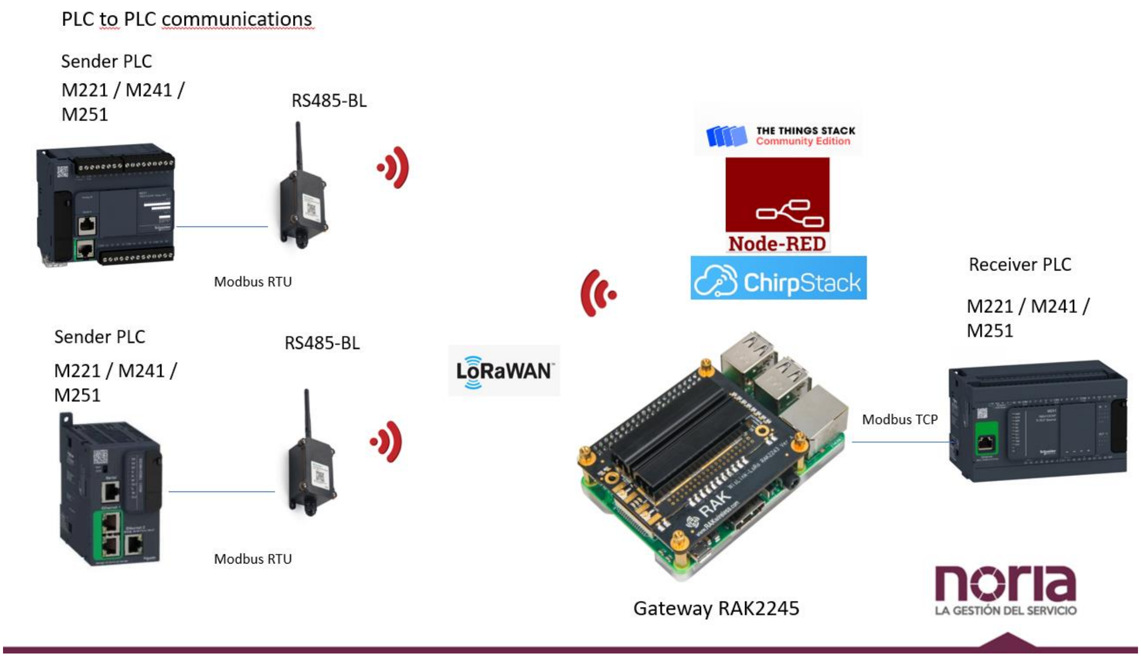
LORAWAN PLC NETWORK

PLC M221 as LoRaWAN Node



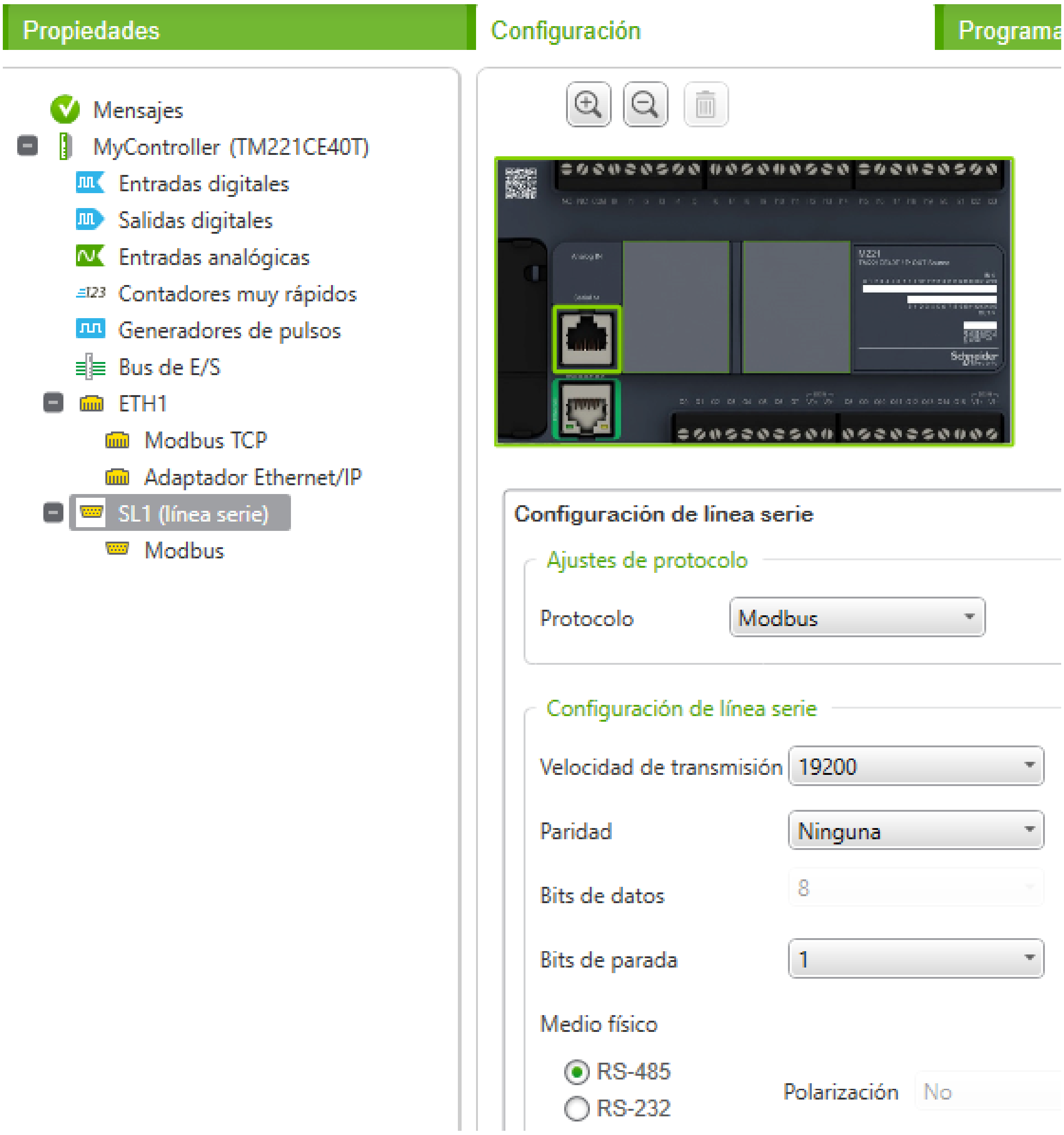
Let’s apply a Modbus to LoRaWAN converter like RS485-BL or RS485-LN

Using a RJ45 patchcord with free wires for the serial Modbus RTU port.

Blue A+

White B-

Configuring PLC Modbus settings



- ✓ Mensajes
- ⊟ MyController (TM221CE40T)
  - Entradas digitales
  - Salidas digitales
  - Entradas analógicas
  - Contadores muy rápidos
  - Generadores de pulsos
  - Bus de E/S
  - ⊟ ETH1
    - Modbus TCP
    - Adaptador Ethernet/IP
  - ⊟ SL1 (línea serie)
    - Modbus

Configuración de línea serie

Ajustes de protocolo

Protocolo: Modbus

Configuración de línea serie

Velocidad de transmisión: 19200

Paridad: Ninguna

Bits de datos: 8

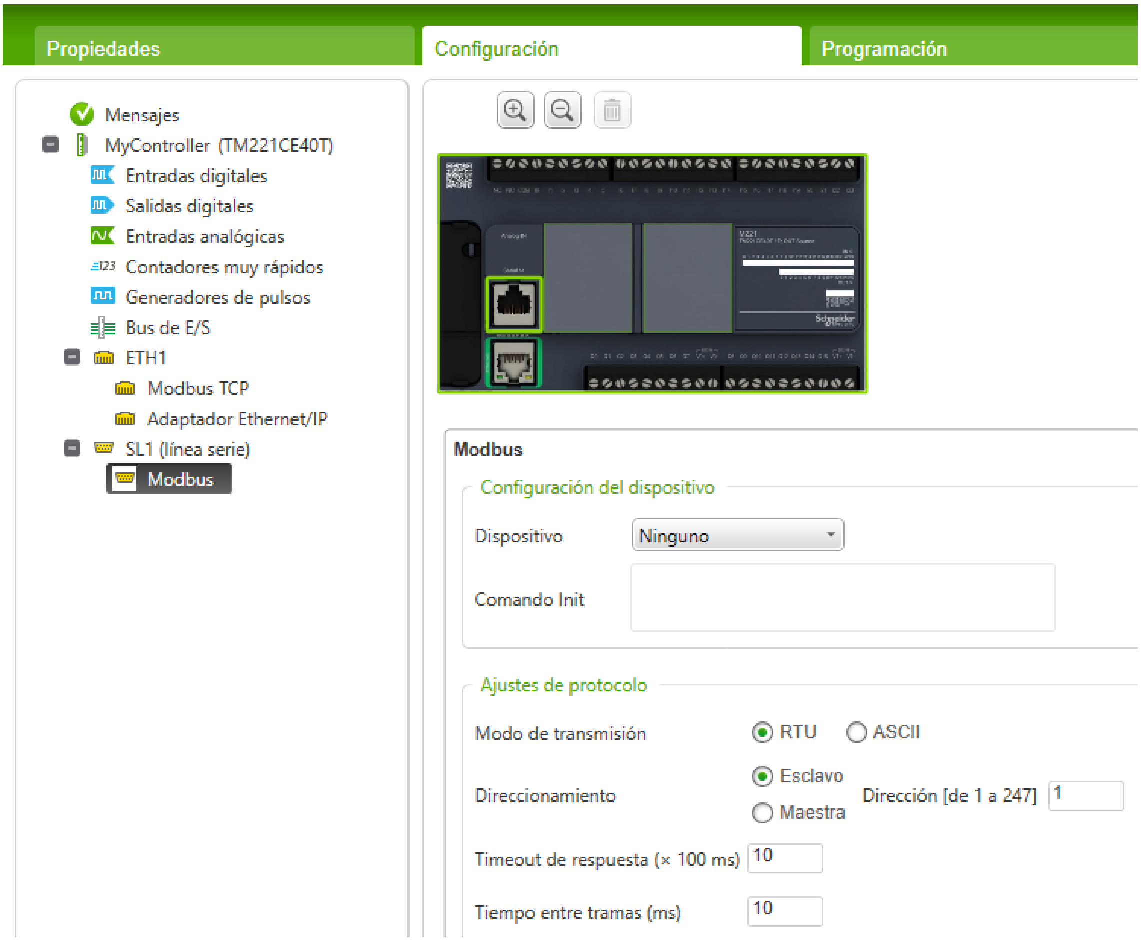
Bits de parada: 1

Medio físico

● RS-485

○ RS-232

Polarización: No



Modbus

Configuración del dispositivo

Dispositivo: Ninguno

Comando Init:

Ajustes de protocolo

Modo de transmisión: ● RTU ○ ASCII

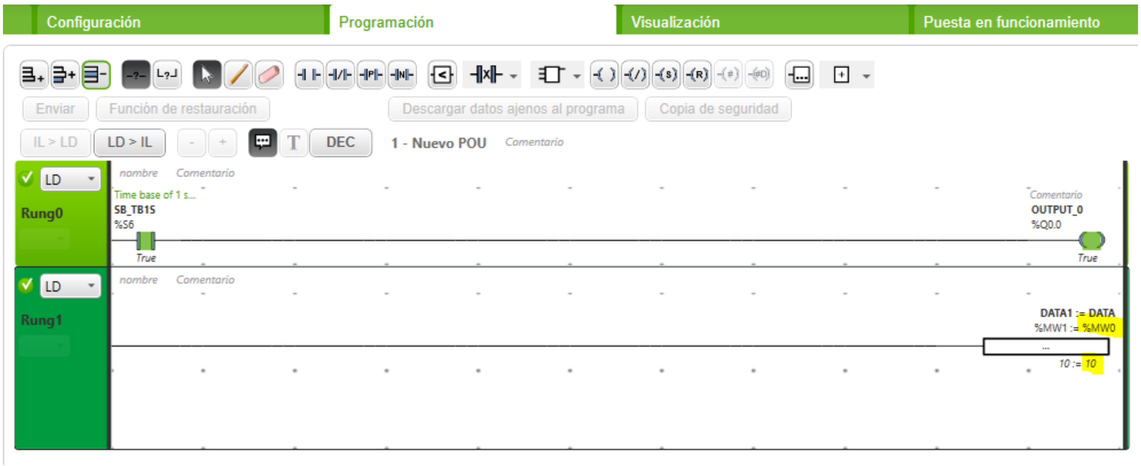
Direccionamiento: ● Esclavo ○ Maestra — Dirección [de 1 a 247]: 1

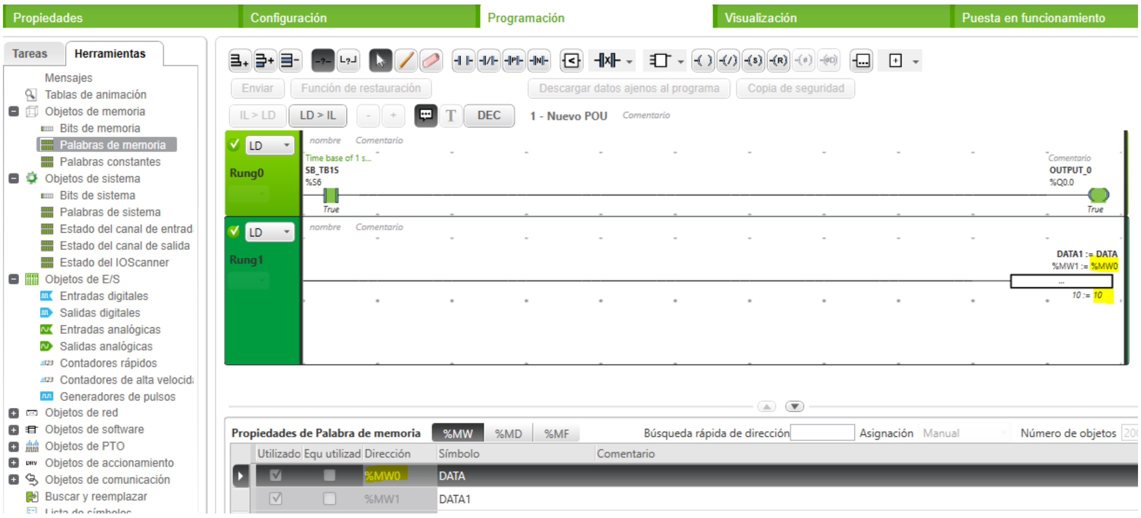
Timeout de respuesta (× 100 ms): 10

Tiempo entre tramas (ms): 10

Reading %MW0 with M221

Let's have some value in %MW0





Propiedades de Palabra de memoria %MW %MD %MF Búsqueda rápida de dirección Asignación Manual Número de objetos

Utilizado	Equ utilizad	Dirección	Símbolo	Comentario
☑	□	%MW0	DATA	
☑	□	%MW1	DATA1	

For example inserting the value of %MW0 with animation tables

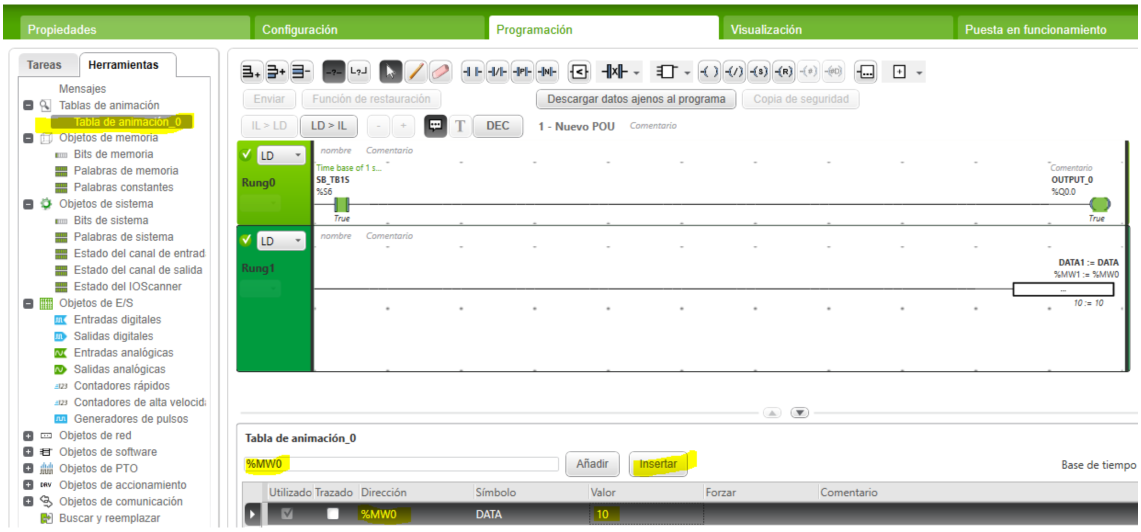


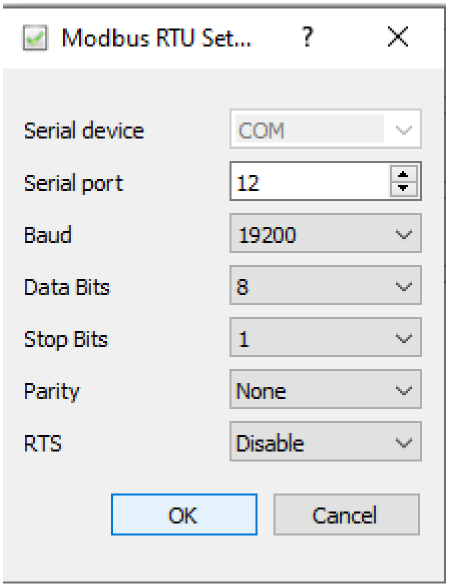
Tabla de animación_0

%MW0 Añadir Insertar Base de tiempo

Utilizado	Trazado	Dirección	Símbolo	Valor	Forzar	Comentario
☑	□	%MW0	DATA	10		

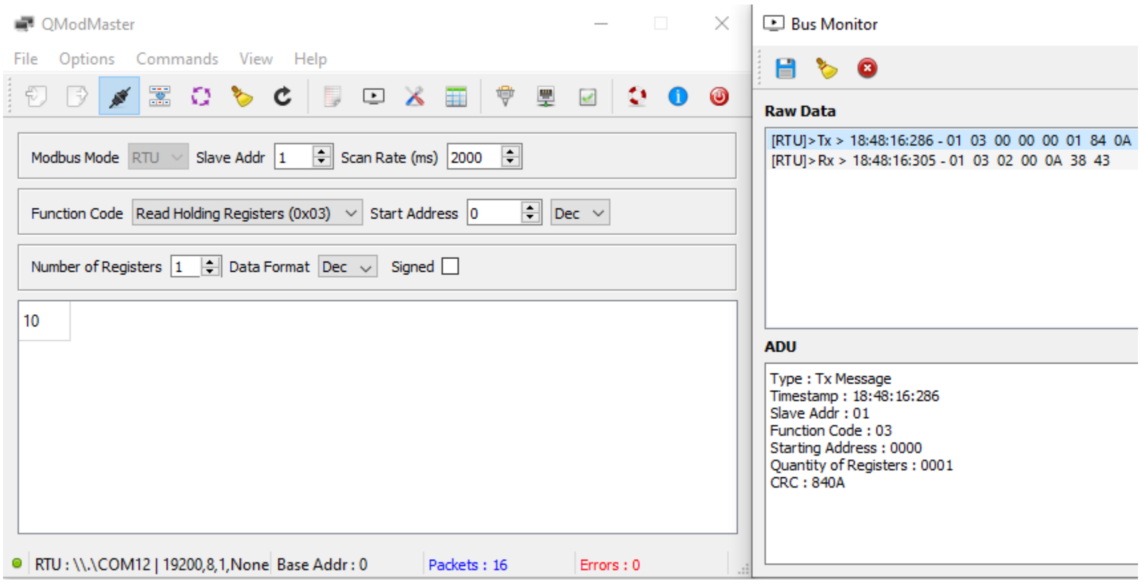
Let’s read with Qmod Master

With this settings



Modbus RTU Set...

Serial device	COM
Serial port	12
Baud	19200
Data Bits	8
Stop Bits	1
Parity	None
RTS	Disable



Modbus Mode RTU Slave Addr 1 Scan Rate (ms) 2000

Function Code Read Holding Registers (0x03) Start Address 0 Dec

Number of Registers 1 Data Format Dec Signed □

10

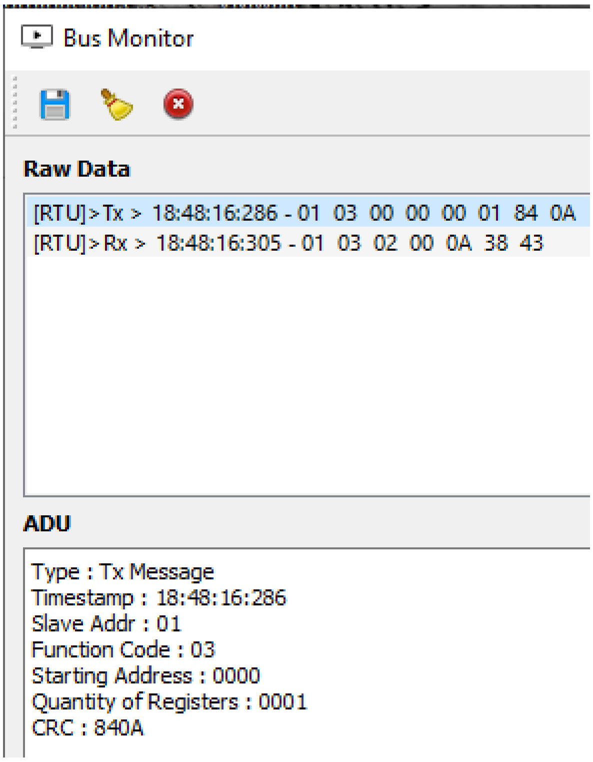
RTU : \\.\COM12 | 19200,8,1,None Base Addr : 0 Packets : 16 Errors : 0

Raw Data

```
[RTU]>Tx > 18:48:16:286 - 01  03  00  00  00  01  84  0A
[RTU]>Rx > 18:48:16:305 - 01  03  02  00  0A  38  43
```

ADU

```
Type : Tx Message
Timestamp : 18:48:16:286
Slave Addr : 01
Function Code : 03
Starting Address : 0000
Quantity of Registers : 0001
CRC : 840A
```

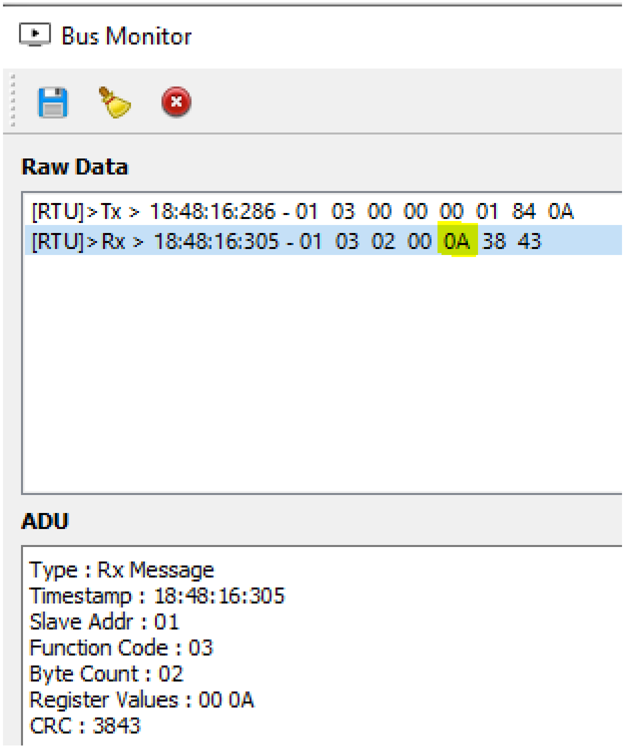


Raw Data

```
[RTU]>Tx > 18:48:16:286 - 01  03  00  00  00  01  84  0A
[RTU]>Rx > 18:48:16:305 - 01  03  02  00  0A  38  43
```

ADU

```
Type : Tx Message
Timestamp : 18:48:16:286
Slave Addr : 01
Function Code : 03
Starting Address : 0000
Quantity of Registers : 0001
CRC : 840A
```



Raw Data

```
[RTU]>Tx > 18:48:16:286 - 01  03  00  00  00  01  84  0A
[RTU]>Rx > 18:48:16:305 - 01  03  02  00  0A  38  43
```

ADU

```
Type : Rx Message
Timestamp : 18:48:16:305
Slave Addr : 01
Function Code : 03
Byte Count : 02
Register Values : 00 0A
CRC : 3843
```

So the right values for the Dragino RS485-BL will be:

AT+COMMAND1=01 03 00 00 00 01,1

AT+DATACUT1=7,1,4+5

The things Stack

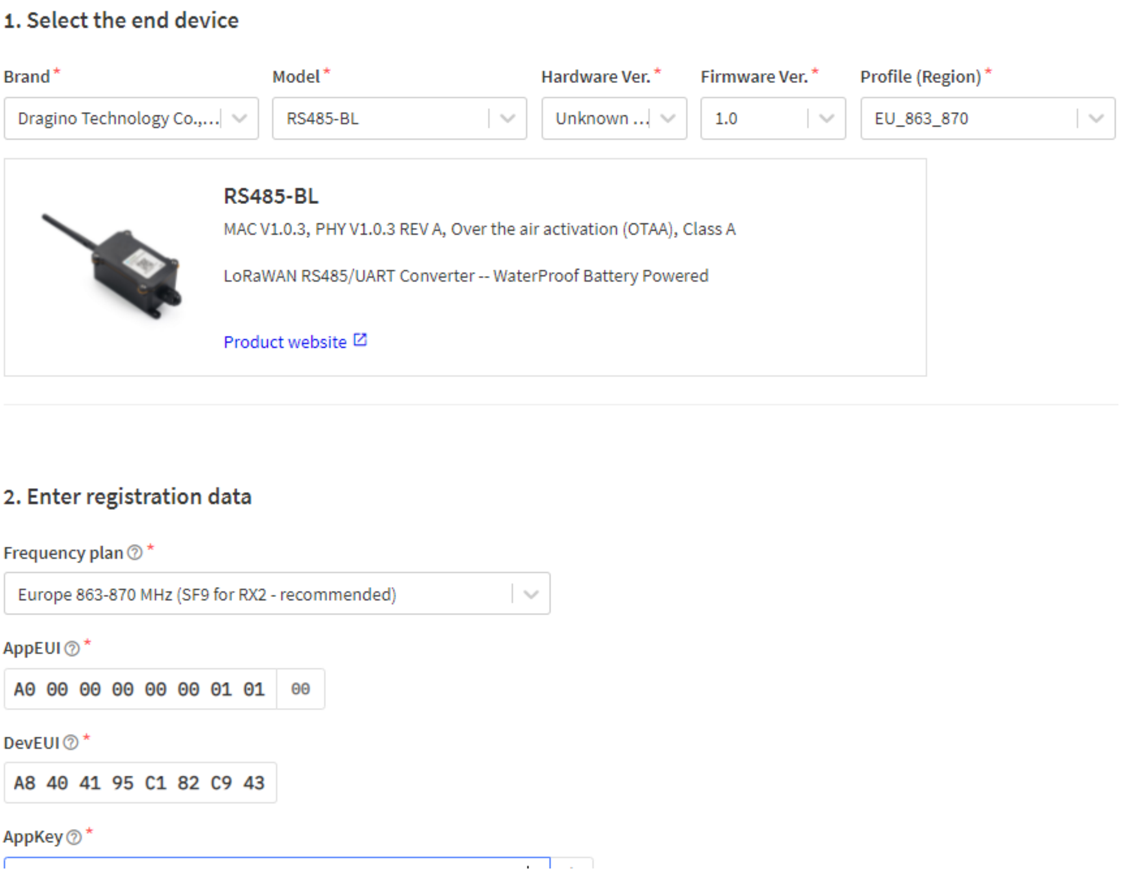
Let’s setup the Dragino on TTS

DevEui= A8 40 41 95 C1 82 C9 43

AT+DEUI=?

AT+APPKEY=?

AT+APPEUI=?



1. Select the end device

Brand *	Model *	Hardware Ver. *	Firmware Ver. *	Profile (Region) *
Dragino Technology Co.,...	RS485-BL	Unknown ...	1.0	EU_863_870

RS485-BL

MAC V1.0.3, PHY V1.0.3 REV A, Over the air activation (OTAA), Class A

LoRaWAN RS485/UART Converter -- WaterProof Battery Powered

Product website

2. Enter registration data

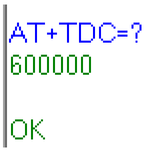
Frequency plan *
Europe 863-870 MHz (SF9 for RX2 - recommended)

AppEUI *
A0 00 00 00 00 00 01 01 | 00

DevEUI *
A8 40 41 95 C1 82 C9 43

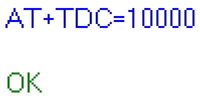
AppKey *

Let’s change the period of messages

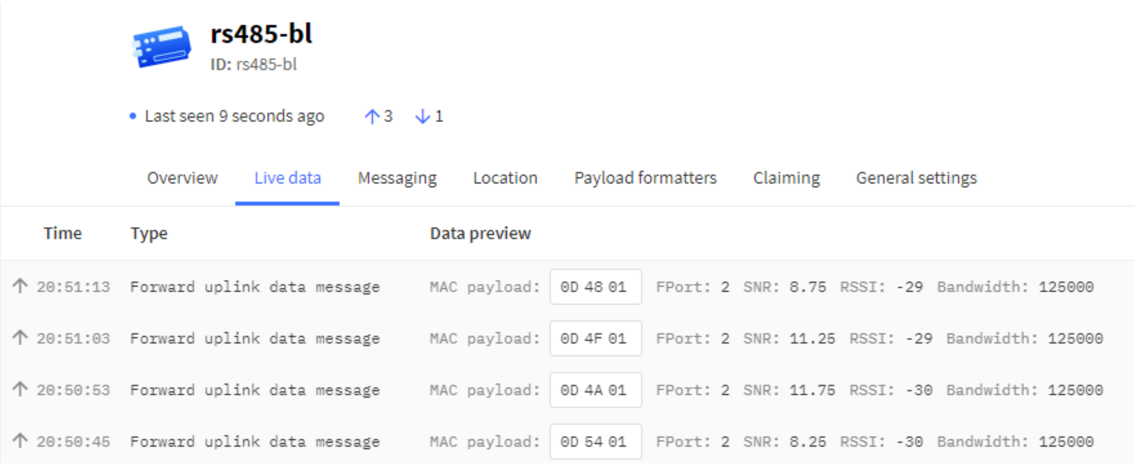


```
AT+TDC=?
600000

OK
```

To 10 seconds



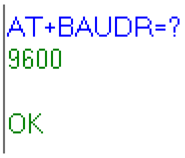
```
AT+TDC=10000

OK
```



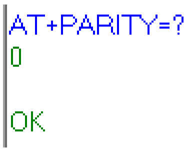
rs485-bl
ID: rs485-bl

• Last seen 9 seconds ago ↑3 ↓1

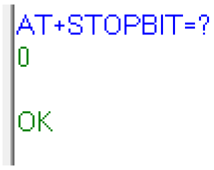
Time	Type	Data preview
↑ 20:51:13	Forward uplink data message	MAC payload: 0D 48 01 FPort: 2 SNR: 8.75 RSSI: -29 Bandwidth: 125000
↑ 20:51:03	Forward uplink data message	MAC payload: 0D 4F 01 FPort: 2 SNR: 11.25 RSSI: -29 Bandwidth: 125000
↑ 20:50:53	Forward uplink data message	MAC payload: 0D 4A 01 FPort: 2 SNR: 11.75 RSSI: -30 Bandwidth: 125000
↑ 20:50:45	Forward uplink data message	MAC payload: 0D 54 01 FPort: 2 SNR: 8.25 RSSI: -30 Bandwidth: 125000



```
AT+BAUDR=?
9600

OK
```

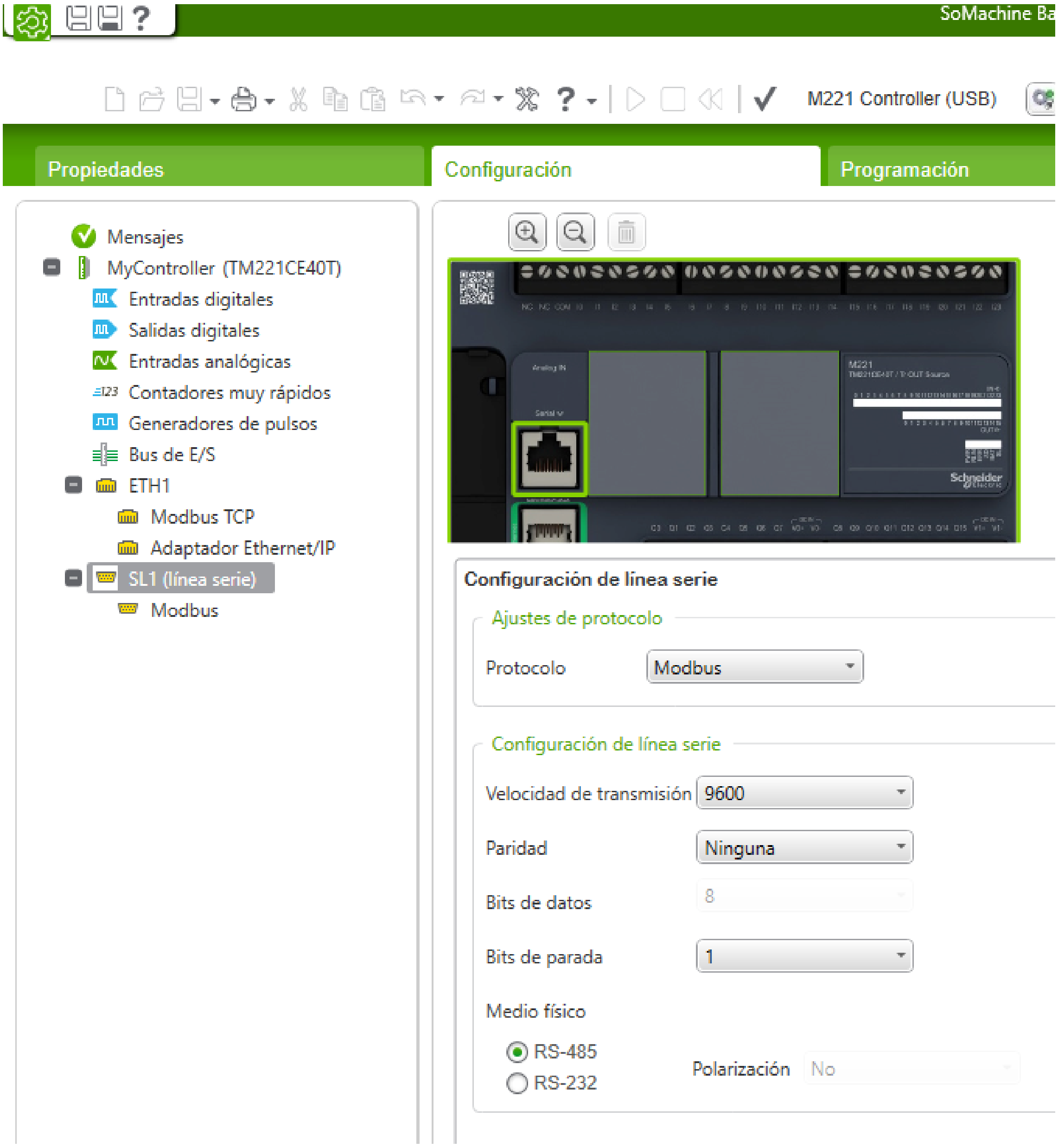


```
AT+PARITY=?
0

OK
```



```
AT+STOPBIT=?
0

OK
```

Let’s change the PLC serial speed to 9600 so it is the default value from Dragino



Configuración de línea serie

Ajustes de protocolo

Protocolo: Modbus

Configuración de línea serie

Velocidad de transmisión: 9600

Paridad: Ninguna

Bits de datos: 8

Bits de parada: 1

Medio físico

● RS-485

○ RS-232

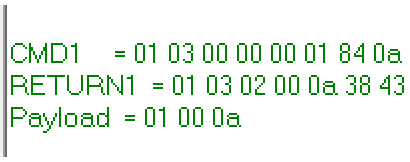
Polarización: No

You can find the PLC code here

https://github.com/xavierflorensa/Schneider-M221-as-LoRaWAN-node

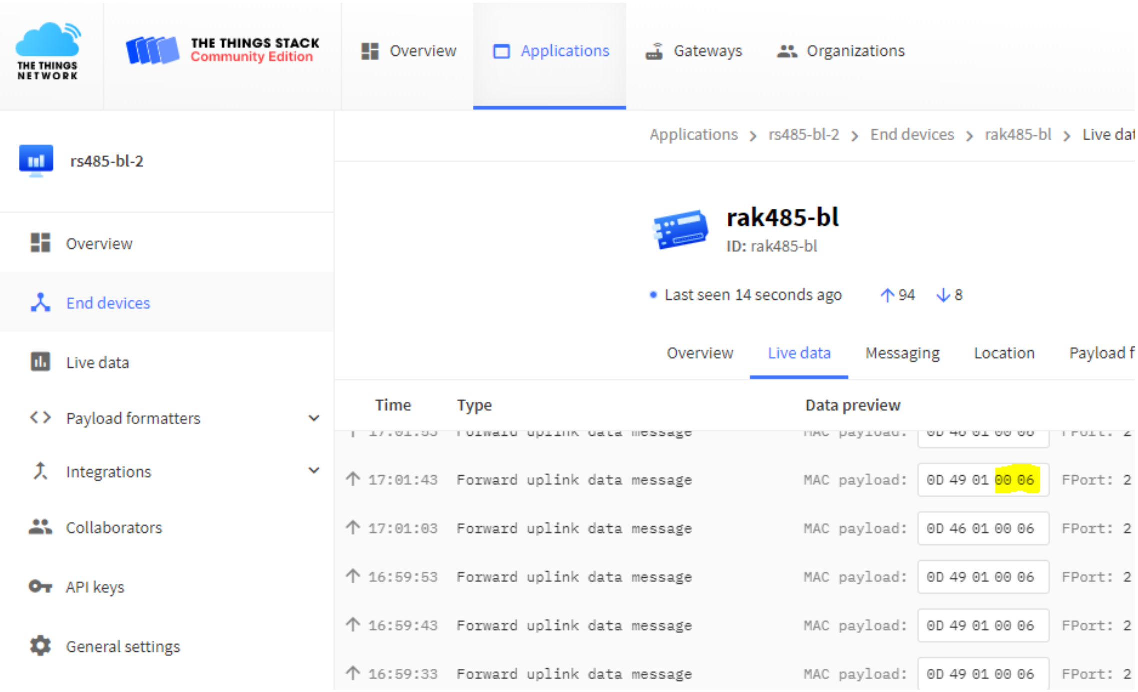
We test now with Dragino RS485 connected to the PLC

It Works



```
CMD1     = 01 03 00 00 00 01 84 0a
RETURN1  = 01 03 02 00 0a 38 43
Payload  = 01 00 0a
```

Let’s change the value to 6

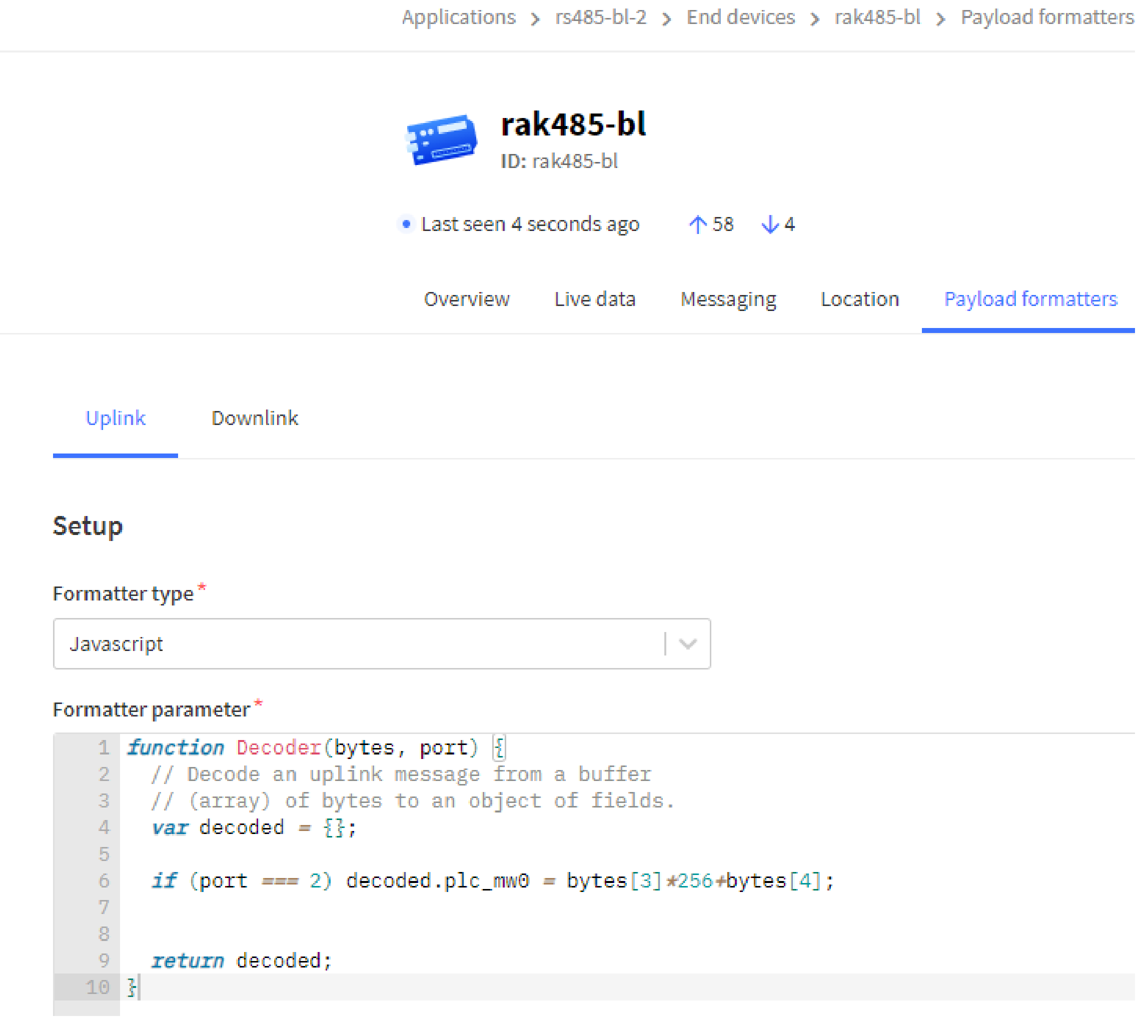


rak485-bl
ID: rak485-bl

Last seen 14 seconds ago ↑94 ↓8

```
Time      Type                          Data preview
17:01:53  Forward uplink data message   MAC payload: 0D 46 01 00 06  FPort: 2
17:01:43  Forward uplink data message   MAC payload: 0D 49 01 00 06  FPort: 2
17:01:03  Forward uplink data message   MAC payload: 0D 46 01 00 06  FPort: 2
16:59:53  Forward uplink data message   MAC payload: 0D 49 01 00 06  FPort: 2
16:59:43  Forward uplink data message   MAC payload: 0D 49 01 00 06  FPort: 2
16:59:33  Forward uplink data message   MAC payload: 0D 49 01 00 06  FPort: 2
```

Let’s change payload decoder


Applications > rs485-bl-2 > End devices > rak485-bl > Payload formatters
Overview
Live data
Messaging
Location
Payload formatters
Uplink
Downlink

rak485-bl
ID: rak485-bl

Last seen 4 seconds ago ↑58 ↓4

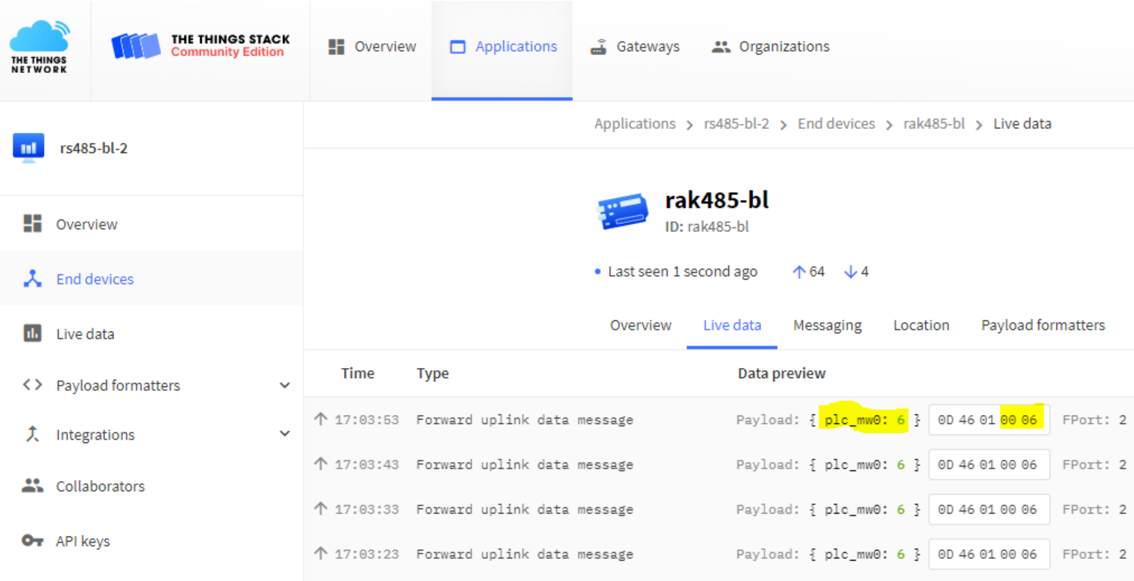
Setup

Formatter type*: Javascript

Formatter parameter*

```
function Decoder(bytes, port) {
  // Decode an uplink message from a buffer
  // (array) of bytes to an object of fields.
  var decoded = {};

  if (port === 2) decoded.plc_mw0 = bytes[3]*256+bytes[4];


  return decoded;
}
```

```
function Decoder(bytes, port) {
  // Decode an uplink message from a buffer
  // (array) of bytes to an object of fields.
  var decoded = {};

  if (port === 2) decoded.plc_mw0 = bytes[3]*256+bytes[4];



  return decoded;
}
```



Applications > rs485-bl-2 > End devices > rak485-bl > Live data

rak485-bl
ID: rak485-bl

Last seen 1 second ago ↑ 64 ↓ 4

Time	Type	Data preview
↑ 17:03:53	Forward uplink data message	Payload: { plc_mw0: 6 } 0D 46 01 00 06 FPort: 2
↑ 17:03:43	Forward uplink data message	Payload: { plc_mw0: 6 } 0D 46 01 00 06 FPort: 2
↑ 17:03:33	Forward uplink data message	Payload: { plc_mw0: 6 } 0D 46 01 00 06 FPort: 2
↑ 17:03:23	Forward uplink data message	Payload: { plc_mw0: 6 } 0D 46 01 00 06 FPort: 2

Now you can get these data thru mqtt by Edge computing on the receiver PLC and inject per Modbus to it.

Now let's inject the sender PLC input values on the receiver PLC and Chirpstack network server

Receiver PLC is connected to a Linux machine with

Gateway hardware

CHIRP STACK network server 192.168.1.105

Node-RED 192.168.1.105

Fixed Gateway IP

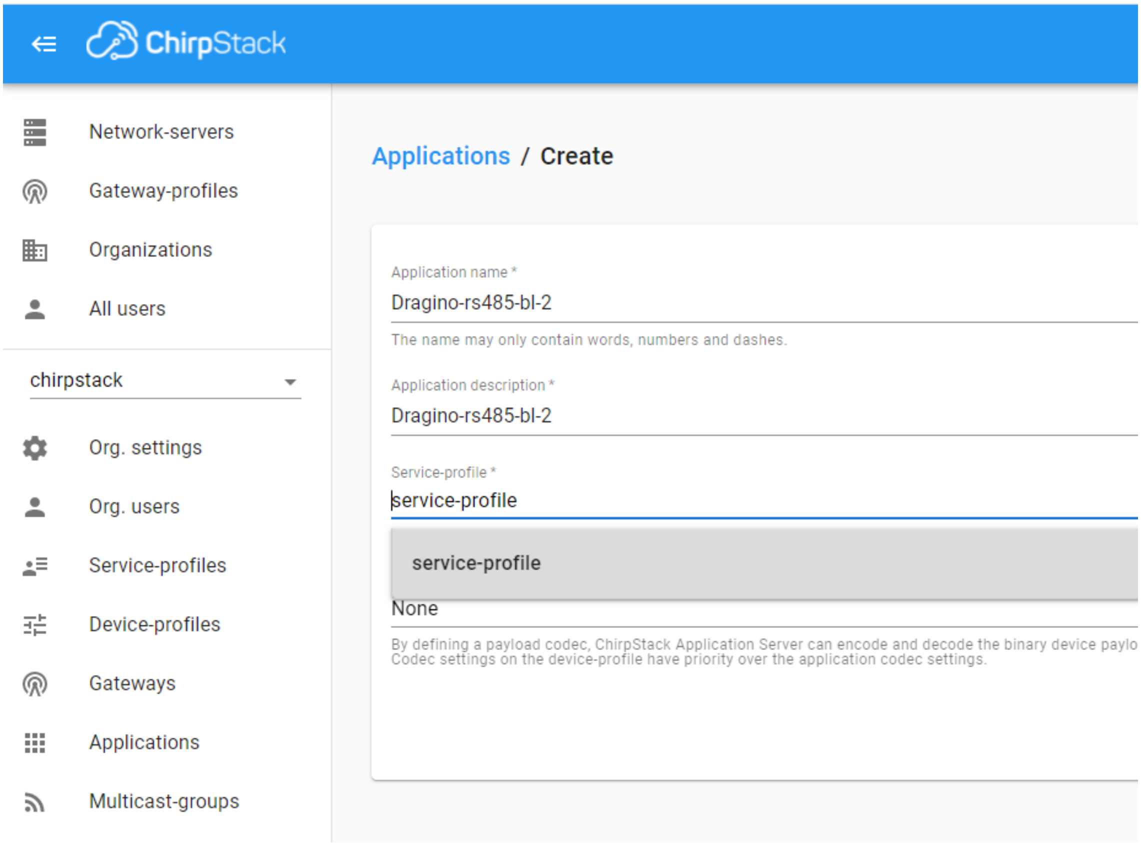
Let’s enter on the Chirpstack server:

http://192.168.1.105:8080/

user: admin

password: admin

Let’s create a new application



Applications / Create

Application name *

Dragino-rs485-bl-2

The name may only contain words, numbers and dashes.

Application description *

Dragino-rs485-bl-2

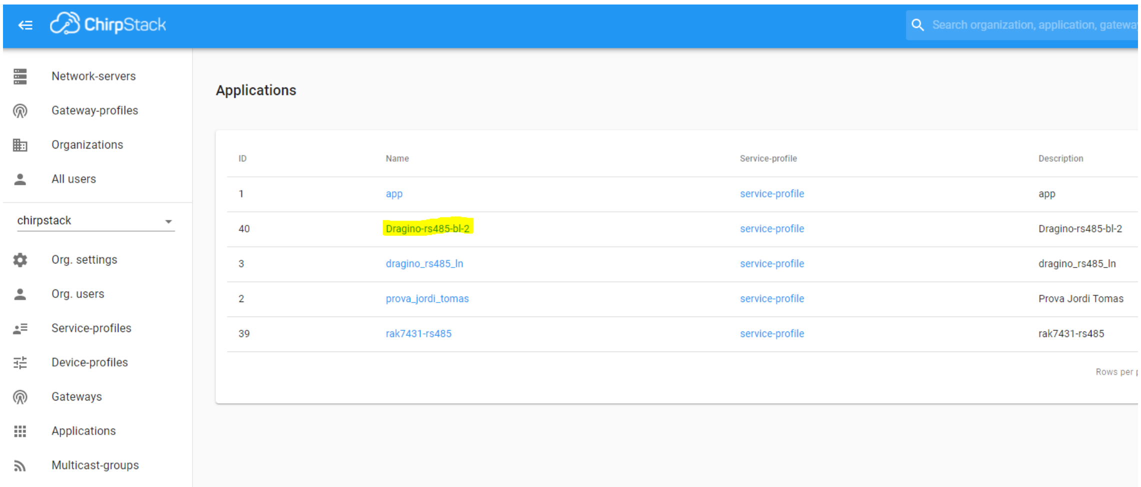
Service-profile *

service-profile

service-profile

None

By defining a payload codec, ChirpStack Application Server can encode and decode the binary device paylo
Codec settings on the device-profile have priority over the application codec settings.

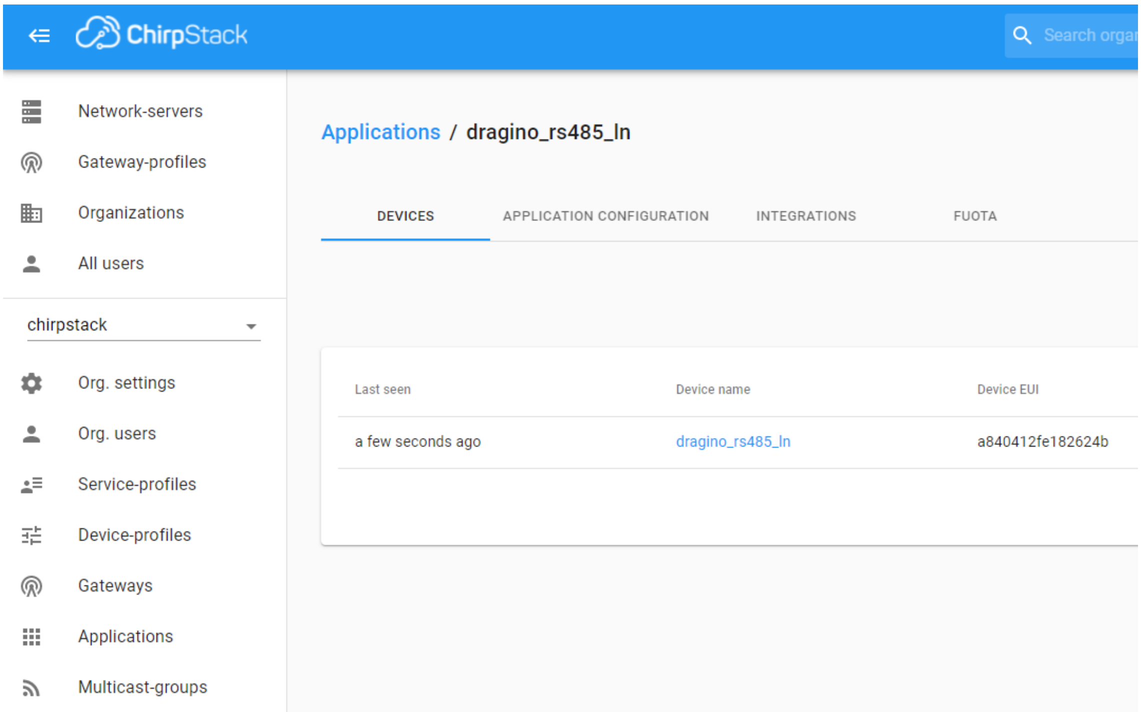


Applications

ID	Name	Service-profile	Description
1	app	service-profile	app
40	Dragino-rs485-bl-2	service-profile	Dragino-rs485-bl-2
3	dragino_rs485_ln	service-profile	dragino_rs485_ln
2	prova_jordi_tomas	service-profile	Prova Jordi Tomas
39	rak7431-rs485	service-profile	rak7431-rs485

Rows per p

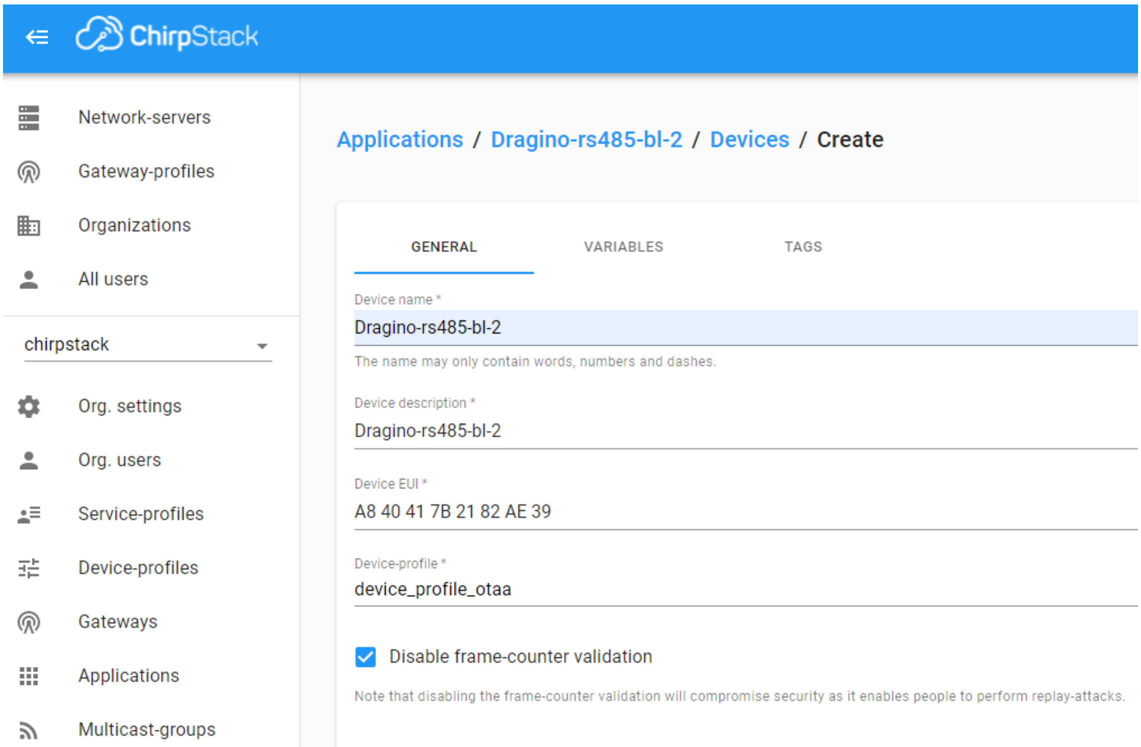
Let’s add a new device.



Applications / dragino_rs485_ln

Last seen	Device name	Device EUI
a few seconds ago	dragino_rs485_ln	a840412fe182624b

We use same device as before with same DEVEUI than in The Things stack



Applications / Dragino-rs485-bl-2 / Devices / Create

Device name *
Dragino-rs485-bl-2
The name may only contain words, numbers and dashes.

Device description *
Dragino-rs485-bl-2

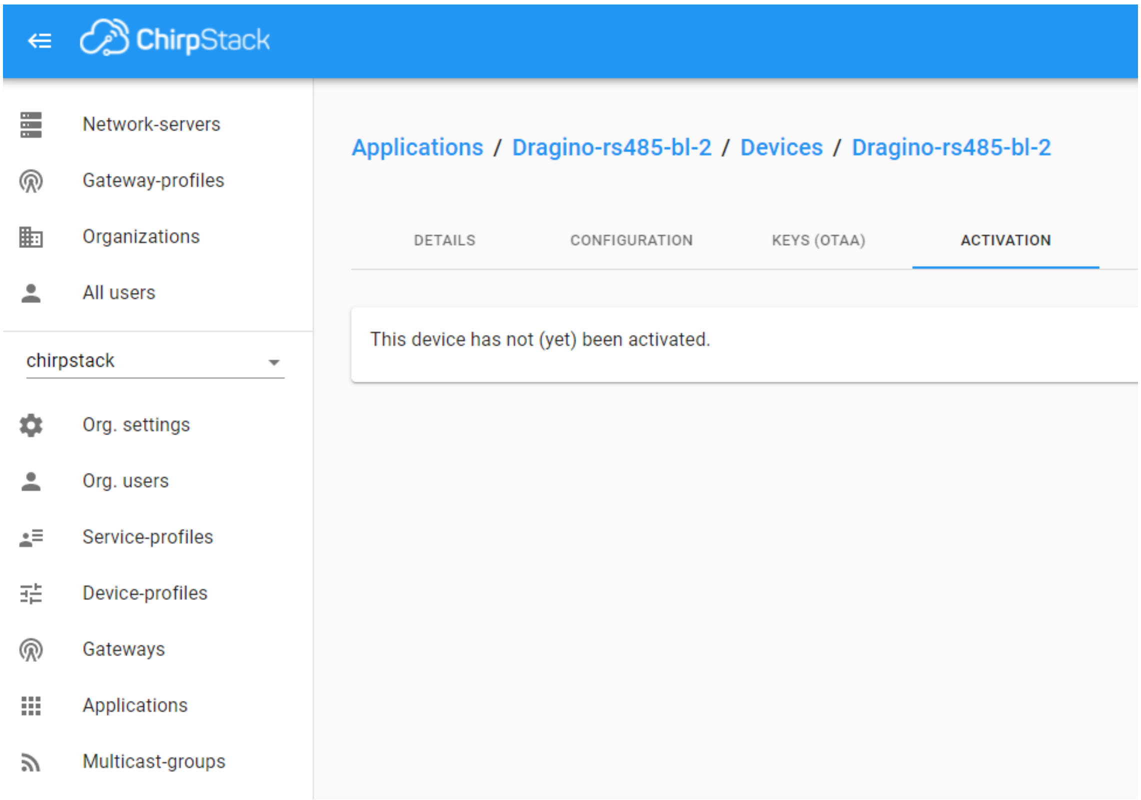
Device EUI *
A8 40 41 7B 21 82 AE 39

Device-profile *
device_profile_otaa

☑ Disable frame-counter validation

Note that disabling the frame-counter validation will compromise security as it enables people to perform replay-attacks.

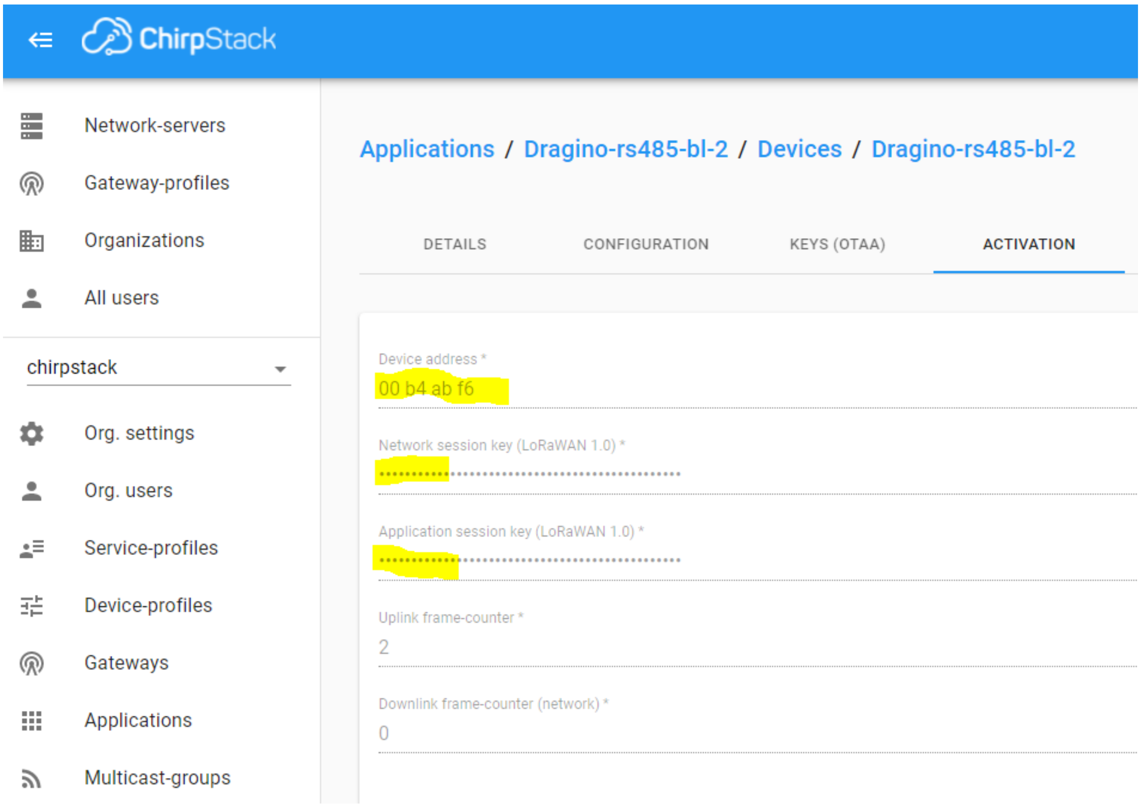
And we have to enter the same AppKey than in The things stack



Applications / Dragino-rs485-bl-2 / Devices / Dragino-rs485-bl-2

This device has not (yet) been activated.

Then we see that the device has been activated, with a different Device address than in the Things Stack



Applications / Dragino-rs485-bl-2 / Devices / Dragino-rs485-bl-2

Device address *
00 b4 ab f6

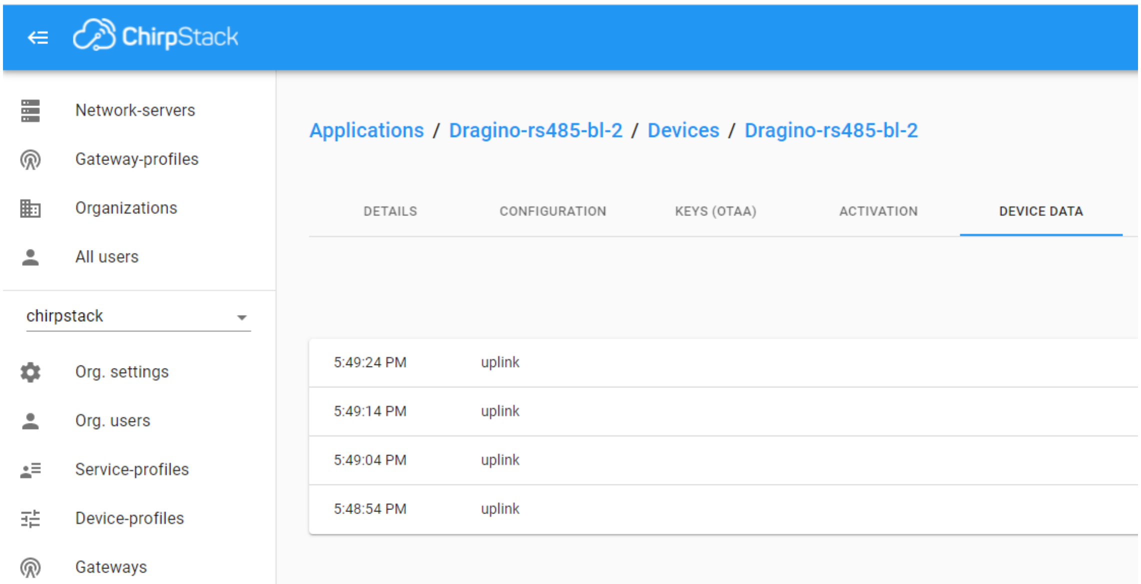
Network session key (LoRaWAN 1.0) *
••

Application session key (LoRaWAN 1.0) *
••

Uplink frame-counter *
2

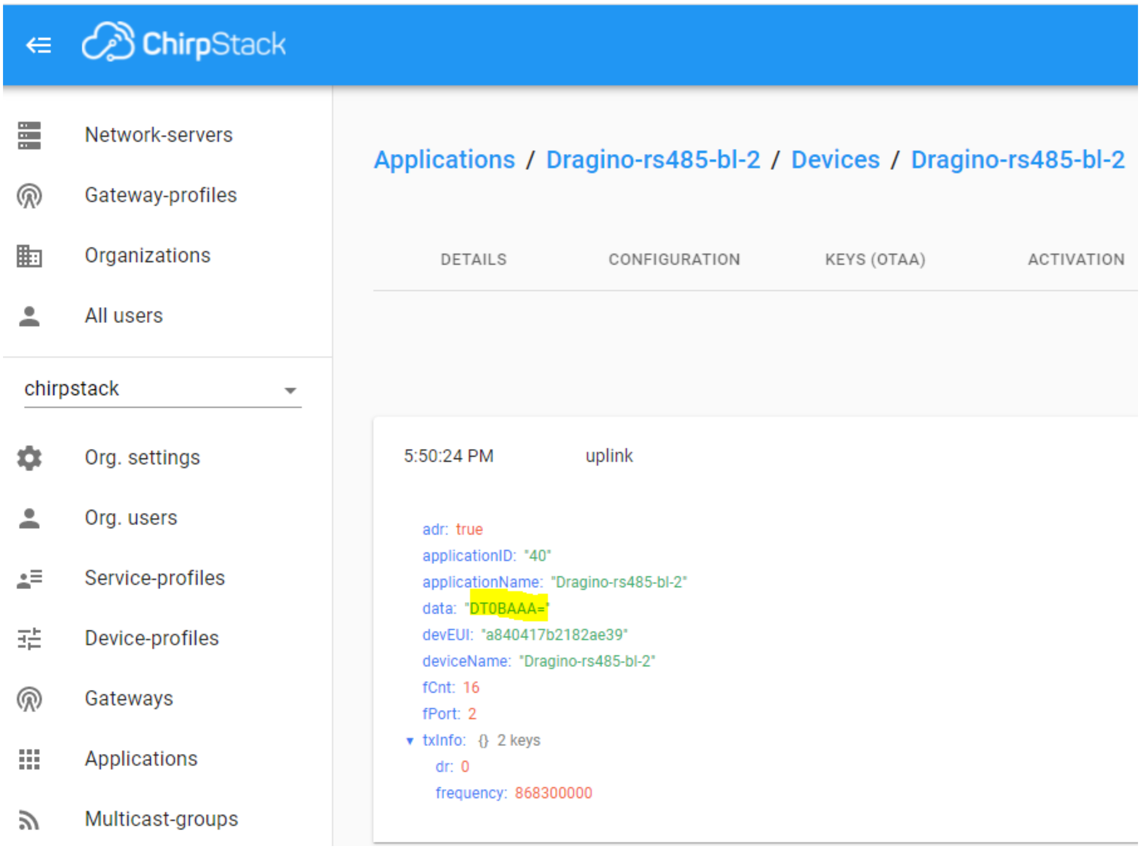
Downlink frame-counter (network) *
0

And starts transmitting



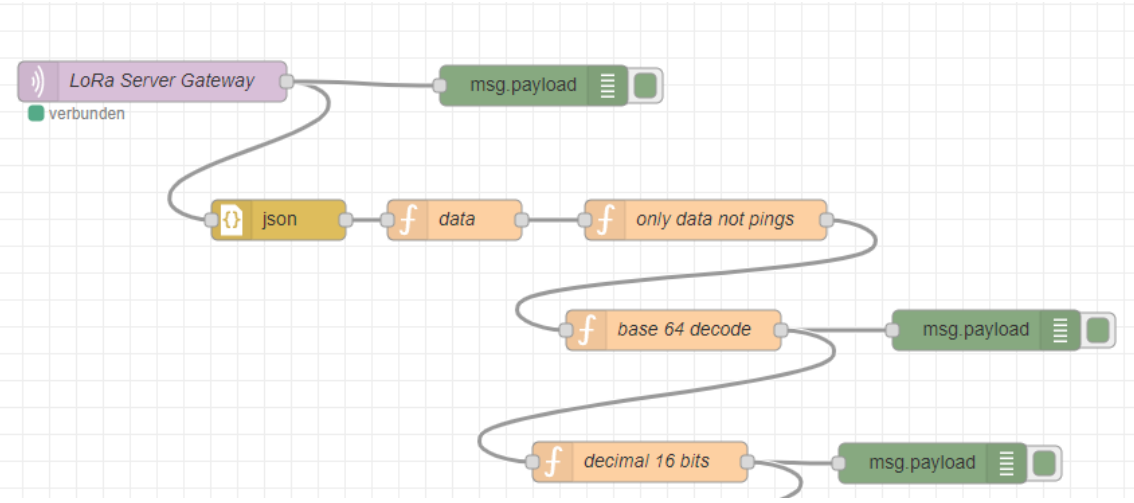
```
5:49:24 PM    uplink
5:49:14 PM    uplink
5:49:04 PM    uplink
5:48:54 PM    uplink
```

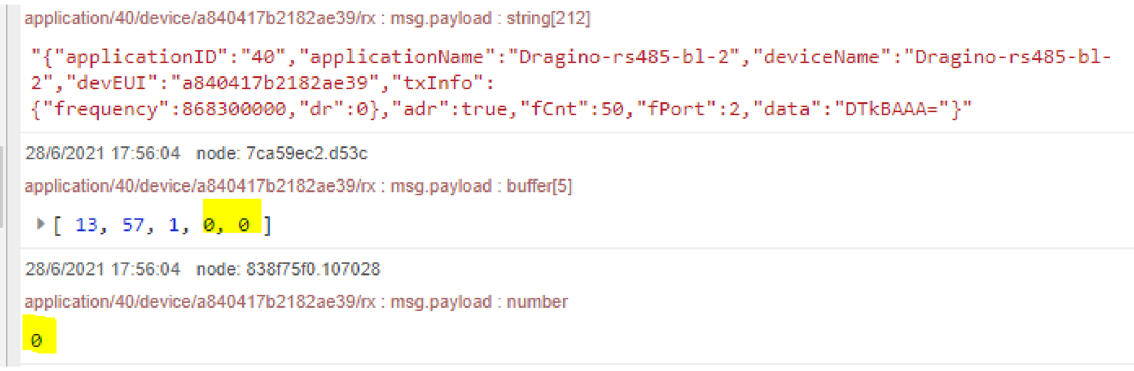
And the data is base 64 encoded



```
5:50:24 PM    uplink

  adr: true
  applicationID: "40"
  applicationName: "Dragino-rs485-bl-2"
  data: "DT0BAAA="
  devEUI: "a840417b2182ae39"
  deviceName: "Dragino-rs485-bl-2"
  fCnt: 16
  fPort: 2
▼ txInfo: {} 2 keys
    dr: 0
    frequency: 868300000
```

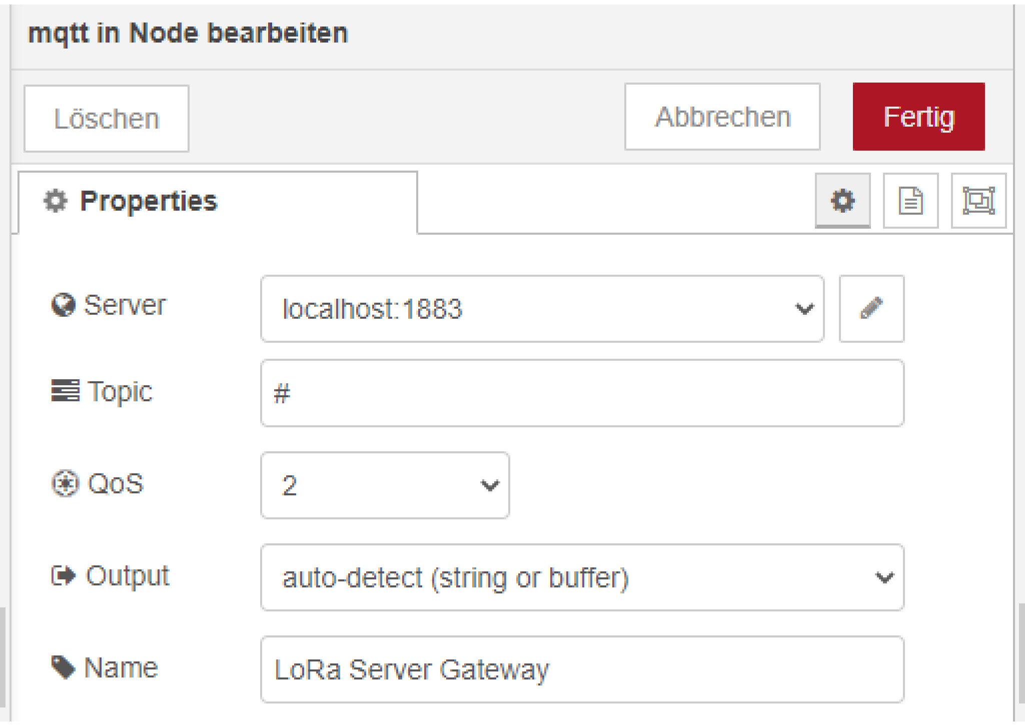
Let’s use Node-RED on same machine





```
application/40/device/a840417b2182ae39/rx : msg.payload : string[212]
"{"applicationID":"40","applicationName":"Dragino-rs485-bl-2","deviceName":"Dragino-rs485-bl-
2","devEUI":"a840417b2182ae39","txInfo":
{"frequency":868300000,"dr":0},"adr":true,"fCnt":50,"fPort":2,"data":"DTkBAAA="}"

28/6/2021 17:56:04   node: 7ca59ec2.d53c
application/40/device/a840417b2182ae39/rx : msg.payload : buffer[5]
▸[ 13, 57, 1, 0, 0 ]

28/6/2021 17:56:04   node: 838f75f0.107028
application/40/device/a840417b2182ae39/rx : msg.payload : number
0
```

So we receive the right payload



mqtt in Node bearbeiten

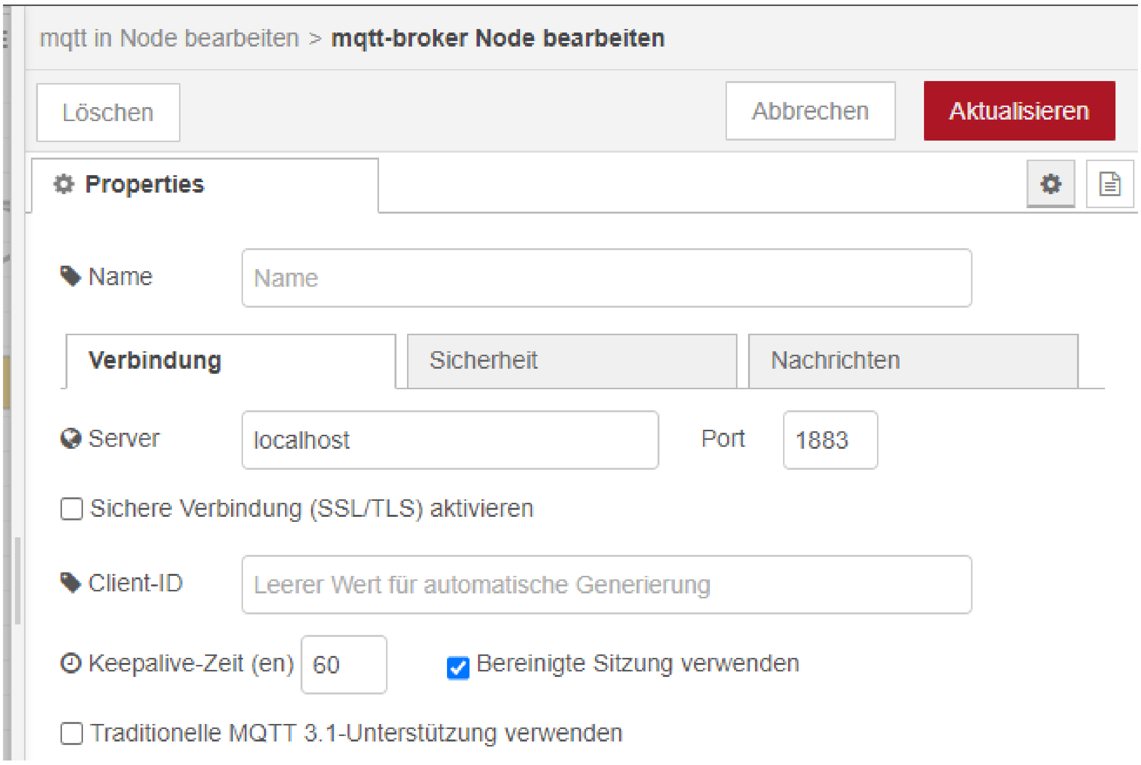
Server: localhost:1883

Topic: #

QoS: 2

Output: auto-detect (string or buffer)

Name: LoRa Server Gateway



Name: Name

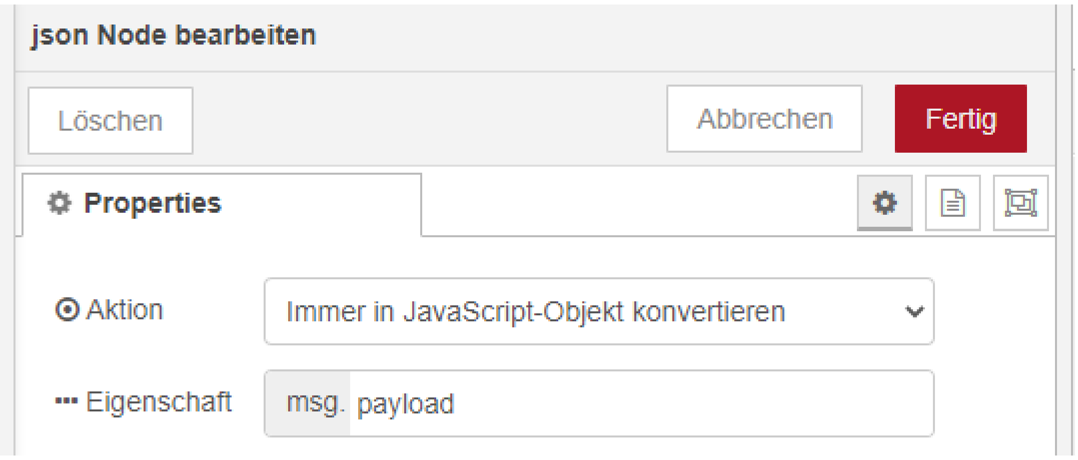
Server: localhost   Port: 1883

☐ Sichere Verbindung (SSL/TLS) aktivieren

Client-ID: Leerer Wert für automatische Generierung

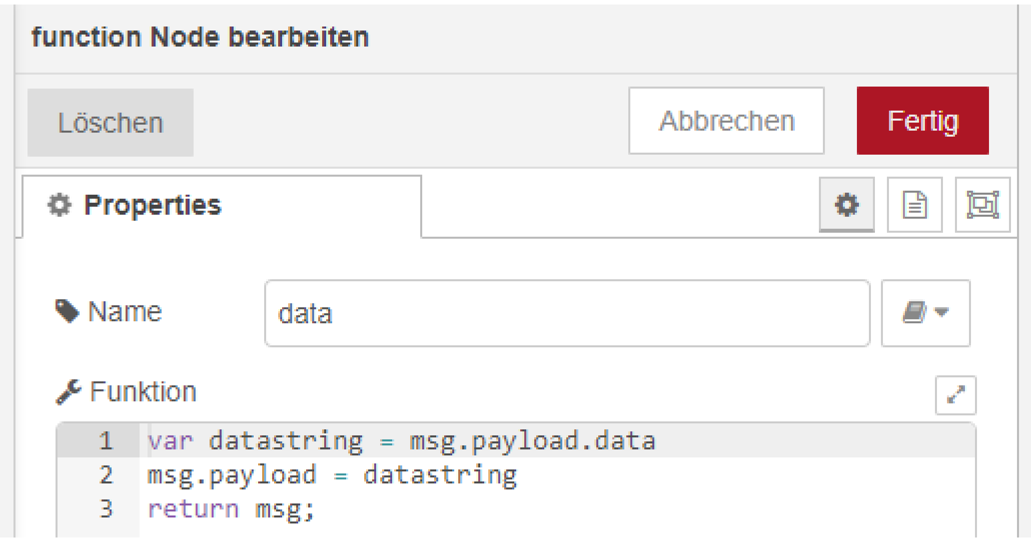
Keepalive-Zeit (en): 60   ☑ Bereinigte Sitzung verwenden

☐ Traditionelle MQTT 3.1-Unterstützung verwenden



Aktion: Immer in JavaScript-Objekt konvertieren

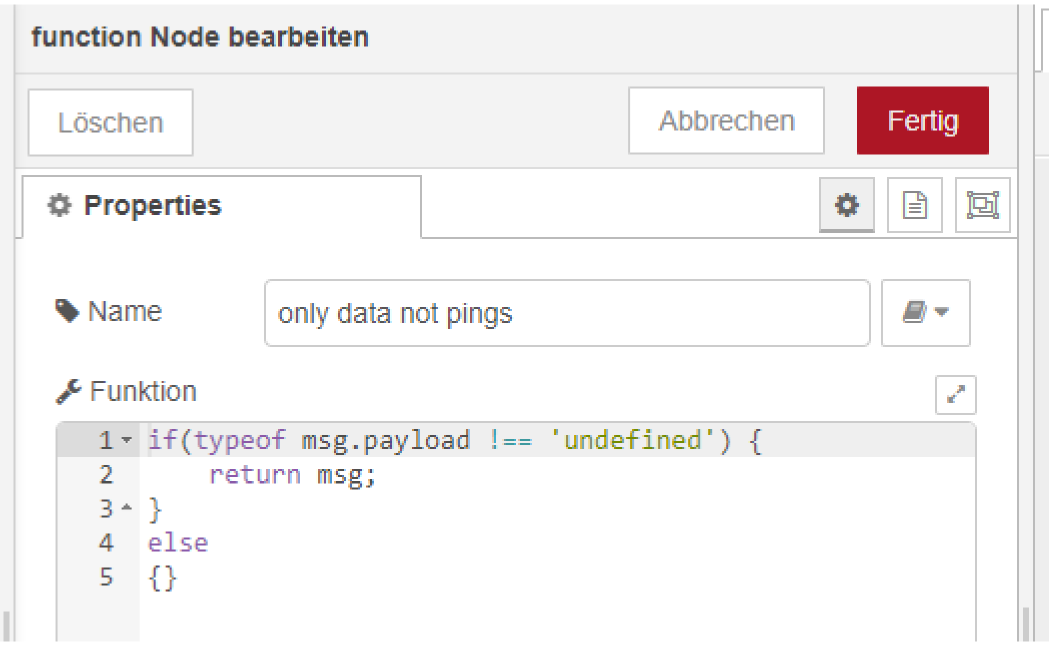
Eigenschaft: msg. payload



Name: data

Funktion

```
var datastring = msg.payload.data
msg.payload = datastring
return msg;
```

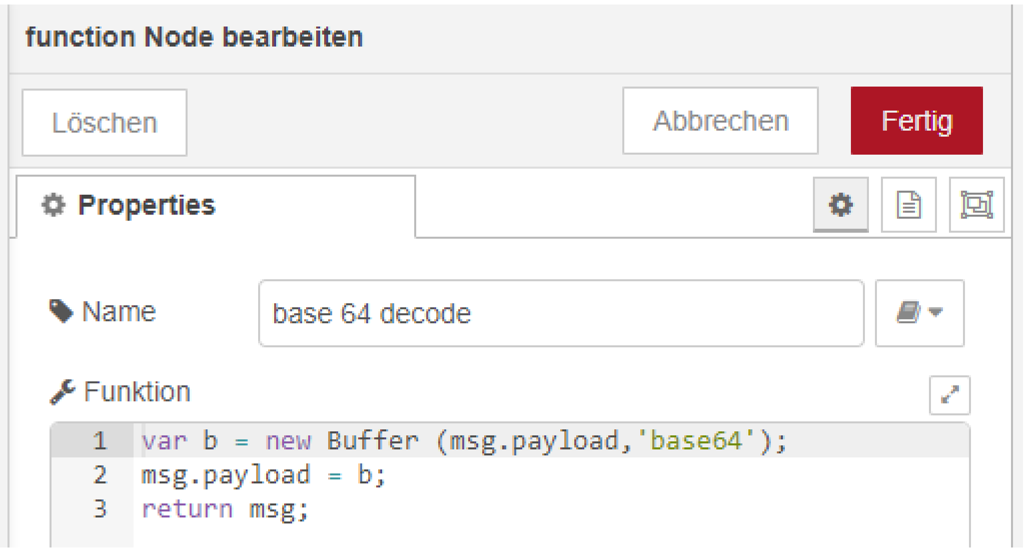


function Node bearbeiten

Name: only data not pings

Funktion

```
if(typeof msg.payload !== 'undefined') {
    return msg;
}
else
{}
```

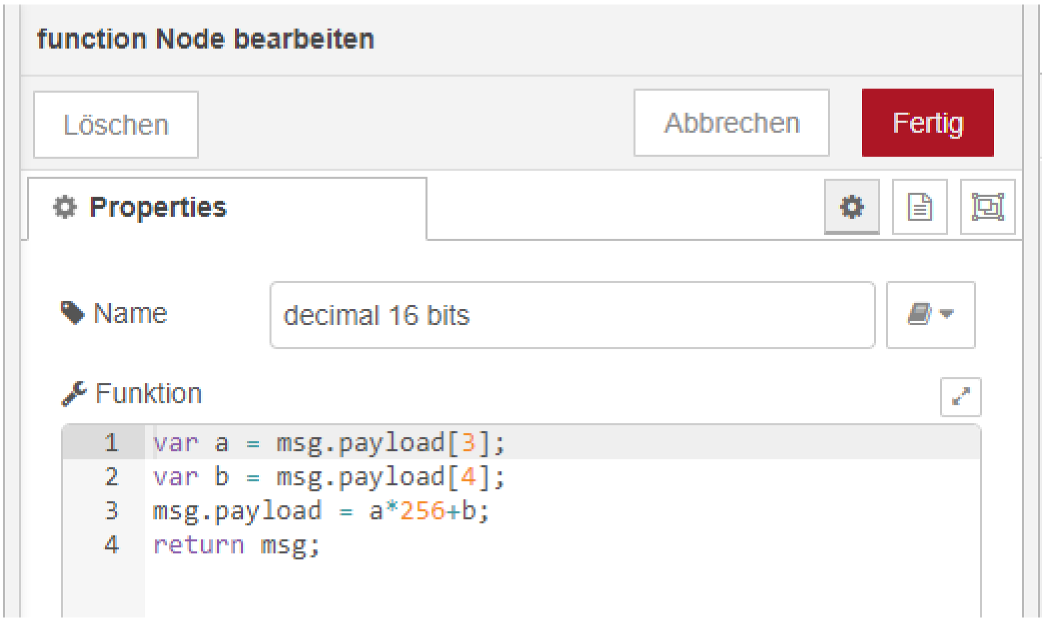


function Node bearbeiten

Name: base 64 decode

Funktion

```
var b = new Buffer (msg.payload,'base64');
msg.payload = b;
return msg;
```

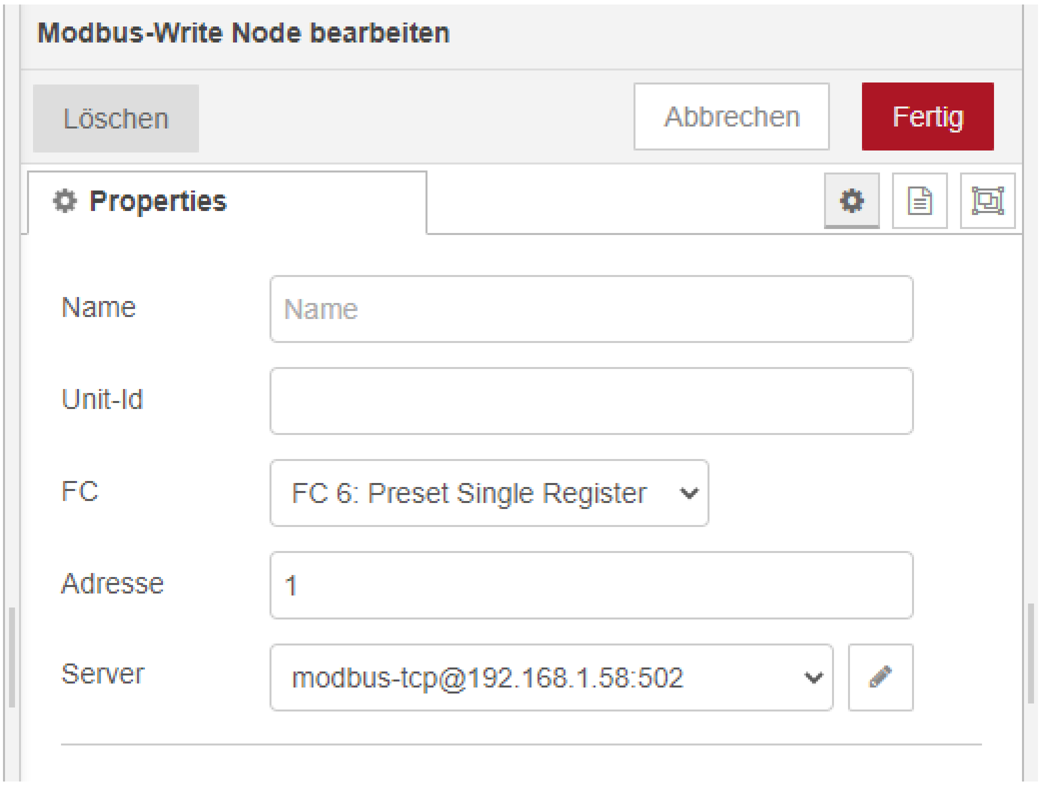


function Node bearbeiten

Name: decimal 16 bits

Funktion

```
var a = msg.payload[3];
var b = msg.payload[4];
msg.payload = a*256+b;
return msg;
```



Modbus-Write Node bearbeiten

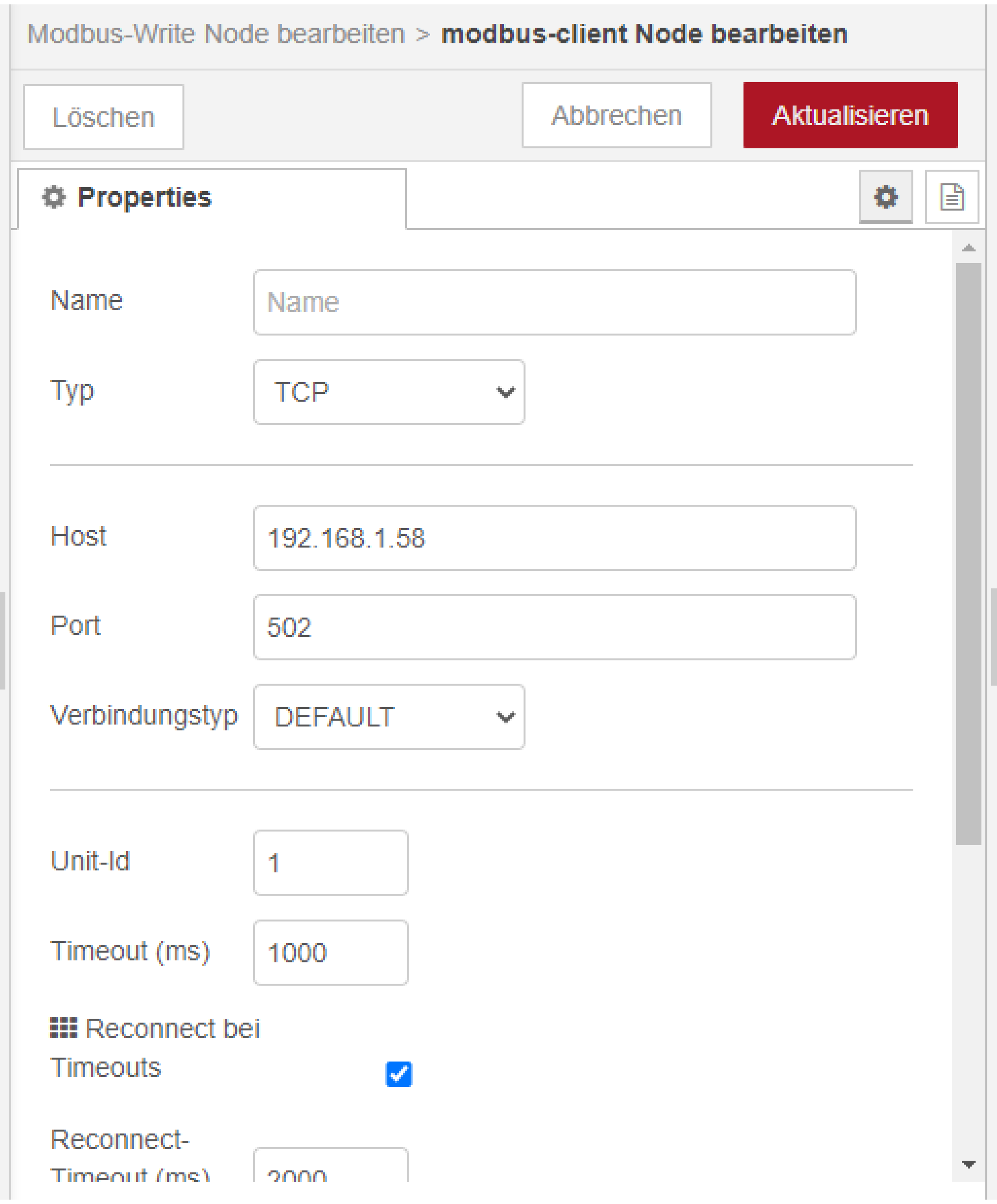
Name: Name

Unit-Id:

FC: FC 6: Preset Single Register

Adresse: 1

Server: modbus-tcp@192.168.1.58:502



Modbus-Write Node bearbeiten > **modbus-client Node bearbeiten**

Name: Name

Typ: TCP

Host: 192.168.1.58

Port: 502

Verbindungstyp: DEFAULT

Unit-Id: 1

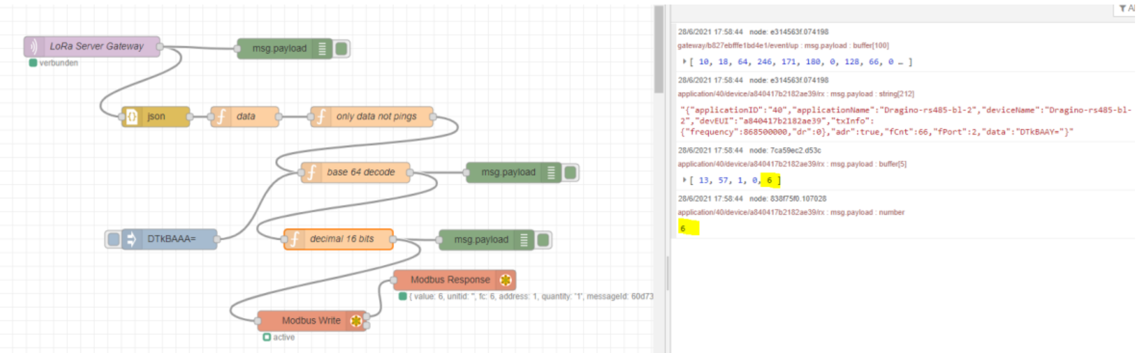
Timeout (ms): 1000

Reconnect bei Timeouts ☑

Reconnect-Timeout (ms): 2000

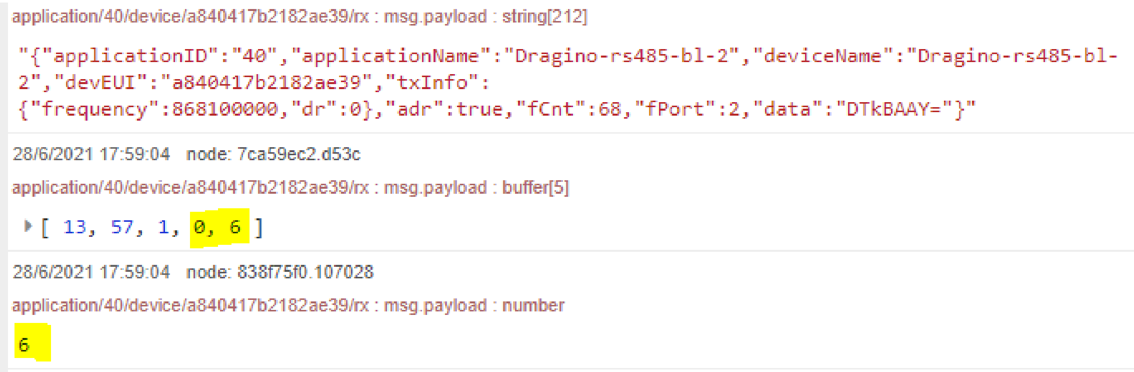
Now let’s activate input 1 and 2 on the sender PLC

Which is 6 coded in binary



{ value: 6, unitid: '', fc: 6, address: 1, quantity: '1', messageId: 60d73

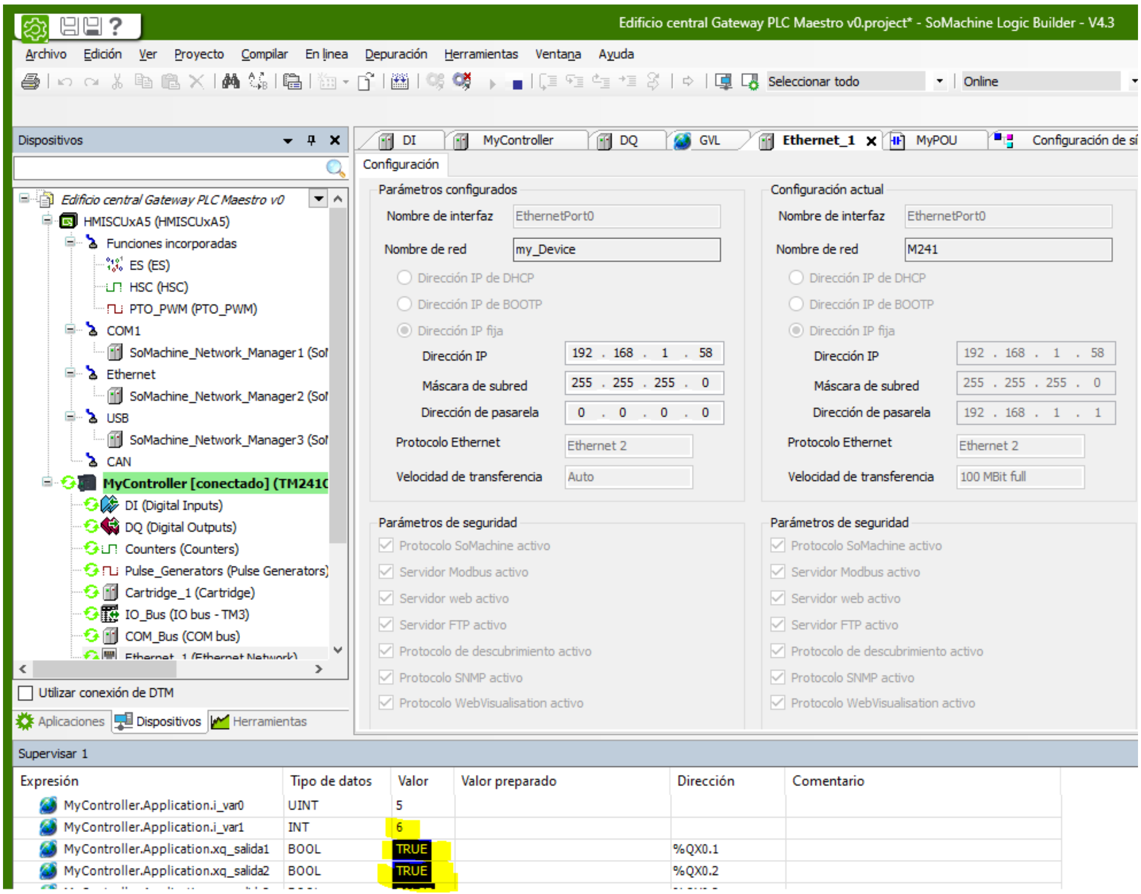
```
28/6/2021 17:58:44  node: e314563f.074198
gateway/b827ebfffe1bd4e1/event/up : msg.payload : buffer[100]
▸[ 10, 18, 64, 246, 171, 180, 0, 128, 66, 0 … ]
28/6/2021 17:58:44  node: e314563f.074198
application/40/device/a840417b2182ae39/rx : msg.payload : string[212]
"{"applicationID":"40","applicationName":"Dragino-rs485-bl-2","deviceName":"Dragino-rs485-bl-
2","devEUI":"a840417b2182ae39","txInfo":
{"frequency":868500000,"dr":0},"adr":true,"fCnt":66,"fPort":2,"data":"DTkBAAY="}"
28/6/2021 17:58:44  node: 7ca59ec2.d53c
application/40/device/a840417b2182ae39/rx : msg.payload : buffer[5]
▸[ 13, 57, 1, 0, 6 ]
28/6/2021 17:58:44  node: 838f75f0.107028
application/40/device/a840417b2182ae39/rx : msg.payload : number
6
```



```
application/40/device/a840417b2182ae39/rx : msg.payload : string[212]
"{"applicationID":"40","applicationName":"Dragino-rs485-bl-2","deviceName":"Dragino-rs485-bl-2","devEUI":"a840417b2182ae39","txInfo":{"frequency":868100000,"dr":0},"adr":true,"fCnt":68,"fPort":2,"data":"DTkBAAY="}"

28/6/2021 17:59:04   node: 7ca59ec2.d53c
application/40/device/a840417b2182ae39/rx : msg.payload : buffer[5]
▸[ 13, 57, 1, 0, 6 ]

28/6/2021 17:59:04   node: 838f75f0.107028
application/40/device/a840417b2182ae39/rx : msg.payload : number
6
```

And on the receiver PLC:

We have the number 6 on the register %MW1 and outputs 1 and 2 on ON state



Parámetros configurados

Nombre de interfaz: EthernetPort0

Nombre de red: my_Device

○ Dirección IP de DHCP
○ Dirección IP de BOOTP
● Dirección IP fija

Dirección IP: 192 . 168 . 1 . 58

Máscara de subred: 255 . 255 . 255 . 0

Dirección de pasarela: 0 . 0 . 0 . 0

Protocolo Ethernet: Ethernet 2

Velocidad de transferencia: Auto

Configuración actual

Nombre de interfaz: EthernetPort0

Nombre de red: M241

○ Dirección IP de DHCP
○ Dirección IP de BOOTP
● Dirección IP fija

Dirección IP: 192 . 168 . 1 . 58

Máscara de subred: 255 . 255 . 255 . 0

Dirección de pasarela: 192 . 168 . 1 . 1

Protocolo Ethernet: Ethernet 2

Velocidad de transferencia: 100 MBit full

Parámetros de seguridad

☑ Protocolo SoMachine activo
☑ Servidor Modbus activo
☑ Servidor web activo
☑ Servidor FTP activo
☑ Protocolo de descubrimiento activo
☑ Protocolo SNMP activo
☑ Protocolo WebVisualisation activo

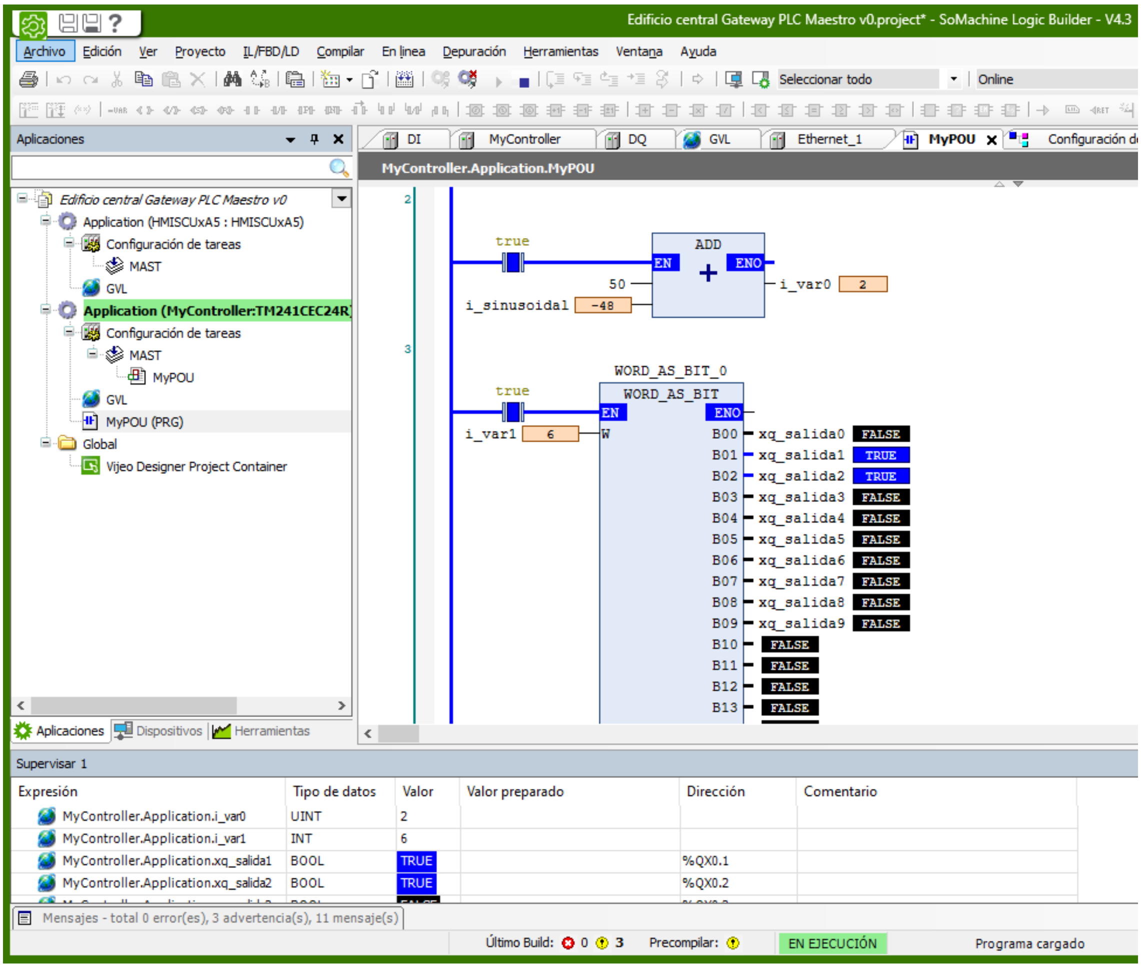
Parámetros de seguridad

☑ Protocolo SoMachine activo
☑ Servidor Modbus activo
☑ Servidor web activo
☑ Servidor FTP activo
☑ Protocolo de descubrimiento activo
☑ Protocolo SNMP activo
☑ Protocolo WebVisualisation activo

□ Utilizar conexión de DTM

Supervisar 1

Expresión	Tipo de datos	Valor	Valor preparado	Dirección	Comentario
MyController.Application.i_var0	UINT	5			
MyController.Application.i_var1	INT	6			
MyController.Application.xq_salida1	BOOL	TRUE		%QX0.1	
MyController.Application.xq_salida2	BOOL	TRUE		%QX0.2	



Expresión	Tipo de datos	Valor	Valor preparado	Dirección	Comentario
MyController.Application.i_var0	UINT	2			
MyController.Application.i_var1	INT	6			
MyController.Application.xq_salida1	BOOL	TRUE		%QX0.1	
MyController.Application.xq_salida2	BOOL	TRUE		%QX0.2	

Mensajes - total 0 error(es), 3 advertencia(s), 11 mensaje(s)

Último Build: 0 3 Precompilar: EN EJECUCIÓN Programa cargado

And that’s all

You can find the Node-RED code here

https://github.com/xavierflorensa/Schneider-M221-as-LoRaWAN-node