Frame Data Parser for WISE LoRa modules (JavaScript version)

V1.3.7

CONTENTS

1. Introduction 6

1.1. About This Manual 6

1.2. Requirements 6

1.3. Note 6

2. Installations 8

2.1. Installation Requirement 8

2.2. Operation Steps 8

3. Data Structure Definition 16

3.1. Definitions 16

3.1.1 Temperature 17

3.1.2 Accelerometer 18

3.1.3 Device 20

3.1.4 Digital Input 20

3.1.5 Digital Output 22

3.1.6 Analog Input 23

3.1.7 RS-485 Coil Data 25

3.1.8 RS-485 Register Data 26

3.1.9 FFT (Fast Fourier Transform) 27

3.1.9.1 FFT in CSV format 27

3.1.9.2 FFT in JSON format 27

3.1.9.3 FFT Data Re-transmission 28

3.1.10 Axis Data 28

3.2. Sample Output 29

4. Appendix 38

Change History

Version	Date	Description
1.0	2020/01/02	Initial version.
1.1	2020/01/30	In Chapter 3, add parsing of DI, DO, AI, RS-485 COM data.
1.2	2020/02/17	In Chapter 3.1.4, add description for DI field: Value and Status.
1.3	2020/02/25	Add FFT data in Chapter 2 and Chapter 3.1.9.
1.3.1	2020/03/20	In Chapter 3.1.9, modify description for FFT Data.
1.3.2	2020/03/27	In Chapter 3, change FFT Data format
1.3.3	2020/05/11	Add get FFT data re-transmission information(optional) in Chapter 2.2 and Chapter 3.1.9.3
1.3.4	2020/05/14	In Chapter 3.1.2, add column LogIndex
1.3.5	2020/07/08	In Chapter 3.1.2, edit field name Displacement
1.3.6	2020/09/03	In Chapter 3.1.2, add column Time
1.3.7	2020/09/30	In Chapter 3, change DO output format

Chapter 1

1. Introduction

1.1. About This Manual

This document describes the Frame Data Parser for Advantech WISE LoRa modules (ex: WISE-2410, WISE-4610 series). User could use this sample program in Node-Red to parse frame data of WISE LoRa modules received in LoRa Gateway. WISE-2410 is a vibration sensor with LoRa/LoRaWAN wireless technology provided by Advantech. In this document, we use Advantech WISE-6610 as the gateway to receive frame data sent from WISE-2410.

For WISE-2410 vibration sensor, amount of FFT (Fast Fourier Transform) data for each axis is 800. Each FFT data is 2 bytes. If user configures WISE-2410 to send 3 axis (X, Y, Z) FFT data, there will be 2400 data which is 4800 bytes.

1.2. Requirements

- Web Browser(Google Chrome is recommended)
- WISE-2410 or WISE-4610
- WISE-6610 LoRa/LoRaWan Gateway with Node-Red installed

1.3. Note

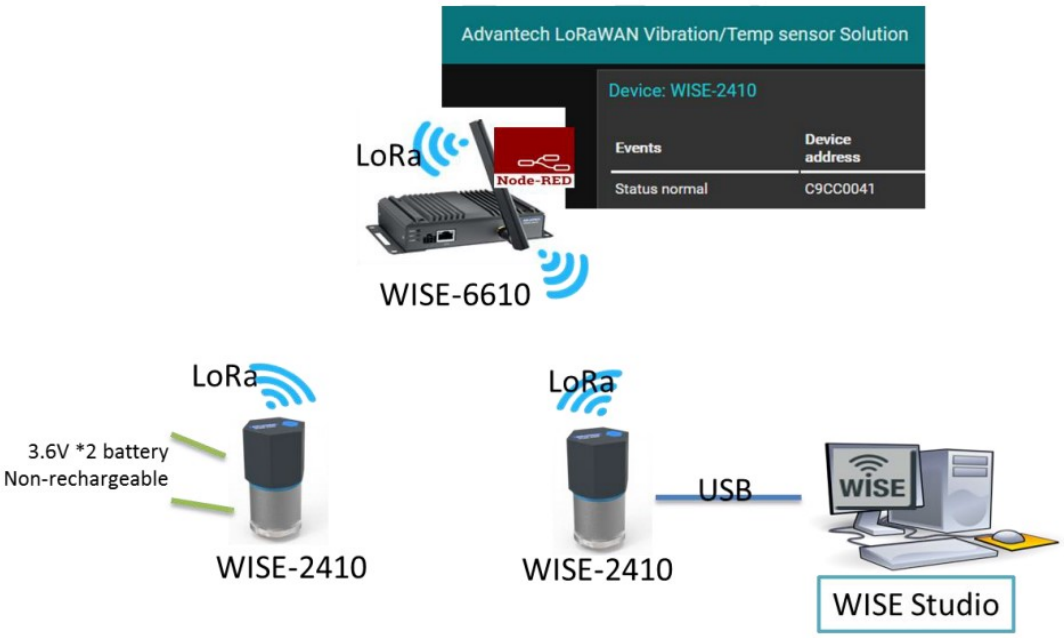
This sample program only works for parsing of single module (LoRa node) data. Parsing of multiple modules data is not supported. In Node-RED, user must subscribe one MQTT topic which contains only one module data.

Chapter 2

2. Installations

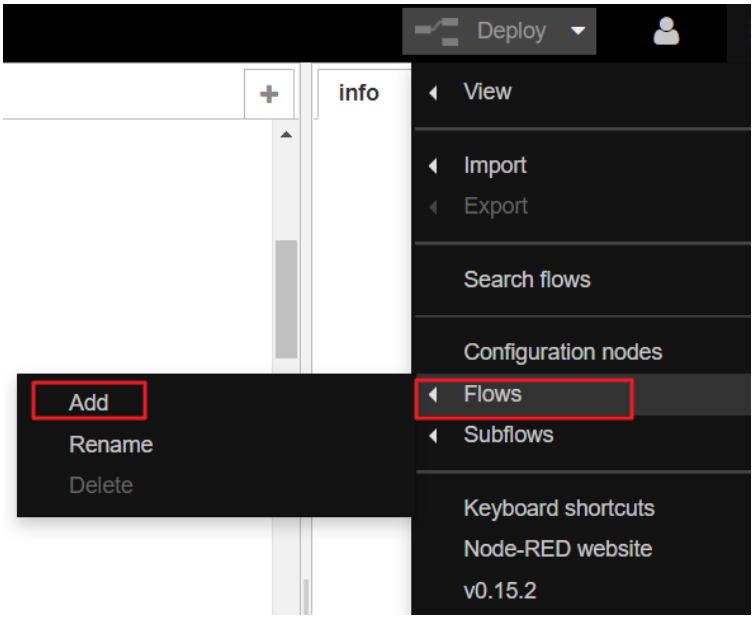
2.1. Installation Requirement

First, please establish connection between WISE-6610 and WISE-2410. Then start the Node-Red service in WISE-6610.

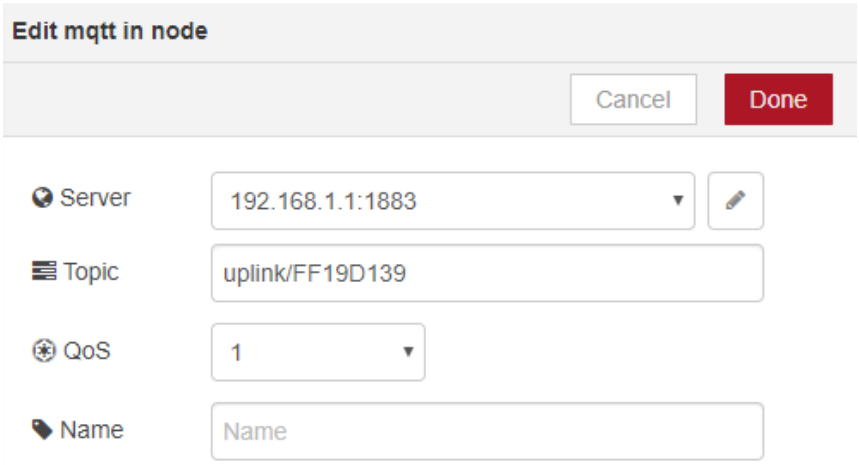


2.2. Operation Steps

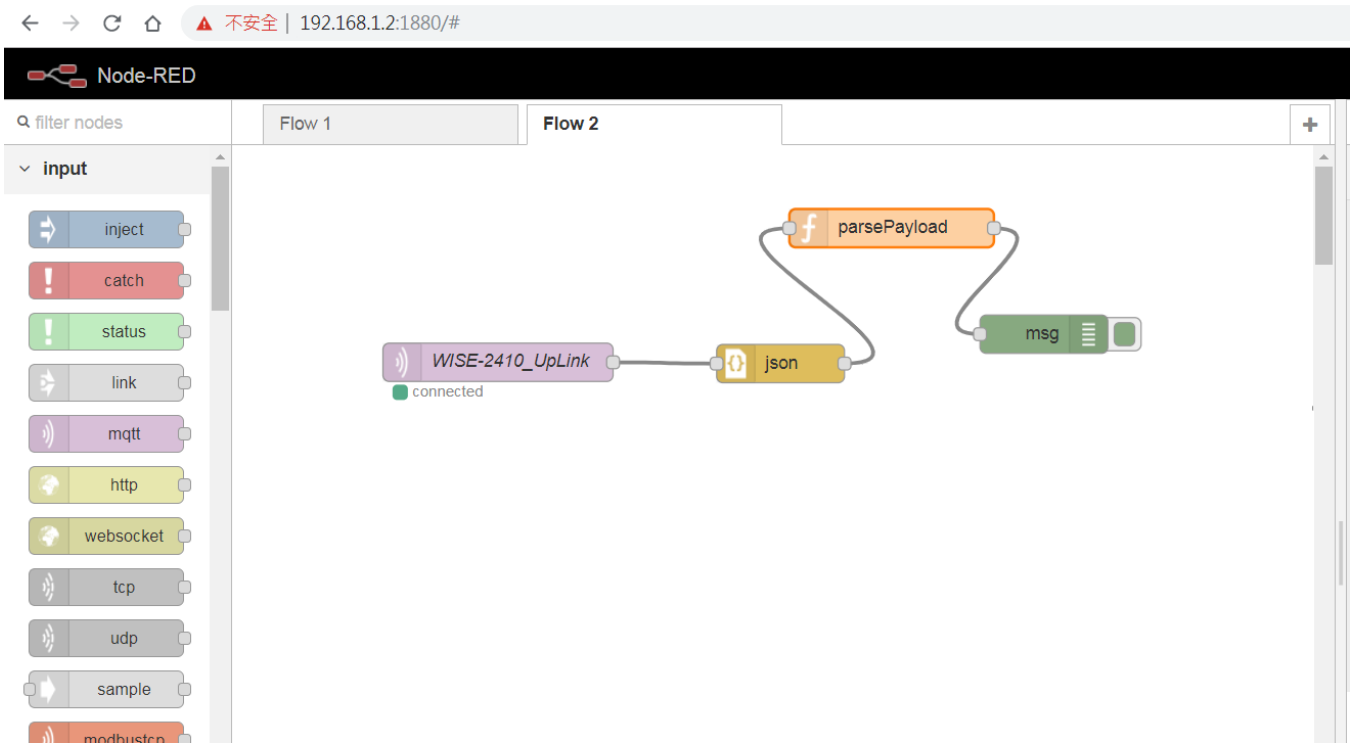
1. Open web browser and navigate to Node-Red pages in WISE-6610.
2. In Node-Red, create a new Flow and add MQTT node, JSON node.



3. Fill the MQTT server information in MQTT node. One parser function only support a single node uplink, Topic is the node MAC.

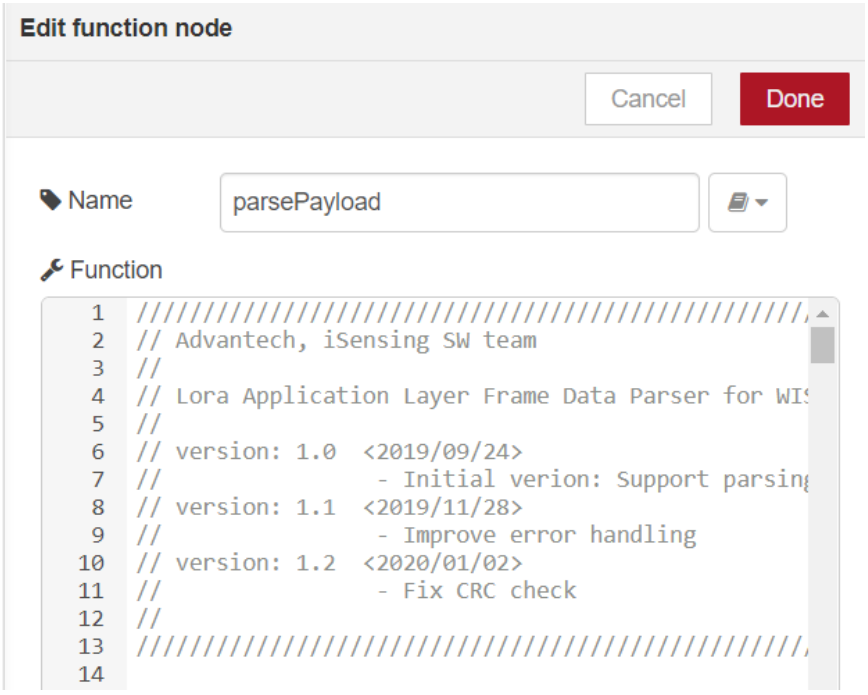


4. Place a Function Node and a Debug node. Then connect all nodes mentioned above. You will see below figure.



5. Use Text Editor to open the JS file provided by this sample and copy all file content into clipboard.

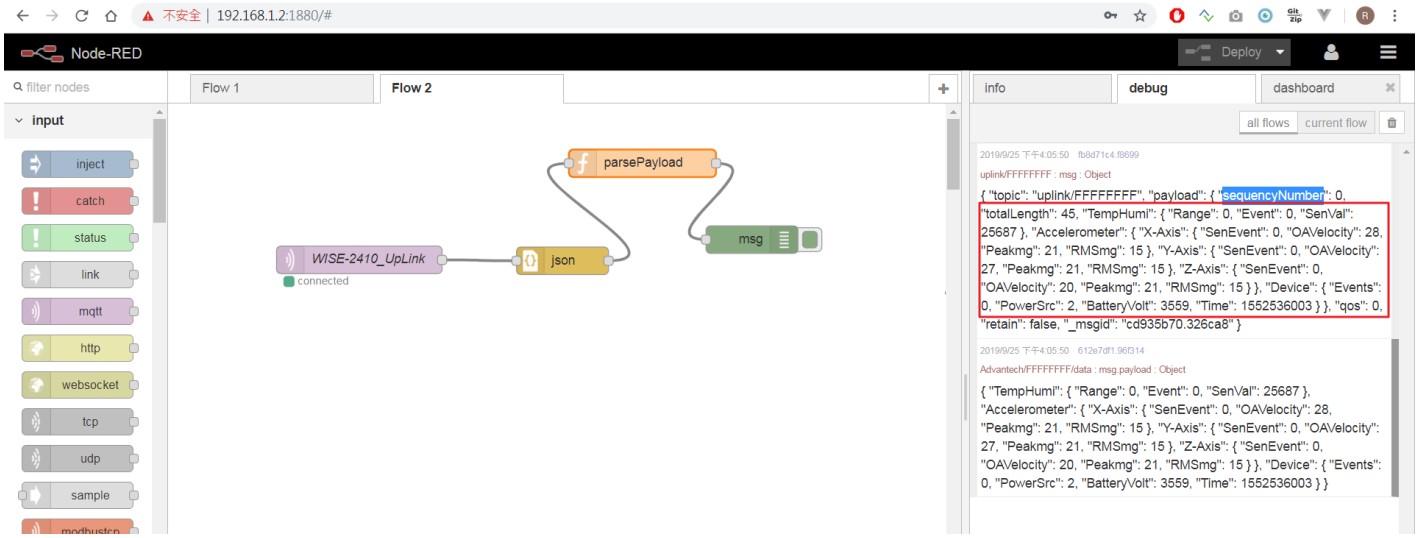
6. Double click Function node (named “parsePayload” here) and paste all content of JS file into Function node.



7. Please make sure the variable bIsRunNodeRed is set to true. This variable enables the Node-Red environment setting for this sample program.

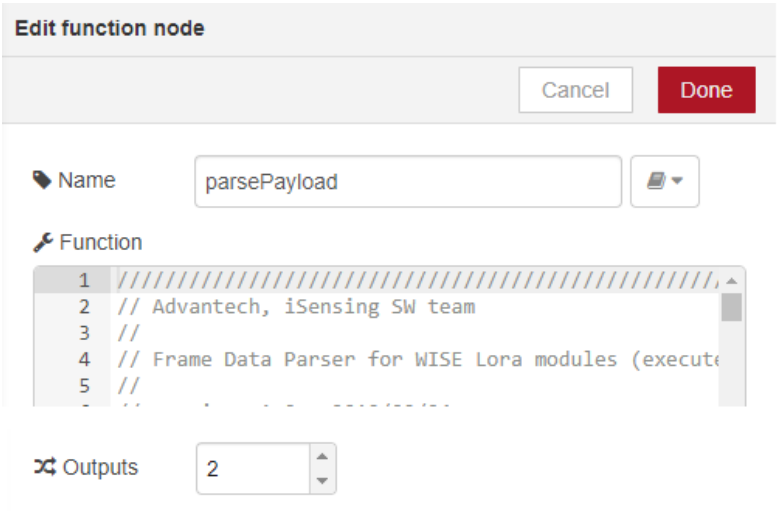
```
14  ////////////////////////////////////////
15  // User defined variables
16  ////////////////////////////////////////
17
18  // If program is run in NodeRed. True: run in NodeRed, False: not run in NodeRed
19  var bIsRunNodeRed = false;
```

8. Click Deploy button in Node-Red. When WISE-2410 sends frame data to WISE-6610, you will see the parsed output of sample program in the Debug Tab on the right side.

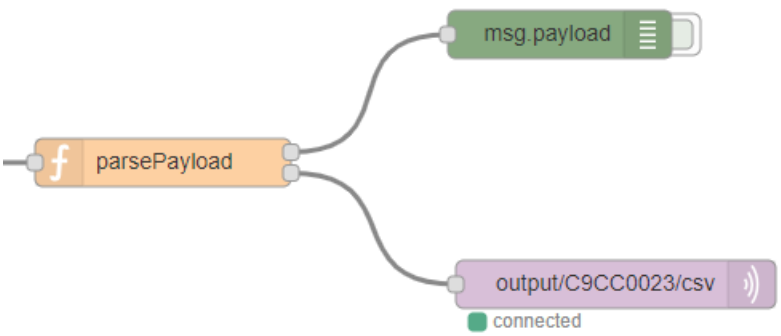


9. (Optional) Get FFT (Fast Fourier Transform) data in CSV format from MQTT subscription.

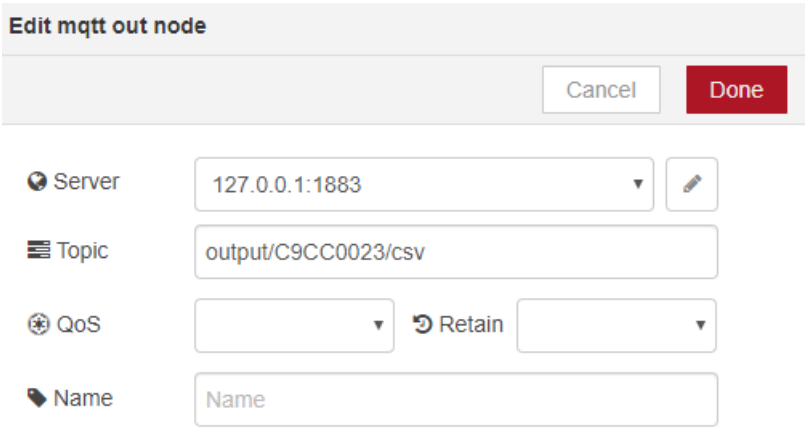
   i. Set outputs of function node to 2.



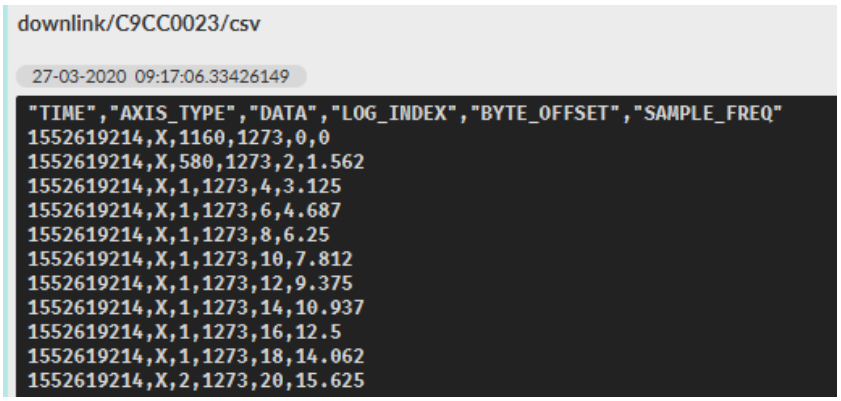
ii. Place a MQTT out node and connect it with “parsePayload“ node.



iii. Fill the MQTT server and Topic information in MQTT out node. User could define MQTT Server and Topic name.

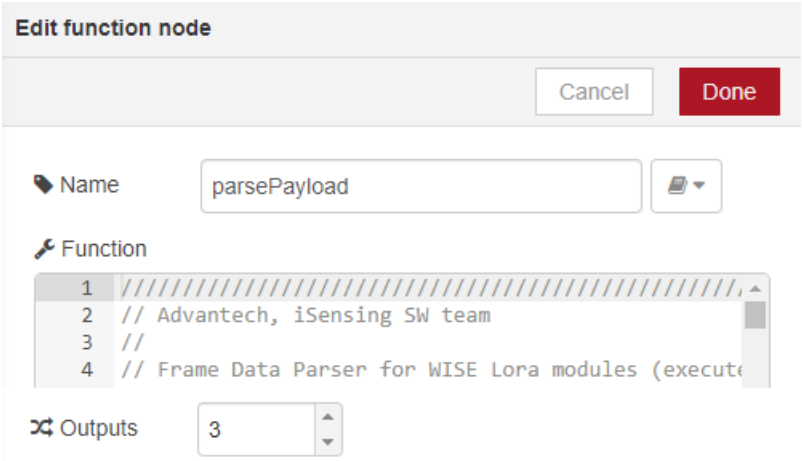


iv. Deploy the Node-Red and use software tool (ex: MQTT.fx) to subscribe this topic. You will get FFT data in CSV format when WISE-2410 sends FFT data to WISE-6610.

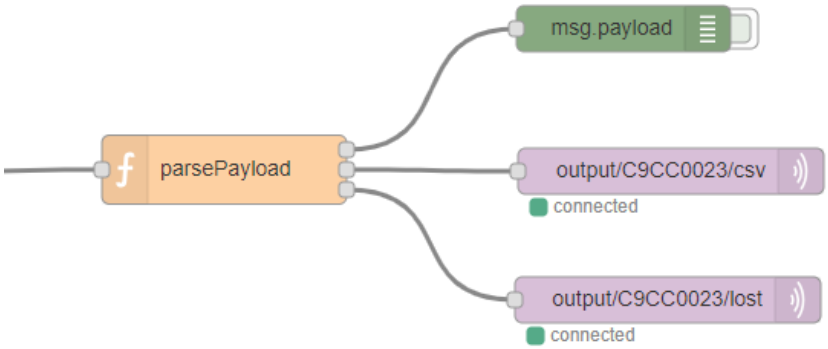


10. (Optional) If packet lost occurs, user could get packet re-transmission information from MQTT subscription.

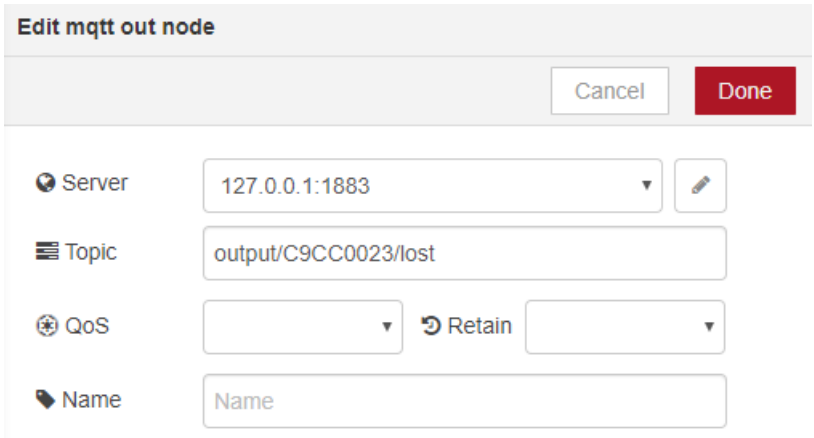
i. Set outputs of function node to 3.



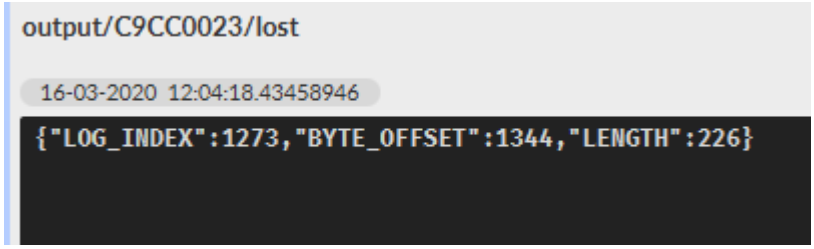
ii. Place a MQTT out node and connect it with “parsePayload“ node.



iii. Fill the MQTT server and Topic information in MQTT out node.



iv. Deploy the Node-Red and use software tool (ex: MQTT.fx) to subscribe this topic. You will get data re-transmission information in JSON format when lost packet. Please refer to Chapter 3.1.9.3 for JSON definition.



Chapter 3

3. Data Structure Definition

3.1. Definitions

This sample program parses received LoRa frame data and translates into JSON/CSV format. Please refer to below table for data definition:

Field Name	Data Type	Description
SequenceNumber	int	Sequence number of frame data
TotalLength	int	Total length of frame data
SourceAddress	string	Source device address
TempHumi	Temperature	Temperature and Humidity. Please refer to Chapter 3.1.1.
Accelerometer	Accelerometer	Accelerometer. Please refer to Chapter 3.1.2.
Device	Device	Device Status. Please refer to Chapter 3.1.3.
DI(channel)	Digital Input	Digital Input. Please refer to Chapter 3.1.4.
DO(channel)	Digital Output	Digital Output. Please refer to Chapter 3.1.5.
AI(channel)	Analog Input	Analog Input. Please refer to Chapter 3.1.6.
RtuCoil(port)-(channel)	Coil Data	RS-485 Coil data. Please refer to Chapter 3.1.7.

RtuRegister(port)-(channel)	Register Data	RS-485 Register data. Please refer to Chapter 3.1.8.
FFT	FFT (Fast Fourier Transform) Data	FFT (Fast Fourier Transform) Data. Please refer to Chapter 3.1.9.

3.1.1 Temperature

Field Name	Data Type	Description
Range	int	0x0 - Temperature (℃) 0x1 - Temperature (℉) 0x2 - Temperature (K) 0x3 - Humidity (%)
Event	int	2 octets

Bit	Range: 0x0 – 0x3 Temp./Humidity
Bit 0	High alarm status Read 1: high alarm occurred. 0: not occurred Write 0: clear the high alarm status
Bit 1	Low Alarm Status Read 1: low alarm occurred. 0: not occurred

			Write 0: clear the low alarm status
		Bit 2	Clear Maximum Sensor Value 1: Clear the maximum Sensor value
		Bit 3	Clear Minimum Sensor Value 1: Clear the minimum Sensor value
		Bit 4	Alarm update
		Bit 5 ~ 15	(Reserved)
Status	int	0: Normal 1: Sensor failed	
SenVal	int	Temperature or humidity value (0.001)	

3.1.2 Accelerometer

Field Name	Data Type	Description
LogIndex	int	Index number of logged feature data, raw data or FFT data
Time	int	The time to start measuring vibration
X-Axis, Y-Axis, Z-Axis	Object	X, Y, or Z Axis

<table>
<tr><td>SenEvent</td><td>int</td><td><table>
<tr><td>Bit</td><td>Range: 0x4

Accelerometer</td></tr>
<tr><td>Bit 0</td><td>High alarm status

Read 1: high alarm occurred.
0: not occurred

Write 0: clear the high alarm status</td></tr>
<tr><td>Bit 1</td><td>(Reserved)</td></tr>
<tr><td>Bit 2</td><td>(Reserved)</td></tr>
<tr><td>Bit 3</td><td>(Reserved)</td></tr>
<tr><td>Bit 4</td><td>(Reserved)</td></tr>
<tr><td>Bit 5 ~ 15</td><td>(Reserved)</td></tr>
</table></td></tr>
<tr><td>OAVelocity</td><td>int</td><td>OA Value of Vibration Velocity (0.01 mm/sec) (2 Bytes)</td></tr>
<tr><td>Peakmg</td><td>int</td><td>Peak Value of Acceleration (0.001g or 0.01 m/s^2) (2 Bytes)</td></tr>
<tr><td>RMSmg</td><td>int</td><td>RMS of Acceleration (0.001g or 0.01 m/s^2) (2 Bytes)</td></tr>
<tr><td>Kurtosis</td><td>int</td><td>Kurtosis (0.01) (2 Bytes)</td></tr>
<tr><td>CrestFactor</td><td>int</td><td>Crest factor (0.01) (2 Bytes)</td></tr>
<tr><td>Skewness</td><td>int</td><td>Skewness (0.01) (2 Bytes)</td></tr>
<tr><td>Deviation</td><td>int</td><td>Standard deviation (0.01) (2 Bytes)</td></tr>
</table>

Peak-to-Peak Displacement	int	Peak-to-Peak Displacement, unit: um (2 Bytes)

3.1.3 Device

Field Name	Data Type	Description
Events	int	Bit Order: Description **0**: Battery low **1**: RTC low **2~7**: Reserved
PowerSrc	int	Bit Order: Description **0**: Power line **1**: Battery **2**: Solar panels **3~7**: Reserved
BatteryVolt	int	The battery voltage, unit: mV.
Time	int	Timestamp
GNSS	Object	Latitude Longitude

3.1.4 Digital Input

Field Name	Data Type	Description

<table>
<tr><td>mode</td><td>int</td><td><table>
<tr><td>0</td><td>DI</td></tr>
<tr><td>1</td><td>Counter</td></tr>
<tr><td>2</td><td>LowToHighLatch</td></tr>
<tr><td>3</td><td>HighToLowLatch</td></tr>
<tr><td>4</td><td>Frequency</td></tr>
</table></td></tr>
<tr><td>status</td><td>Object</td><td><table>
<tr><td>Signal Logic Status

1, 0: Input signal is Logic High or Low.</td></tr>
<tr><td>Start Counter

Read 1: counter is counting

0: not counting

Write 1: start counting

0: stop counting</td></tr>
<tr><td>Get/Clear Counter Overflow Status

Read 1: overflow occurred.

0: no overflow

Write 0: clear the overflow status</td></tr>
<tr><td>Get/Clear L2H Latch Status

Read 1: L2H latch occurred.

0: no L2H latch

Write 0: clear the L2H latch status</td></tr>
<tr><td>Get/Clear H2L Latch Status

Read 1: H2L latch occurred.

0: no H2L latch</td></tr>
</table></td></tr>
</table>

		Write 0: clear the H2L latch status DI change of status
value	int	It will be the frequency value when DI Mode is Frequency mode. It will be the counter value when DI Mode is Counter mode.

3.1.5 Digital Output

Field Name	Data Type	Description
Mode	string	DO Mode DO Pulse output Low to High delay High to Low delay AI alarm drive
status	Object	Signal Logic 1 / 0: Output signal is Logic High or Low Pulse Output Continue 1 / 0: Pulse outputting is continuous or not DO Change
PulsAbs	int	When DO mode is set in Pulse Output mode, this is the absolute pulse value.

PulsInc	int	When DO mode is set in Pulse Output mode, this is the incremental pulse value.

3.1.6 Analog Input

Field Name	Data Type	Description	
Range	int	0	-150mv~150mv
		1	-500mv~500mv
		2	-1v~1v
		3	-5v~5v
		4	-10v~10v
		5	0~150mv
		6	0~500mv
		7	0~1v
		8	0~5v
		9	0~10v
		10	4mA~20mA
		11	-20mA~20mA
		12	0~20mA
		13	Reserved
		14	Reserved
		15	PT100(385) -200~ +600℃

<table>
<tr><td></td><td></td><td><table>
<tr><td>16</td><td>PT100(392) -200~ +600℃</td></tr>
<tr><td>17</td><td>PT1000 -40~ +160℃</td></tr>
</table></td></tr>
<tr><td>status</td><td>Object</td><td><table>
<tr><td>Low Alarm Status
Read 1: low alarm occurred.
0: not occurred
Write 0: clear the low alarm status</td></tr>
<tr><td>High alarm status
Read 1: high alarm occurred.
0: not occurred
Write 0: clear the high alarm status</td></tr>
</table></td></tr>
<tr><td>Raw Data</td><td>int</td><td>AI value is the measurement raw data with range 0 to 0xFFFF.</td></tr>
<tr><td>Event</td><td>int</td><td><table>
<tr><th>Bit Order</th><th>Description</th></tr>
<tr><td>0</td><td>Fail to provide AI value (UART timeout, ADC error)</td></tr>
<tr><td>1</td><td>Over Range</td></tr>
<tr><td>2</td><td>Under Range</td></tr>
<tr><td>3</td><td>Open Circuit (Burnout)</td></tr>
<tr><td>4</td><td>Reserved</td></tr>
<tr><td>5</td><td>Unavailable Channel Configuration (Channel Disabled, DI Mode Used)</td></tr>
</table></td></tr>
</table>

		6	Reserved	
		7	ADC initializing/Error	
		8	Reserved	
		9	Zero/Span Calibration Error	
		10~15	Reserved	
MaxVal	int	The maximum of AI raw data with range 0 to 0xFFFF.		
MinVal	int	The minimum of AI raw data with range 0 to 0xFFFF.		

3.1.7 RS-485 Coil Data

Field Name	Data Type	Description		
Status	int	Status Value	Description	
		0 (0x00)	No error	
		1 (0x01)	Illegal function	
		2 (0x02)	Illegal data address	
		3 (0x03)	Illegal data value	
		4 (0x04)	Slave device failure	
		5 (0x05)	Acknowledge	
		6 (0x06)	Slave device busy	
		7 (0x07)	Negative acknowledge	

		8 (0x08)	Memory parity error	
		9 (0x09)	Reserved	
		10 (0x0A)	Gateway path unavailable	
		11 (0x0B)	Gateway target device failed to respond	
		12 ~15	Reserved	
		16 (0x10)	Unavailable	
		17 (0x11)	Slave response timeout	
		18 (0x12)	Checksum error	
		19 (0x13)	Received data error	
		20 (0x14)	Send request fail	
		21(0x15)	Unprocessed	
		22(0x16)	Read only	
		23(0x17)	In processing	
Data	int	The coil data, 0 or 1.		

3.1.8 RS-485 Register Data

Field Name	Data Type	Description
Status	Int	The error status of polling this channel can refer to Coil Status in the previous section.
Data	int	The register data with range 0 to 0xFFFF.

3.1.9 FFT (Fast Fourier Transform)

3.1.9.1 FFT in CSV format

Field Name	Data Type	Description
TIME	Int	FFT Data time.
AXIS_TYPE	String	X, Y, or Z Axis.
DATA	Int	FFT Data value, unit: mg.
LOG_INDEX	Int	(For packet transmission only) Index number of each FFT data round.
BYTE_OFFSET	Int	Byte offset in one FFT data round. Since each FFT data round contains 4800 bytes (2400 data), the range of the value is 0 to 4799.
SAMPLE_FREQ	Float	Frequency number of FFT data for each axis (X,Y,Z).

3.1.9.2 FFT in JSON format

Field Name	Data Type	Description
LOG_INDEX	Int	(For packet transmission only) Index number of each FFT data round.
TIME	Int	FFT Data time.
SAMPLING_RATE	Int	The average number of samples obtained in one second.

NUMBER_OF_SAMPLES	Int	Number of FFT Data samples.
START_BYTE_OFFSET	Int	Start of Byte offset in one FFT data round. Since each FFT data round contains 4800 bytes (2400 data), the range of the value is 0 to 4799.
END_BYTE_OFFSET	Int	End of Byte offset in one FFT data round. Since each FFT data round contains 4800 bytes (2400 data), the range of the value is 0 to 4799.
AXIS_DATA	Array Axis Data	Axis Data Object.

3.1.9.3 FFT Data Re-transmission

Field Name	Data Type	Description
LOG_INDEX	Int	(For packet transmission only) Index number of each FFT data round.
BYTE_OFFSET	Int	Byte offset in one FFT data round. Since each FFT data round contains 4800 bytes (2400 data), the range of the value is 0 to 4799.
LENGTH	Int	Length of lost FFT data, unit: byte.

3.1.10 Axis Data

Field Name	Data Type	Description
AXIS_TYPE	String	X, Y, or Z Axis.

START_ SAMPLE_INDEX	Int	Start of Index number of FFT data for each axis (X,Y,Z). Since each axis contains max 800 data, the range of the value is 0 to 799.
END_ SAMPLE_INDEX	Int	End of Index number of FFT data for each axis (X,Y,Z). Since each axis contains max 800 data, the range of the value is 0 to 799.
DATA	Array	Array of FFT Data value, unit: mg.

3.2. Sample Output


```
{
    "SequenceNumber": 110,
    "TotalLength": 45,
    "SourceAddress": null,
    "TempHumi": {
        "Range": 0,
        "Event": 0,
        "SenVal": 33312
    },
    "Accelerometer": {
        "X-Axis": {
            "SenEvent": 0,
            "OAVelocity": 530,
```

```
            "Peakmg": 369,
            "RMSmg": 261,
            "Kurtosis": -12,
            "CrestFactor": 359,
            "Skewness": 46,
            "Deviation": 24,
            "Peak-to-Peak Displacement": 6
        },
        "Y-Axis": {
            "SenEvent": 0,
            "OAVelocity": 174,
            "Peakmg": 257,
            "RMSmg": 182,
            "Kurtosis": -26,
            "CrestFactor": 654,
            "Skewness": 38,
            "Deviation": 26,
            "Peak-to-Peak Displacement": 4
        },
        "Z-Axis": {
            "SenEvent": 0,
```

```
            "OAVelocity": 264,
            "Peakmg": 231,
            "RMSmg": 164,
            "Kurtosis": 0,
            "CrestFactor": 0,
            "Skewness": 0,
            "Deviation": 0,
            "Peak-to-Peak Displacement": 0
        },
        "LogIndex": 10,
        "Time":1487060882
    },
    "DI0": {
        "status": {
            "Signal Logic": 0,
            "Start Counter": 1,
            "Get/Clean Counter Overflow": 0,
            "Get/Clean L2H Latch": 0,
            "Get/Clean H2L Latch": 0,
            "DI Change": 0
        },
```

```
        "mode": 0,
        "value": 0
    },
    "DI1": {
        "status": {
            "Signal Logic": 0,
            "Start Counter": 1,
            "Get/Clean Counter Overflow": 0,
            "Get/Clean L2H Latch": 0,
            "Get/Clean H2L Latch": 0,
            "DI Change": 0
        },
        "mode": 0,
        "value": 0
    },
    "DO0": {
        "Mode": "Pulse output",
        "status": {
            "Signal Logic": 0,
            "Pulse Output Continue": 0,
            "DO Change": 0
```

```
        },
        "PulsAbs": 0,
        "PulsInc": 0
    },
    "AI0": {
        "Range": 4,
        "status": {
            "Low Alarm": 0,
            "High Alarm": 0
        },
        "Raw Data": 32767,
        "Event": 0,
        "MaxVal": 32768,
        "MinVal": 32767
    },
    "AI1": {
        "Range": 4,
        "status": {
            "Low Alarm": 0,
            "High Alarm": 0
        },
```

```
        "Raw Data": 32768,
        "Event": 0,
        "MaxVal": 32768,
        "MinVal": 32767
    },
    "AI2": {
        "Range": 4,
        "status": {
            "Low Alarm": 0,
            "High Alarm": 0
        },
        "Raw Data": 32768,
        "Event": 0,
        "MaxVal": 32768,
        "MinVal": 32767
    },
    "AI3": {
        "Range": 4,
        "status": {
            "Low Alarm": 0,
            "High Alarm": 0
```

```
        },
        "Raw Data": 32767,
        "Event": 0,
        "MaxVal": 32768,
        "MinVal": 32767
    },
    "RtuCoil0-2": {
        "Status": 17,
        "Data": 0
    },
    "RtuRegister0-0": {
        "Status": 17,
        "Data": 0
    },
    "RtuCoil1-0": {
        "Status": 0,
        "Data": 1
    },
    "RtuCoil1-1": {
        "Status": 0,
        "Data": 1
```

```
},
"RtuRegister1-0": {
    "Status": 0,
    "Data": 65535
},
"Device": {
    "Events": 0,
    "PowerSrc": 2,
    "BatteryVolt": 4064,
    "Time": 1552645805,
    "GNSS": {
        "Latitude": "48.11730 N",
        "Longitude": "11.51666 W"
    }
},
"FFT": {
    "LOG_INDEX": 126,
    "TIME": 1552536694,
    "SAMPLING_RATE": 3200,
    "NUMBER_OF_SAMPLES": 2048,
    "START_BYTE_OFFSET": 1570,
```

```
"END_BYTE_OFFSET": 1581,
"AXIS_DATA": [
    {
        "AXIS_TYPE": "X",
        "START_SAMPLE_INDEX": 797,
        "END_SAMPLE_INDEX": 799,
        "DATA": [
            4,
            7,
            3
        ]
    },
    {
        "AXIS_TYPE": "Y",
        "START_SAMPLE_INDEX": 0,
        "END_SAMPLE_INDEX": 2,
        "DATA": [
            1160,
            580,
            1
        ]
```

```
            }

        ]

    }

}
```

4. Appendix

In FFT Data payload, user could check if there is any packet lost by checking the discontinuous of SequenceNumber or BYTE_OFFSET. If packet lost occurs, user could use FFT index, offset and lost length to get lost packet.