-types@2.1.12, form-dat
a@2.0.0, bl@1.1.2, hawk@3.1.3, http-signature@1.1.1, har-validator@2.0.6)
└── async@2.0.1 (lodash@4.16.0)

C:\>
```

Fig. 2

2. Restart Node-RED. The Anybus node should now be listed in the left pane under the Anybus section:

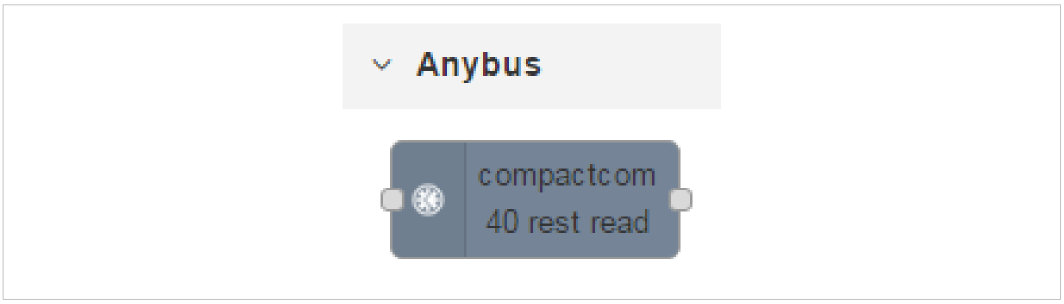


Fig. 3

3.3 Example 1 – Access Data in your Anybus Device

In this example you will connect to an Anybus enabled device and read information from the module.

Add the Anybus node (`compactcom 40 rest read`) to your working area together with an `inject` node (input group) and a `debug` node (output group).

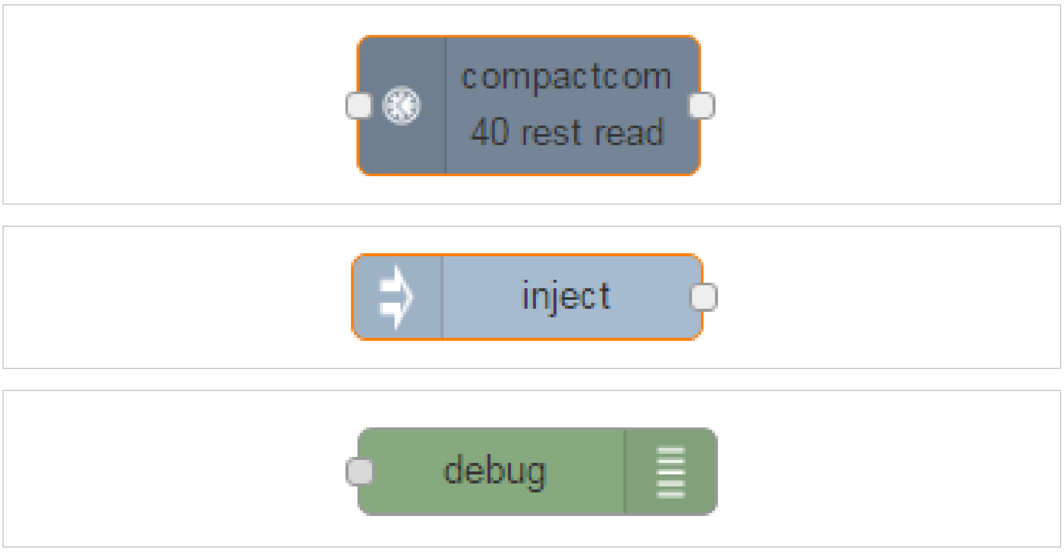


Fig. 4

Wire them together as shown in the screenshot below:

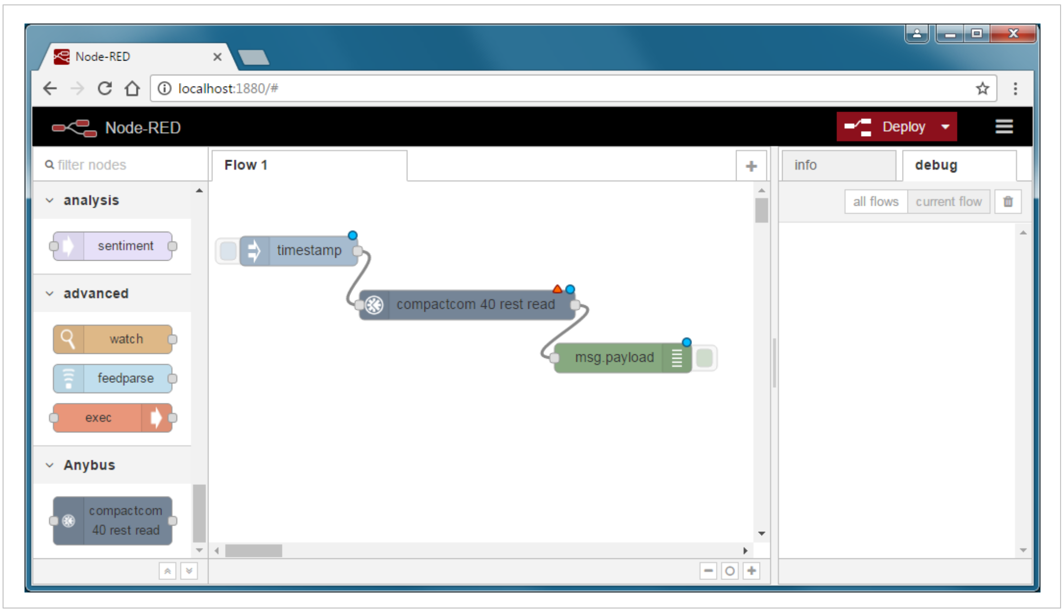
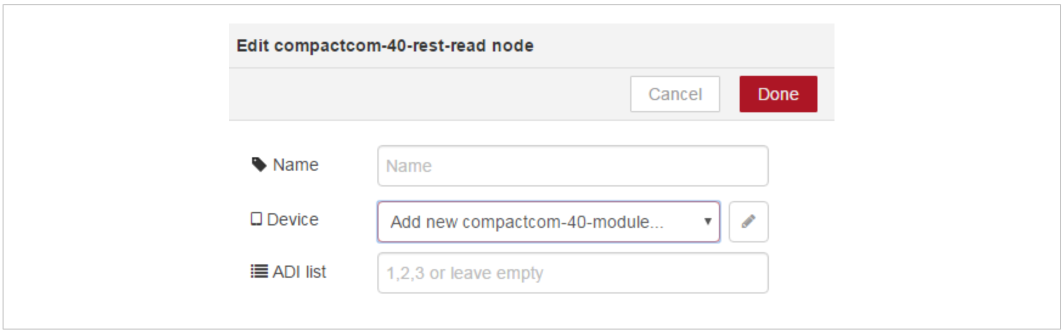


Fig. 5

3.3.1 Configure the Nodes

1. Double click on the Anybus node to bring up the dialog below.


Edit compactcom-40-rest-read node

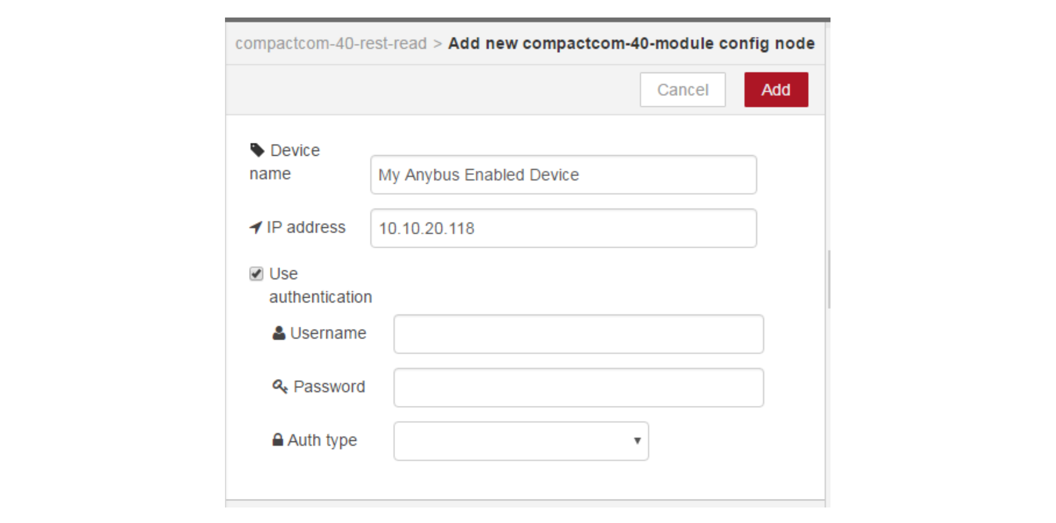
Name: Name

Device: Add new compactcom-40-module...

ADI list: 1,2,3 or leave empty

Fig. 6

2. Click the pen, to the right at the Device line, to enter the access details of your Anybus module



compactcom-40-rest-read > **Add new compactcom-40-module config node**

Device name: My Anybus Enabled Device

IP address: 10.10.20.118

☑ Use authentication

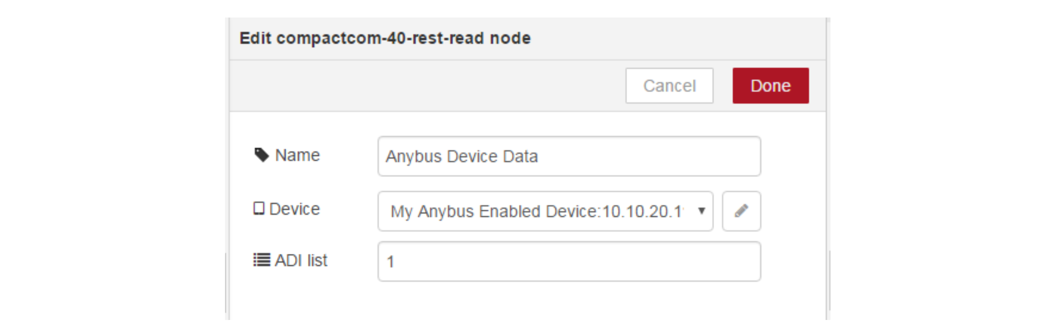
Username:

Password:

Auth type:

Fig. 7

3. Enter a name for the node, for example, My Anybus Enabled Device
4. Enter the IP address of the module (Tip: Use the HMS' IPConfig tool to find the IP address. The tool can be downloaded from the Files and Documentation section for your network module on www.anybus.com/support)
5. If you have enabled password protected access in the Anybus module; check the "Use authentication" checkbox and enter Username, Password and Authentication type.(To learn more about password protected access, see the Authorization section in the Web server chapter of the Network guide. You find the Network guide in the Files and Documentation section for your network module on www.anybus.com/support)
6. Click **Add** at the top
7. Add a Name for the data to be fetched, for example, Anybus Device Data
8. Add a comma separated list of the ADIs you want to access (the parameters in the device). In this example we access ADI 1. You can also leave the list blank to only retrieve general module information. (What ADIs that are available depend on the Anybus integration – normally every parameter in the device that is exposed to the network is mapped to an ADI)



Edit compactcom-40-rest-read node

Name: Anybus Device Data

Device: My Anybus Enabled Device:10.10.20.1

ADI list: 1

Fig. 8

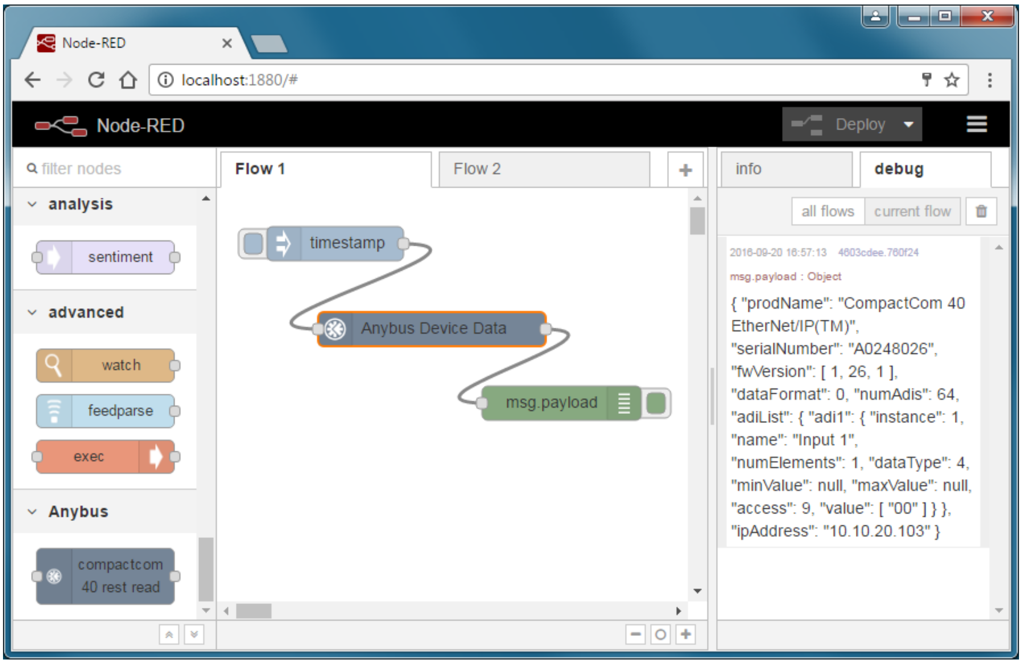
9. Click **Done**

10. Deploy the changes by clicking **Deploy** in the upper right corner of the Node-RED workspace.

Fig. 9

3.3.2 Test the Flow

1. Click the blue square on the inject node (“timestamp”) to trigger a data request to the Anybus module.
2. In the debug pane, to the right, you will get all data retrieved from the Anybus module. Static module information (like serial number) is cached to reduce data traffic, only ADI data is read in subsequent accesses. In Example 2 – you will learn how to filter out relevant data.
3. You are now done with basic set-up.



```
2016-09-20 16:57:13   4603cdee.760f24
msg.payload : Object
{ "prodName": "CompactCom 40
EtherNet/IP(TM)",
"serialNumber": "A0248026",
"fwVersion": [ 1, 26, 1 ],
"dataFormat": 0, "numAdis": 64,
"adiList": { "adi1": { "instance": 1,
"name": "Input 1",
"numElements": 1, "dataType": 4,
"minValue": null, "maxValue": null,
"access": 9, "value": [ "00" ] } },
"ipAddress": "10.10.20.103" }
```

Fig. 10

3.4 Example 2 — Filter Data

In this example we will use the configuration from Example 1 but will filter the output data for relevant parts.

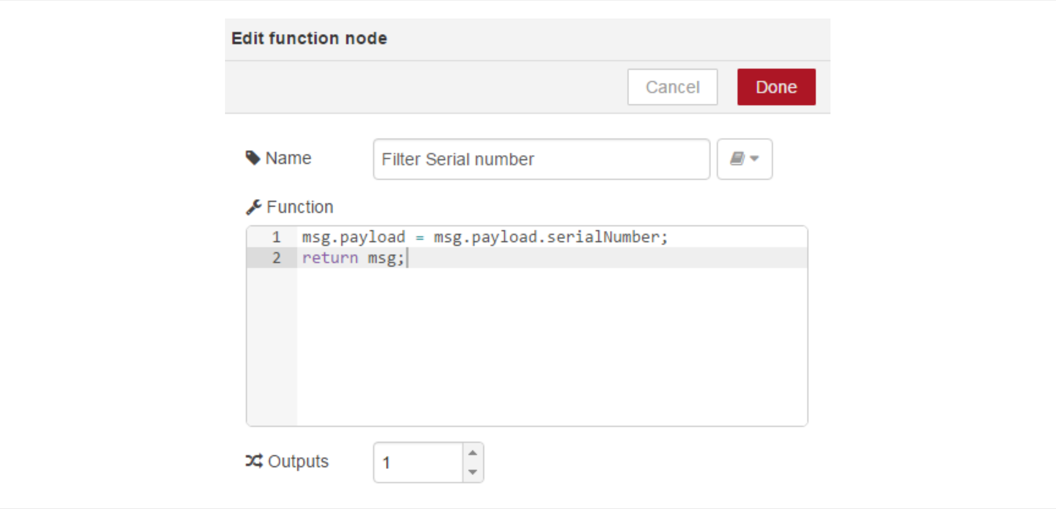
1. Add a function block to the flow:

Fig. 11

2. Double click the node to modify the content
3. Give it a name, for example, “Filter Serial number”

4. Add JavaScript code to filter out the serial number:

   `msg.payload = msg.payload.serialNumber;return msg;`



Edit function node

Name: Filter Serial number

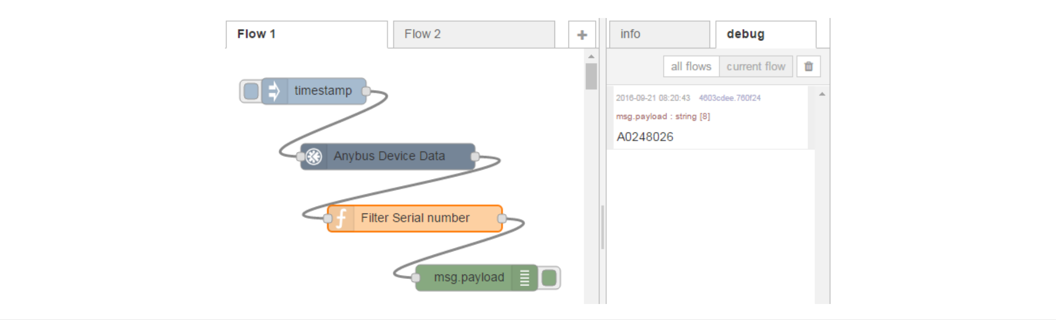
Function

```
msg.payload = msg.payload.serialNumber;
return msg;
```

Outputs: 1

Fig. 12

5. Click **Done**
6. Connect the node between the Anybus node and the Debug node
7. Deploy the changes by clicking **Deploy**
8. Press the inject node (“timestamp”)
9. You should now only receive the Anybus module’s serial number in the debug pane
10. Done



```
2016-09-21 08:20:43   4603cdee.760f24
msg.payload : string [8]
A0248026
```

Fig. 13

To filter out other parts of data, modify the last part of the JavaScript.

In the debug output in chapter *3.3.2* you can see the data structure delivered from the node.

If you, for example, would like to retrieve the data value from ADI 1, change the JavaScript code to:

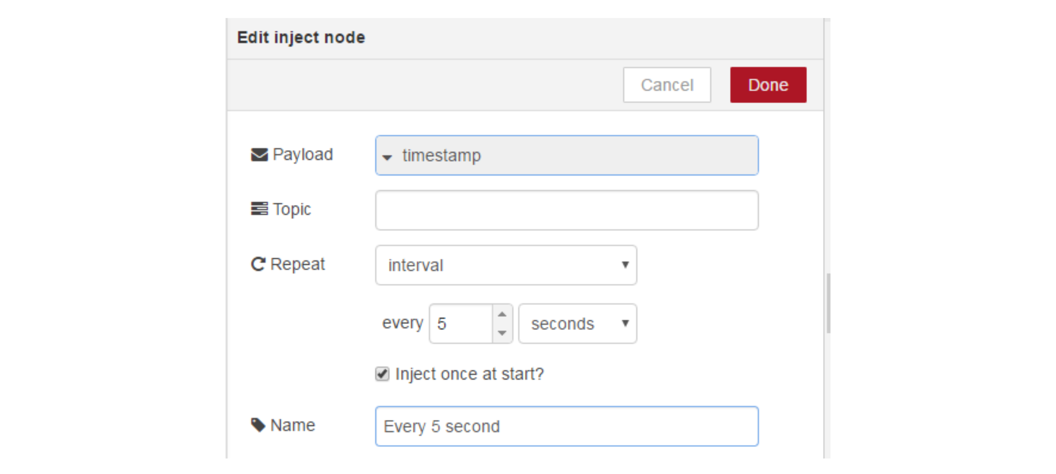
`msg.payload = msg.payload.adiList.adi1.value;`

3.5 Example 3 – Log a Message When a Parameter (ADI) in the Anybus Enabled Device Changes

1. First step is to read data cyclically:

   a. Double click the inject node (“timestamp”)

   b. Change the setting to trigger automatically every 5 seconds



Edit inject node

Payload: timestamp

Topic:

Repeat: interval

every 5 seconds

☑ Inject once at start?

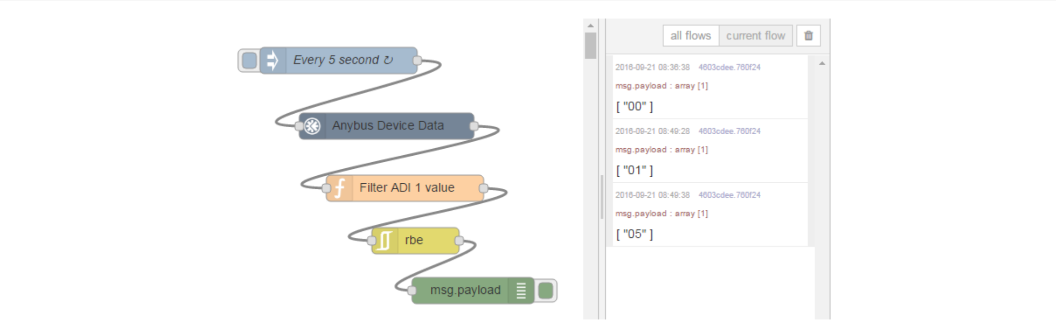
Name: Every 5 second

Fig. 14

   c. Click **done**.

2. Next step is to add a filter to only react on data changes. We will use the function node called rbe (Report by exception)

   a. Connect the node between the Filter ADI node and the debug node



```
2016-09-21 08:36:38   4603cdee.760f24
msg.payload : array [1]
[ "00" ]
2016-09-21 08:49:28   4603cdee.760f24
msg.payload : array [1]
[ "01" ]
2016-09-21 08:49:38   4603cdee.760f24
msg.payload : array [1]
[ "05" ]
```

Fig. 15

3. Deploy the changes

Now, a message will be logged every time the parameter value (ADI 1) in the Anybus enables device changes. You will find a lot of examples at http://flows.nodered.org/, showing how to create flows to connect to other IoT applications and web services.

4 More Information about Networks and Products

The latest manuals and tools, can be found on the HMS support page, www.anybus.com/support

Node-RED information and downloads are available from nodered.org.

This page intentionally left blank

HMS Industrial Networks AB
Box 4126
300 04 Halmstad, Sweden

info@hms.se