
Getting started with the FP-NET-6LPWIFI1 software package connecting 6LoWPAN IoT nodes to the Internet via Wi-Fi networks

Introduction

FP-NET-6LPWIFI1 is an STM32Cube function pack which lets you connect your IoT node in a 6LoWPAN wireless sensor network to the Internet via a Wi-Fi network.

The software, together with the suggested combination of STM32 and ST devices, can be used, for example, to develop smart home, building, lighting or remote monitoring applications.

The package contains sample applications to manage the 6LoWPAN devices through the OMA Lightweight M2M (LwM2M) protocol and to connect the devices to the IBM Watson IoT cloud services via the MQTT protocol.

The software runs on the STM32 microcontroller and includes drivers for the S2-LP sub-1GHz RF transceiver; the software also comes with ready-to-use binary firmware for wireless sensor nodes.

The software is based on STM32Cube technology and expands STM32Cube-based packages.

1 Acronyms and abbreviations

Table 1. List of acronyms

Acronym	Description
IDE	Integrated development environment
BSP	Board support package
HAL	hardware abstraction layer
UDP	User datagram protocol
6LoWPAN	IPv6 over low power wireless personal area networks
RPL	Routing protocol for low power and lossy networks
MCU	Microcontroller unit
MQTT	Message queuing telemetry transport
RF	Radio frequency
OS	Operating system
CoAP	Constrained application protocol
LWM2M	Lightweight machine to machine
IoT	Internet of things
MEMS	Micro electro-mechanical systems
Wi-Fi	Wireless LAN based on IEEE 802.11
GUI	Graphical user interface

2 FP-NET-6LPWIFI1 software expansion for STM32Cube

2.1 Overview

The [FP-NET-6LPWIFI1](#) software package expands STM32Cube functionality.

The key features of the package are:

- Complete firmware to connect 6LoWPAN and Wi-Fi networks
- Middleware libraries to support Contiki OS and Contiki 6LoWPAN protocol stack 3.x, MQTT protocol and Wi-Fi connectivity
- Support for mesh networking technology via the standard RPL protocol
- Sample applications to connect a 6LoWPAN network node to a remote server with the OMA Lightweight M2M (LWM2M) protocol or to Watson IoT cloud services provided by IBM
- Sample implementation available for the [X-NUCLEO-S2868A1](#) expansion board connected to a [B-L475E-IOT01A](#) Discovery board
- Easy portability across different MCU families, thanks to [STM32Cube](#)
- Free, user-friendly license terms

This software uses the Contiki OS to develop two sample applications:

- Wi-Fi bridge: to connect 6LoWPAN network nodes to the Internet with network level bridging (i.e. it carries the same protocol end-to-end);
- LWM2M to IBM: to connect an OMA Lightweight M2M network to the Watson IoT services provided by IBM.

6LoWPAN network communication is carried out by the sub-1GHz radio (X-NUCLEO-S2868A1, X-NUCLEO-IDS01A4 or X-NUCLEO-IDS01A5 expansion board). The Wi-Fi bridge forwards all packets destined to the Internet pass via Wi-Fi radio (X-NUCLEO-IDW01M1 expansion board).

Contiki OS technical details can be found at <https://github.com/contiki-os/contiki/wiki> in the **Internals** section. For information regarding Contiki APIs, refer to the documentation in the FP-NET-6LPWIFI1 package.

An open source implementation of the MQTT protocol (<http://www.eclipse.org/paho/>) ported to STM32 Nucleo is integrated in the package middleware; thus, the STM32 Nucleo-based microsystem can connect to the IBM Watson IoT cloud services.

MQTT is a lightweight messaging protocol with a small code footprint and low power and bandwidth use. It is particularly suitable for sensor data telemetry and implementation in embedded systems. Further information regarding the MQTT protocol is available at www.mqtt.org.

2.2 Architecture

This software is based on the STM32CubeHAL hardware abstraction layer for the STM32 microcontroller. The package extends [STM32Cube](#) by providing a board support package (BSP) for the Wi-Fi and the sub-1GHz RF communication expansion boards.

The drivers abstract low-level details of the hardware and allow the middleware components and applications to access sensor data in a hardware-independent manner to access and control the [S2-LP](#) sub-1GHz RF transceiver.

The package includes some middleware libraries to support Wi-Fi and 6LoWPAN stacks, along with a sample application accessing sensors and actuators on the 6LoWPAN nodes using standard protocols such as LWM2M and CoAP over UDP, and another sample application to connect the LWM2M devices to the Watson IoT cloud. Developers can use it to prototype end-to-end IoT applications.

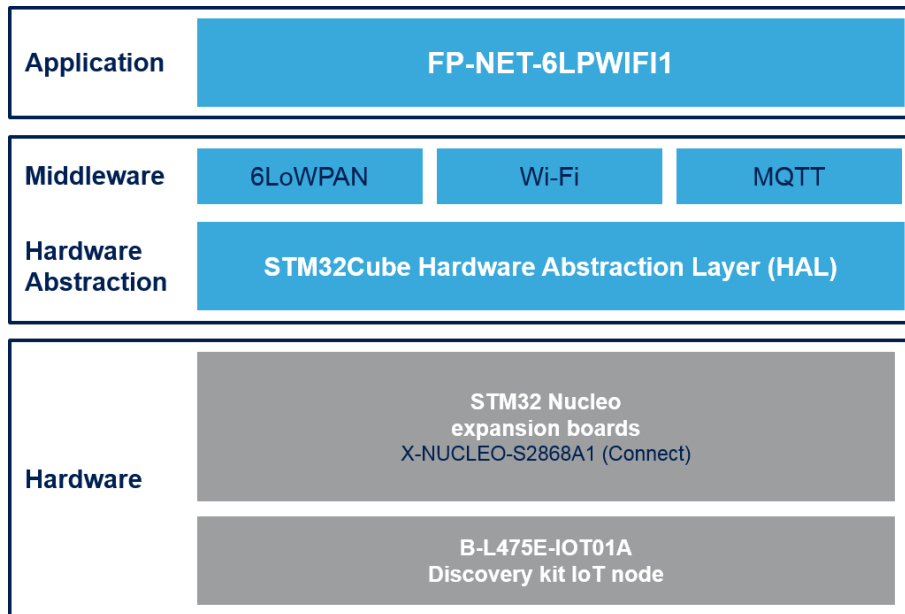
The application software accesses the X-NUCLEO-S2868A1, X-NUCLEO-IDS01A4 or X-NUCLEO-IDS01A5 and X-NUCLEO-IDW01M1 expansion boards via:

- The **STM32Cube HAL** driver layer, which provides a simple, generic, multi-instance set of application programming interfaces (APIs) to interact with the upper application, library and stack layers. It has generic and extension APIs and is directly built around a generic architecture and allows successive layers like the middleware layer to implement functions without requiring specific hardware configurations for a given

microcontroller unit (MCU). This structure improves library code reusability and facilitates portability to other devices.

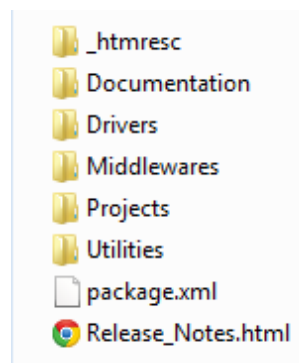
- The **board support package** (BSP) layer supports all the peripherals on the STM32 Nucleo except the MCU. This limited set of APIs provides a programming interface for certain board-specific peripherals like the LED, the user button, etc. This interface also helps in identifying the specific board version.

Figure 1. FP-NET-6LPWIFI1 software architecture



2.3 Folder structure

Figure 2. FP-NET-6LPWIFI1 package folder structure



The following folders are included in the software package:

- Documentation: contains a compiled HTML file generated from the source code which details the software components and APIs.
- Drivers: contains the HAL drivers and the board-specific drivers for each supported board or hardware platform, including the on-board components and the CMSIS vendor-independent hardware abstraction layer for ARM Cortex-M processor series.
- Middlewares: contains libraries for the Contiki OS, for the MQTT protocol, and the interface for the Wi-Fi expansion software.
- Projects: contains a sample application using application-level functions to bridge a 6LoWPAN network with a Wi-Fi network (UDP protocol support only), and a sample application for connecting a network of LwM2M client nodes to the IBM Watson IoT cloud. The projects are built for B-L475E-IOT01A Discovery board, and for NUCLEO-F401RE boards and the following development environments:

1. IAR Embedded Workbench for ARM
 2. RealView Microcontroller Development Kit (MDK-ARM)
 3. System Workbench for STM32 (SW4STM32)
- Utilities: contains ready-to-use firmware binaries that allow a node in the 6LoWPAN network to connect to the Internet and share its resources (e.g., sensors or actuators) with the OMA lightweight M2M protocol.

2.4 APIs

Detailed technical information with full user API function and parameter description are in a compiled HTML file in the “Documentation” folder.

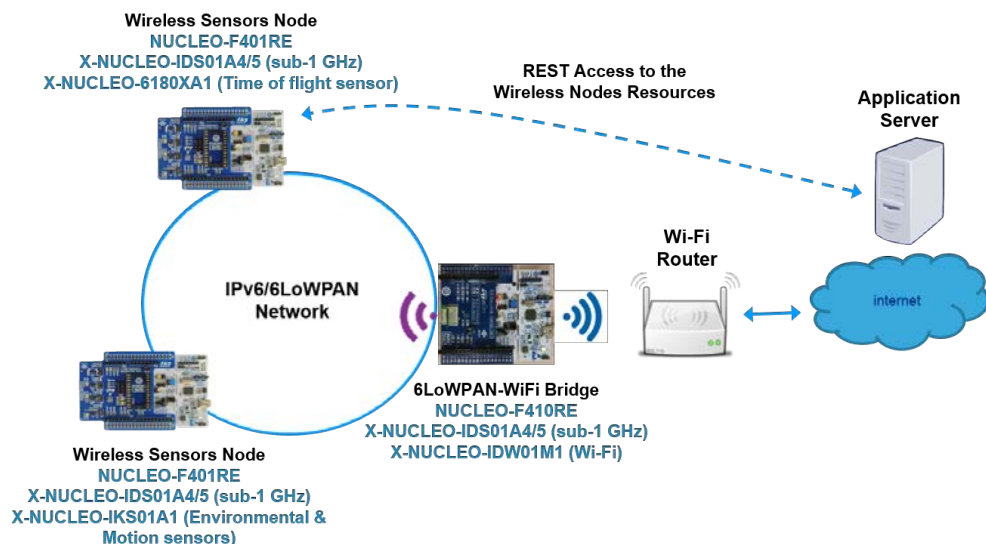
2.5 Sample application description

2.5.1 Wi-Fi bridge

This sample application with ready-to-build projects for multiple IDEs is located in the ‘Projects’ directory. Use it to create a bridge between a 6LoWPAN network and the Internet using Contiki OS and the Wi-Fi module. This application supports the UDP protocol and uses NAT64 technology to translate IPv6 packets to IPv4 packets. While the nodes connected to the Wi-Fi bridge issue direct requests to a specific server, it is actually the Wi-Fi bridge which connects the nodes to the requested server by transparently forwarding every packet from the 6LoWPAN network to the IPv4 network.

Note: For bidirectional communication, the application can support up to 4/8 sensor nodes simultaneously (the actual number depends on the Wi-Fi module in use as this limit is related to the max. number of connections that can be opened in parallel); for unidirectional node-to-server communication (e.g., sensor data transmission), this limit does not apply.

Figure 3. Overall system architecture (example with bridge based on NUCLEO-F401RE)



To run the application:

- Step 1.** Compile the project using one of the supported IDEs, see [Section 3.2 Software requirements](#)
- Step 2.** Program the firmware on the [STM32 Nucleo](#) board: you can copy (or drag and drop) the binary file to the USB mass storage location created when you plug the STM32 Nucleo board to your PC. If the host is a Linux PC, the STM32 Nucleo F4 device can be found in the /media folder with the name “NUCLEO_F401RE”. For example, if the created mass storage location is “/media/NUCLEO_F401RE”, then the command to program the board with a binary file named “my_firmware.bin” is simply: `cp my_firmware.bin /media/NUCLEO_F401RE`. Alternatively, you can program the STM32 Nucleo board directly through one of the supported development toolchains (please refer to the corresponding toolchain user manual for further information).

- Step 3.** Open a serial line monitor utility, select the serial port name to which the board is connected and configure it thus:
- Baud Rate = 115200
 - Parity = None
 - Bits = 8
 - Stopbits = 1
- Step 4.** Press the RESET (black) button on the STM32 Nucleo board. The following configuration prompt should appear.

Figure 4. Wi-Fi bridge configuration

```

COM3 - Tera Term VT
File Edit Setup Control Window EscapeCode Help
*****
* FP-NET-ELPWIFI Function Pack v2.0.0
* X-NUCLEO-IDW01M1 Wi-Fi Configuration.
* Wi-Fi Bridge Application
*****
(HAL 1.5.2.0)
Compiled Mar 29 2017 12:52:49 (opensta32)

Initializing the wifi module...
WiFi Initialised and Ready..

Starting configure procedure for SSID and PWD....
Keep pressed user button within next 5 seconds to set Wi-Fi Access Point parameters (SSID and PWD)
via serial terminal. Otherwise parameters saved to FLASH will be used.
...0
Read from FLASH:
  SSID ..... = STM
  Key ..... = STMdemoPWD
  Authentication .. = WPA2

AP settings set.
>>network present...connecting to AP...
*****
>>connected...

Successfully configured WiFi module and connected to the selected AP... Starting Contiki and SPIRITI configuration
  
```

- Step 5.** When asked, press the User Button to enter your own credentials (see [Figure 5. Wi-Fi bridge custom credentials](#)) or just wait five seconds to retrieve from FLASH the credentials (if any) you used the previous time (see [Figure 4. Wi-Fi bridge configuration](#))

Figure 5. Wi-Fi bridge custom credentials

```

COM3 - Tera Term VT
File Edit Setup Control Window EscapeCode Help
via serial terminal. Otherwise parameters saved to FLASH will be used.
...0
Read parameters from serial terminal.
>>running WiFi Scan...
STWLAN2
STGUEST
STSMARTMOBILE
NavSAC-WiFi
TP-LINK_3B2FB8
eduroam
STM
BronzeNet

Enter the SSID:
STM

Enter the password:
STMdemoPWD

Enter the encryption mode(0:Open, 1:WEP, 2:WPA2/WPA2-Personal):
2

SaveSSIDPasswordToMemory OK
AP settings set.
>>network present...connecting to AP...
*****
>>connected...

Successfully configured WiFi module and connected to the selected AP... Starting Contiki and SPIRITI configuration
  
```

- Step 6.** After successfully joining the selected Wi-Fi network, a list of the main system and Contiki parameters is printed, for user information and debugging purposes.

Figure 6. Wi-Fi bridge Contiki parameters

```

COM28 - Tera Term VT
Platform and Contiki parameters:
USE_STM32F4XX_NUCLEO: 1
USE_NUCLEO_I0303TA4: 1
USE_X_CUBE_IDM01M1: 1
RSSI_RX_THRESHOLD: -118.0
RSSI_TX_THRESHOLD: -107.0
POWER_DBM: 11.6
CHANNEL_SPACE: 20e3
FREQ_DEVIATION: 20e3
BANDWIDTH: 100.0e3
MODULATION_SELECT: FSX
DATARATE: 38400
IEEE802154_CONF_PANID: abcd
IEEE802154_CONF_FREQ: 868.0e6
IEEE802154_CONF_CHANNEL_NUMBER: 0
IEEE802154_CONF_NULLDRC_CONF_802154_AUTOACK: 0
IEEE802154_CONF_SICSLWPAN_CONF_MAC_MAX_PAYLOAD: 96
MIP_CONF_BUFFER_SIZE: 1300
MIP_CONF_ROUTER: 1
MIP_CONF_IPV6_RPL: 1
MIP_CONF_RPL_LEAF_ONLY: 0
MIP_CONF_QUEUEBUF_CONF_NUM: 16
MIP_CONF_ND6_REACHABLE_TIME: 600000
MIP_CONF_IP_FORWARD: 0
MIP_CONF_TCP_MSS: 150
MIP_CONF_RECEIVE_WINDOW: 150
MIP_CONF_TCP: 1
MIP_CONF_MAX_ROUTES: 30
MIP_CONF_NBR_TABLE_CONF_MAX_NEIGHBORS: 15
MIP_CONF_ND6_SEND_RA: 0
WITH_IPv6: 1

```

2.5.2 Wireless sensor node for Wi-Fi bridge application

To enable user deployment and testing of end-to-end solutions between wireless sensor nodes and internet servers, sample node applications are available in the form of source code and of pre-compiled and ready-to-use binaries for [STM32 Nucleo](#) platforms equipped with expansion boards.

These applications demonstrate how a node can connect to a remote server with the OMA lightweight M2M (LWM2M) standard. This technology uses CoAP (over UDP) to publish a node's resources in a standard (LWM2M) format and make them available online.

In these examples, the nodes attempt to connect to a public online server (<https://leshan.eclipseprojects.io>), which is a Java implementation of an LWM2M server. Leshan also implements a GUI that communicates with the server through a REST API (see <http://www.eclipse.org/leshan/>).

The reference software package for the implementation of wireless sensor nodes compatible with the Wi-Fi bridge is [FP-SNS-6LPNODE1](#), available at www.st.com.

Note: The Leshan public LWM2M server is used only for evaluation purposes, to demonstrate end-to-end connectivity between a wireless sensor node and an Internet connected server, thanks to the 6LoWPAN-to-Wi-Fi bridge in the [FP-NET-6LPWIF11](#) software package.

2.5.2.1 Connecting the remote node to the Internet

- Step 1.** Power the [STM32 Nucleo](#) board using a Mini-B USB cable connected to the PC.
- Step 2.** Program the firmware on the STM32 Nucleo board: you can copy (or drag and drop) the binary file (in the /Utility/Binary folder) to the USB mass storage that is automatically created when you connect the STM32 Nucleo board to your PC.
- Step 3.** Open a serial line monitor utility, select the serial port name to which the board is connected and configure it thus:
 - Baud Rate = 115200
 - Parity = None
 - Bits = 8
 - Stopbits = 1
- Step 4.** Press the RESET (black) button on the STM32 Nucleo board and wait for the node to complete the registration

Figure 7. Client node registered on the remote server

```
RD Client process started.
Looking for LWM2M server: 'leshan.eclipseprojects.io'
LWM2M Server Address:
::FFFF:23.97.187.154
RD Client started with endpoint '?ep=STL1-F7FF104EEFB1'
Serial number is
'0800f7ff104eefb1'
Registering with [::FFFF:23.97.187.154]:5683 lwm2m endpoint '?ep=STL1-F7FF104EEFB1' (binding mode
U): '<1/0>,<3/0>,<3200/0>,<3311/0>,<3311/1>,'
Request block 0 (size 39): '<1/0>,<3/0>,<3200/0>,<3311/0>,<3311/1>,'
Successfully registered!
```

Step 5. Open your web browser and navigate to <https://leshan.eclipseprojects.io>

Figure 8. Leshan server homepage

LESHAN					
				CLIENTS	SECURITY
				Connected clients: 2	
Client Endpoint	Registration ID	Registration Date	Last Update		
urn:imei:867997030026107	gsFleG0p9e	Apr 8, 2019 2:08:13 PM	Apr 9, 2019 11:32:39 AM ⓘ		
STL1-F7FF104EEFB1	0AefERApbx	Apr 9, 2019 11:29:21 AM	Apr 9, 2019 11:34:56 AM ⓘ		

Step 6. Click on the corresponding Client Endpoint

Figure 9. Leshan server client homepage

The screenshot shows the Leshan server client homepage. At the top, there is a navigation bar with the LESHAN logo and tabs for CLIENTS and SECURITY. Below the navigation bar, there is a search bar for Clients with the value 'STL1-F7FF104EEFB1' highlighted in a red box. To the right of the search bar are dropdown menus for Multi-value and Single-value, both set to TLV. The main content area is divided into two sections: LwM2M Server and Device. Each section contains a table of resources with their paths and associated actions (Observe, Read, Write, Delete, Exec).

Resource Name	Path	Actions
Instance 0	/1/0	Observe, Read, Write, Delete
Short Server ID	/1/0/0	Observe, Read
Lifetime	/1/0/1	Observe, Read, Write
Default Minimum Period	/1/0/2	Observe, Read, Write
Default Maximum Period	/1/0/3	Observe, Read, Write
Disable	/1/0/4	Exec
Disable Timeout	/1/0/5	Observe, Read, Write
Notification Storing When Disabled or Offline	/1/0/6	Observe, Read, Write
Binding	/1/0/7	Observe, Read, Write
Registration Update Trigger	/1/0/8	Exec

Resource Name	Path	Actions
Instance 0	/3/0	Observe, Read, Write, Delete
Manufacturer	/3/0/0	Observe, Read
Model Number	/3/0/1	Observe, Read
Serial Number	/3/0/2	Observe, Read
Firmware Version	/3/0/3	Observe, Read
Reboot	/3/0/4	Exec
Factory Reset	/3/0/5	Exec
Available Power Sources	/3/0/6	Observe, Read
Power Source Voltage	/3/0/7	Observe, Read
Power Source Current	/3/0/8	Observe, Read

Step 7. Observe or read resources

Figure 10. Leshan server, observing and reading resources

Supported Binding and Modes		/3/0/16	Observe ▶	Read	
IPSO Temperature /3303					
Instance 0	/3303/0	Observe ▶	Read	Write Delete 25.69921875	
Min Measured Value	/3303/0/5601	Observe ▶	Read		
Max Measured Value	/3303/0/5602	Observe ▶	Read		
Min Range Value	/3303/0/5603	Observe ▶	Read		
Max Range Value	/3303/0/5604	Observe ▶	Read		
Reset Min and Max Measured Values	/3303/0/5605	Exec ⚙			
Sensor Value	/3303/0/5700	Observe ▶	Read		
Sensor Units	/3303/0/5701	Observe ▶	Read		
IPSO Humidity /3304					
Instance 0	/3304/0	Observe ▶	Read		Write Delete 54.296875
Min Measured Value	/3304/0/5601	Observe ▶	Read		
Max Measured Value	/3304/0/5602	Observe ▶	Read		
Min Range Value	/3304/0/5603	Observe ▶	Read		
Max Range Value	/3304/0/5604	Observe ▶	Read		
Reset Min and Max Measured Values	/3304/0/5605	Exec ⚙			
Sensor Value	/3304/0/5700	Observe ▶	Read		
Sensor Units	/3304/0/5701	Observe ▶	Read		
IPSO Accelerometer /3313					
Instance 0	/3313/0	Observe ▶	Read	Write Delete -0.3955078125	
Min Range Value	/3313/0/5603	Observe ▶	Read		
Max Range Value	/3313/0/5604	Observe ▶	Read		
Sensor Units	/3313/0/5701	Observe ▶	Read		
X Value	/3313/0/5702	Observe ▶	Read		
Y Value	/3313/0/5703	Observe ▶	Read		
Z Value	/3313/0/5704	Observe ▶	Read		
IPSO Digital Input /3200					

2.5.3 LwM2M to IBM

This sample application with ready-to-build projects for multiple IDEs is located in the **Projects** directory.

Use it to connect OMA LwM2M devices (implemented with the firmware available in the FP-SNS-6LPNODE1 function pack) to IBM Watson IoT cloud services via Wi-Fi.

According to a software template (see [Section 2.5.3.1 Template](#)) the user can pre-program the LwM2M_to_IBM gateway to automatically initiate a LwM2M observe/notify process or to issue periodic LwM2M read on predefined resources (i.e., humidity value and temperature max. value) on all 6LoWPAN network nodes.

This sample application implements an OMA LwM2M server that manages the OMA LwM2M clients inside the WSN network. As opposed to the Wi-Fi bridge sample application scenario, the LwM2M protocol is used only inside the 6LoWPAN network. Through the Wi-Fi module, it then connects to IBM Watson cloud, using the MQTT protocol.

This application is configured by default to send data via Wi-Fi to the IBM Watson IoT cloud in quickstart mode for data visualization only. However, it can be quickly modified to use the device in register mode. The latter requires an account on the IBM Watson IoT cloud (if the user wants to command the device; e.g., turn LED LD2 on or off). Further details are provided in the [FP-CLD-WATSON1](#) function pack documentation available at www.st.com.

2.5.3.1 Template

The LwM2M to IBM gateway can automatically and periodically retrieve some specific resources values from the discovered nodes, as follows:

- using the observe/notification operation of LwM2M (mapped in a CoAP GET command including the Observe option), for the resources that support this option: the client node notifies the server when the resource value changes;

- or using the LWM2M read operation (mapped in a simple CoAP GET command) which is executed in polling mode.

You can find the template at the beginning of the file *Projects/Multi/Applications/LWM2M_to_IBM/Src/lwm2m-resource-directory.c*.

Figure 11. Template code for automatic sensor value retrieval

```

/* Syntax:
 * TEMPLATE (name,
 *          IPSO_RESOURCE_TYPE (OBJECT_ID, RESOURCE_ID, OPERATION, ALTERNATIVE_NAME),
 *          );
 *
 *          Note: The IPSO types as well as all the needed #define are in lwm2m-simple-server.h and lwm2m-defs.h
 */
TEMPLATE (template,
        IPSO_RESOURCE_FLOAT (IPSO_TEMPERATURE_SENSOR, SENSOR_VALUE, OPR_TO_OBSERVE, TEMPERATURE_VALUE_STR),
        IPSO_RESOURCE_FLOAT (IPSO_TEMPERATURE_SENSOR, MAX_MEASURED_VALUE, OPR_TO_GET, TEMPERATURE_MAX_VALUE_STR),
        IPSO_RESOURCE_INT (IPSO_BUTTON, BUTTON_COUNTER, OPR_TO_GET, BUTTON_COUNTER_STR),
        IPSO_RESOURCE_FLOAT (IPSO_ACCELEROMETER, Z_VALUE, OPR_TO_OBSERVE, ACCELERATION_Z_STR),
        IPSO_RESOURCE_FLOAT (IPSO_HUMIDITY_SENSOR, SENSOR_VALUE, OPR_TO_OBSERVE, HUMIDITY_VALUE_STR),
        );

```

The actual macro definitions can be found in the file *Projects/Multi/Applications/LWM2M_to_IBM/Inc/lwm2m-simple-server.h*. The LWM2M and IPSO related objects and resource definitions are included in the file *Projects/Multi/Applications/LWM2M_to_IBM/Inc/lwm2m-defs.h*, in compliance with the “OMA Lightweight Machine to Machine Technical Specification V1.0” and “IPSO SmartObject Guideline - Smart Objects Starter Pack1.0” documents.

The rationale is that by adding in the template a line like `IPSO_RESOURCE_TYPE (OBJECT_ID, RESOURCE_ID, OPERATION_TYPE, ALTERNATIVE_NAME)`, the LWM2M to IBM gateway automatically retrieves the URI `OBJECT_ID/INSTANCE_ID/RESOURCE_ID`

This is done for any object instance and for all the network nodes.

The type of resource is specified by `IPSO_RESOURCE_TYPE`:

- `IPSO_RESOURCE_FLOAT` for floating point values (i.e. temperature)
- `IPSO_RESOURCE_INT` for integer values (i.e. button counter)
- `IPSO_RESOURCE_STRING` for text resources (i.e. manufacturer)

The type of retrieval is specified by `OPERATION`:

- `OPR_TO_GET` for read operations (in loop)
- `OPR_TO_OBSERVE` for observe/notifications

The `ALTERNATIVE_NAME` is an optional string to send the datapoints to IBM cloud in a more readable way compared to the LWM2M/IPSO semantic.

This behavior can be triggered by the `USE_LWM2M_FORMAT_FOR_DATAPOINT_NAME` macro, in the project settings.

If this macro is set to 0 (this is the default value), the `ALTERNATIVE_NAME` is used, so with current value of the `TEMPERATURE_VALUE_STR`, the temperature value of node 0 datapoint is sent as `Temperature_Node_0`.

Otherwise the full LWM2M/IPSO semantic is used and the same datapoint is sent as `3303/0/5700_Node_0` with 3303 being the object ID for IPSO temperature, 0 the Instance ID and 5700 the Resource ID for sensor value.

Other URIs to be observed or periodically read can be added by sending commands from IBM Watson IoT to the LWM2M gateway, connecting the latter as a registered device.

2.5.3.2 Sensor data visualization in IBM Quickstart

2.5.3.2.1 Setup based on STM32 Nucleo

The LWM2M to IBM sample application contains a default configuration to connect the [STM32 Nucleo](#) board to IBM Watson IoT cloud, enabling to display sensor data on an IBM web page (<https://quickstart.internetofthings.ibmcloud.com>), depending on the device MAC address.

Once the application is configured and running, you can use a serial line monitor to view messages from the STM32 Nucleo board and to configure the SSID and password for connecting the board to an available Wi-Fi network.

Figure 12. Startup and Wi-Fi credential configuration (STM32 Nucleo)

```

COM28 - Tera Term VT
File Edit Setup Control Window KanjiCode Help

/*****
 *
 * FP-NET-6LPWIF11 Function Pack v2.0.0
 * X-NUCLEO-IDW01M1 Wi-Fi Configuration.
 * LWM2M to IBM Cloud Application
 *
 *****/
(HAL 1.5.2_0)
Compiled Mar 29 2017 18:09:47 (openstm32)

Initializing the wifi module...
WiFi Initialised and Ready..

Starting configure procedure for SSID and PWD...
Keep pressed user button within next 5 seconds to set Wi-Fi Access Point parameters (SSID and PWD)
via serial terminal. Otherwise parameters saved to FLASH will be used.
..0
Read from FLASH:
  SSID ..... = STM
  Key ..... = STMdemoPWD
  Authentication .. = WPA2

AP settings set.
[D]. WiFi Board MAC Address: '00:80:E1:B4:D2:39'

[D] IBM Quickstart URL
https://quickstart.internetofthings.ibmcloud.com/#/device/0080E1B4D239/sensor/
    
```

After connection to a Wi-Fi access point, the application shows the Wi-Fi expansion board MAC address and the IBM Quickstart URL in the serial console.

2.5.3.2.2 Setup based on the Discovery kit IoT node (B-L475E-IOT01A)

When using the IoT Discovery board the Wi-Fi, the provisioning phase is slightly different. The first time the board is powered on, you must enter the Wi-Fi credentials and paste the certificate in the `Projects\B-L475E-IOT01A\Applications\LWM2M_to_IBM\Common\Bluemix\comodo_bluemix.pem` file (these parameters will be saved in the flash memory, so you will not be asked to insert them anymore).

Figure 13. Startup and Wi-Fi credential configuration for Discovery kit

```

COM25 - Tera Term VT
File Edit Setup Control Window Help

Starting WiFi configuration

*****
*** STM32 IoT Discovery kit For STM32L475 MCU ***
*** 6LPwIFI1 3.0.0 - OMA-LWM2M to IBM Application ***
*****

*** Board personalization ***

*** WIFI connection ***

Your WiFi parameters need to be entered to proceed.

Enter SSID: You have entered IBM_CloudTest as the ssid.

Enter Security Mode (0 - Open, 1 - WEP, 2 - WPA, 3 - WPA2):
You have entered 2 as the security mode.

Enter password:
Initializing the WiFi module
Module initialized successfully: Inventek e5-WiFi ISM43362-M3G-L44-SPI C3.5.2.4.STM.BET-SPI
Retrieving the WiFi module MAC address: c4:7f:51:04:2d:b5

Connecting to AP: IBM_CloudTest Attempt 1/3 ...
Connected to AP IBM_CloudTest
Mac address: c4:7f:51:04:2d:b5
Retrieving the IP address.
IP address: 192.168.1.5

Updating TLS security credentials.

Enter the x509 certificates or keys as per the following format:
-----BEGIN CERTIFICATE-----
YMPGn8u67GB9t+aEMr5P+1gmIqNb1LTV+/Xj1i5ww0QuvfWu7uJBVcA0Ln0kcmnL
R7EUQIN9Z/SG9jGr8XmksrUuEvmEF/Bibyc+E1ixVA0hmmM3oTDPb5Lc9un8rNsu
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
YMPGn8u67GB9t+aEMr5P+1gmIqNb1LTV+/Xj1i5ww0QuvfWu7uJBVcA0Ln0kcmnL
    
```

- Step 1.** Choose the **Registration mode** (1 for Quickstart)
- Step 2.** Enter a connection string in the form DeviceType=MY_DEV_TYPE;DeviceId=MY_DEV_ID.

Important: The DeviceId must be unique.

Figure 14. Startup and Wi-Fi credential configuration for Discovery kit (continued)

```

COM25 - Tera Term VT
File Edit Setup Control Window Help

060/nVsJ8dM041P0jmP6P6fbtGbFymbW0W5BjFITtep3Sp+dw0IrwC8AI+0tK1JF
Pn1lki aY4IBIqDfv8NZ5YBber0gOzw6sRBc4L0na4UU+Krk2U886UAb3LujEV01s
ySEY1QStedNso0Brp+uvFRTP2InBuThs4pFstiv9kuXc1VzDAGySj4dZp30d8tbQk
CAUw7C29C79Fv1C5qFPrmAESrciIxp90X40KPMbp1ZwVbd4=
-----END CERTIFICATE-----

Enter Registration Mode (1 - Quickstart, 2 - Simple):
You have selected the Quickstart registration mode.

Enter the Bluemix connection string of your device: (template: DeviceType=xxx;DeviceId=xxx)
connection string: -->
QuickStart;DeviceType=lwm2m;DeviceId=123454321

Setting the RTC from the network time.
Configuring the RTC from Date: Mon, 30 Jul 2018 09:40:45 GMT
Device Client Connected to quickstart.messaging.internetofthings.ibmcloud.com Platform in QuickStart Mode

You can see your published data at https://quickstart.internetofthings.ibmcloud.com
Device Id is 123454321

IPv6 addresses:
aaaa:b50:344e:b035:5920
aaaa:ff:fe00:1
fe80::b50:344e:b035:5920
    
```

- Step 3.** Whichever the case (use of STM32 Nucleo or Discovery kit IoT node), power one or more wireless sensor nodes ("ipso-mems" node in this example) on.
- Step 4.** Wait for the nodes to register to the LWM2M server implemented by the LWM2M_to_IBM application.

2.5.3.3 Connection to IBM Watson IoT cloud as registered device

2.5.3.3.1 Setup based on STM32 Nucleo

To register the **STM32 Nucleo**-based microsystem to IBM Watson IoT cloud, you must create an account on IBM Watson cloud by following the instructions provided at <https://www.ibm.com/marketplace/cloud/internet-of-things-cloud/us/en-us> (for step-by-step screenshot details, see the **FP-NET-6LPWIFI1** Quick Start Guide, available at www.st.com).

Once you have signed in to the IBM Watson IoT platform and you have selected the Internet of Things service, it is possible to register a new device. During the device registration procedure, several device properties are provided (they are also used to configure the hardware during the setup phase):

- organization ID (e.g. "cx44f7");
- device type (e.g. "LWM2M_device");
- authentication method (only the "use-token-auth" method is supported);
- authentication token (e.g. "6qZ60XfM!nPdx3c*m").

These properties must then be copied into the `Config_MQTT_IBM` function in the file `Projects/Multi/Applications/LWM2M_to_IBM/Src/IBM_Bluemix_Config.c` and `ibm_mode` must be set to "REGISTERED" as shown below.

Figure 17. Configuration for registering the STM32 device on IBM Watson IoT cloud

```
void Config_MQTT_IBM ( MQTT_vars *mqtt_ibm_setup, uint8_t *macadd )
{
    /* Default Configuration for QUICKSTART. REGISTERED mode requires account on Bluemix */
    mqtt_ibm_setup->ibm_mode = REGISTERED;

    /* Quickstart visualization */
    if ( mqtt_ibm_setup->ibm_mode == QUICKSTART )
    {
        strcpy((char*)mqtt_ibm_setup->pub_topic, "iot-2/evt/status/fmt/json");
        strcpy((char*)mqtt_ibm_setup->sub_topic, "");
        strcpy((char*)mqtt_ibm_setup->clientid, "d:quickstart:nucleo:");
        strcat((char*)mqtt_ibm_setup->clientid, (char *)macadd);
        mqtt_ibm_setup->qos = QOS0;
        strcpy((char*)mqtt_ibm_setup->username, "");
        strcpy((char*)mqtt_ibm_setup->password, "");
        strcpy((char*)mqtt_ibm_setup->hostname, "quickstart.messaging.internetofthings.ibmcloud.com");
        strcpy((char*)mqtt_ibm_setup->device_type, "");
        strcpy((char*)mqtt_ibm_setup->org_id, "");
        mqtt_ibm_setup->port = 8883; //TLS
        mqtt_ibm_setup->protocol = 's'; // TLS no certificates
    }
    else
    {
        /* REGISTERED DEVICE */
        /* Need to be customized */
        strcpy((char*)mqtt_ibm_setup->pub_topic, "iot-2/evt/status/fmt/json" ); // "iot-2/evt/status/fmt/json"
        strcpy((char*)mqtt_ibm_setup->sub_topic, "iot-2/cmd/+fmt/json");
        mqtt_ibm_setup->qos = QOS0;
        strcpy((char*)mqtt_ibm_setup->username, "use-token-auth");
        strcpy((char*)mqtt_ibm_setup->password, "6qZ60XfM!nPdx3c*m");
        strcpy((char*)mqtt_ibm_setup->hostname, "cx44f7.messaging.internetofthings.ibmcloud.com");
        strcpy((char*)mqtt_ibm_setup->device_type, "LWM2M_device");
        strcpy((char*)mqtt_ibm_setup->org_id, "cx44f7");

        strcpy((char *)mqtt_ibm_setup->clientid, "d:");
        strcat((char *)mqtt_ibm_setup->clientid, (char *)mqtt_ibm_setup->org_id);
        strcat((char *)mqtt_ibm_setup->clientid, ":");
        strcat((char *)mqtt_ibm_setup->clientid, (char *)mqtt_ibm_setup->device_type);
        strcat((char *)mqtt_ibm_setup->clientid, ":");
        strcat((char*)mqtt_ibm_setup->clientid, (char *)macadd);
        mqtt_ibm_setup->port = 8883; //TLS
        mqtt_ibm_setup->protocol = 's'; // TLS no certificates
    }

    return;
}
```

2.5.3.3.2 Setup based on the Discovery kit IoT node

To register the **Discovery**-based microsystem to IBM Watson IoT cloud, you must create an account on IBM Watson cloud by following the instructions provided at <https://www.ibm.com/marketplace/cloud/internet-of-things-cloud/us/en-us> (for step-by-step screenshot details, see the **FP-NET-6LPWIFI1** Quick Start Guide, available at www.st.com).

Once you have signed in to the IBM Watson IoT platform and you have selected the Internet of Things service, it is possible to register a new device. During the device registration procedure, several device properties are provided (they are also used to configure the hardware during the setup phase):

- organization ID (e.g. “yuzagl”);
- device type (e.g. “lwm2m”);
- authentication method (only the “use-token-auth” method is supported);
- authentication token (e.g. “6zfm)SlpQbr7r6jz23”).

The “Simple Registered” mode can be chosen during the boot phase of the Discovery board. If you already have used it in Quickstart mode, you can boot normally and reply “y” only when prompted if you want to update the device credentials and choose option 2 (Simple) as registration mode.

Figure 18. “LWM2M to IBM” setup with Discovery kit

```

COM25 - Tera Term VT
File Edit Setup Control Window Help
*****
*** STM32 IoT Discovery kit for STM32L475 MCU ***
*** 6LPwIF11 3.0.0 - OMA-LWM2M to IBM Application ***
*****

*** Board personalization ***

*** WIFI connection ***

Push the User button (Blue) within the next 5 seconds if you want to update the WiFi network configuration.

Initializing the WiFi module
Module initialized successfully: Inventek eS-WiFi ISM43362-M3G-L44-SPI C3.5.2.4.STM.BET-SPI
Retrieving the WiFi module MAC address: c4:7f:51:04:2d:b5

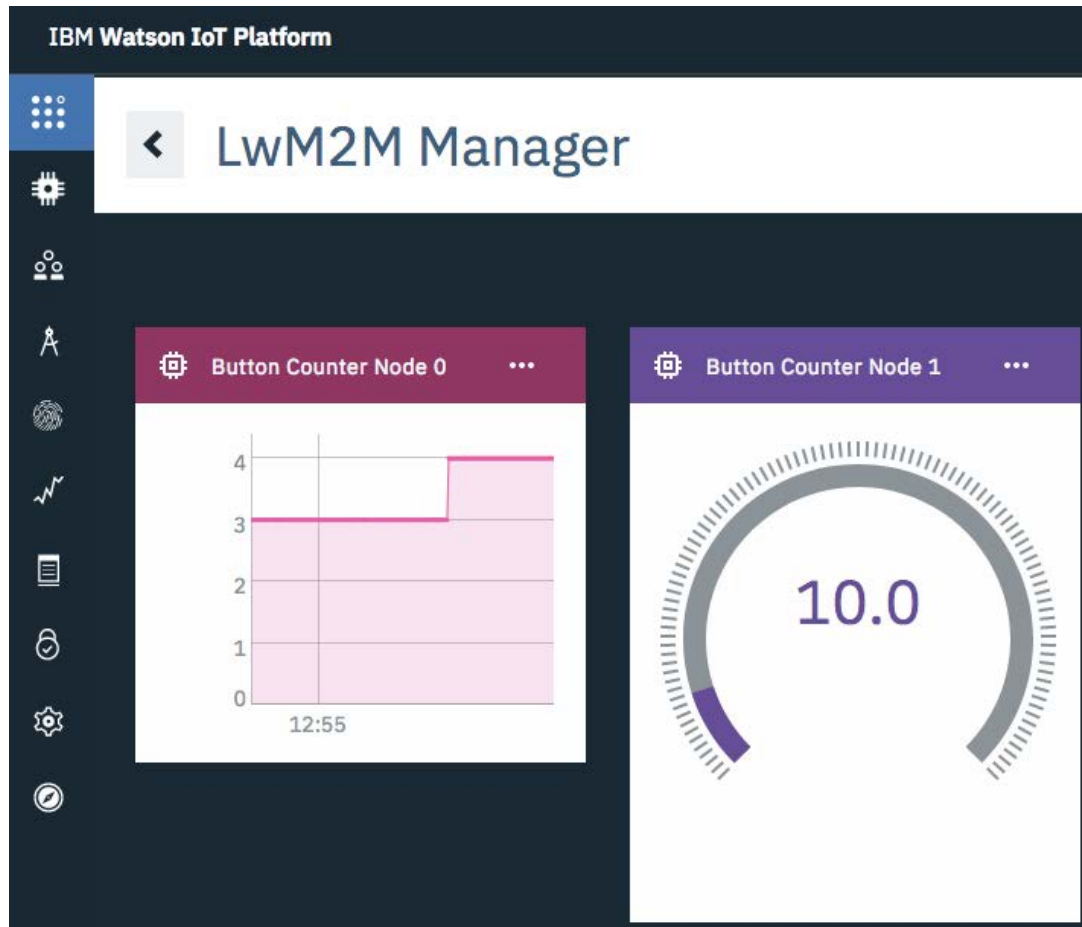
Connecting to AP: IBM_CloudTest Attempt 1/3 ...
Connected to AP IBM_CloudTest
Mac address: c4:7f:51:04:2d:b5
Retrieving the IP address.
IP address: 192.168.1.5
Push the User button (Blue) within the next 5 seconds if you want to update the device security parameters or credential
$.

Updating TLS security credentials.

Enter the x509 certificates or keys as per the following format:
-----BEGIN CERTIFICATE-----
YMPGn8u67GB9t+aEMr5P+1gmIgb1LTV+/Xj115wwOQuvfwu7uJ8VcA0Ln0kcmmL
R7EUQIN9Z/SG9jGr8XmksrUuEvmEF/Bibyc+E1ixVA0hmmM3oTDPb5Lc9un8rNsu
-----
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
YMPGn8u67GB9t+aEMr5P+1gmIgb1LTV+/Xj115wwOQuvfwu7uJ8VcA0Ln0kcmmL
-----
-----END CERTIFICATE-----
Do you want to update the root CA certificate(s)? [y/n]
y
Do you want to update your way to register, or the device credentials? (y/n)
y
Enter Registration Mode (1 - Quickstart, 2 - Simple):2
    
```

The parameters listed above must be entered in a single registration string (for example, OrgId=yuzagl;DeviceType=lwm2m;DeviceId=123454321;Token=6zfm)SlpQbr7r6jz23) when prompted, as shown in the next figure.

Figure 21. Dashboard example

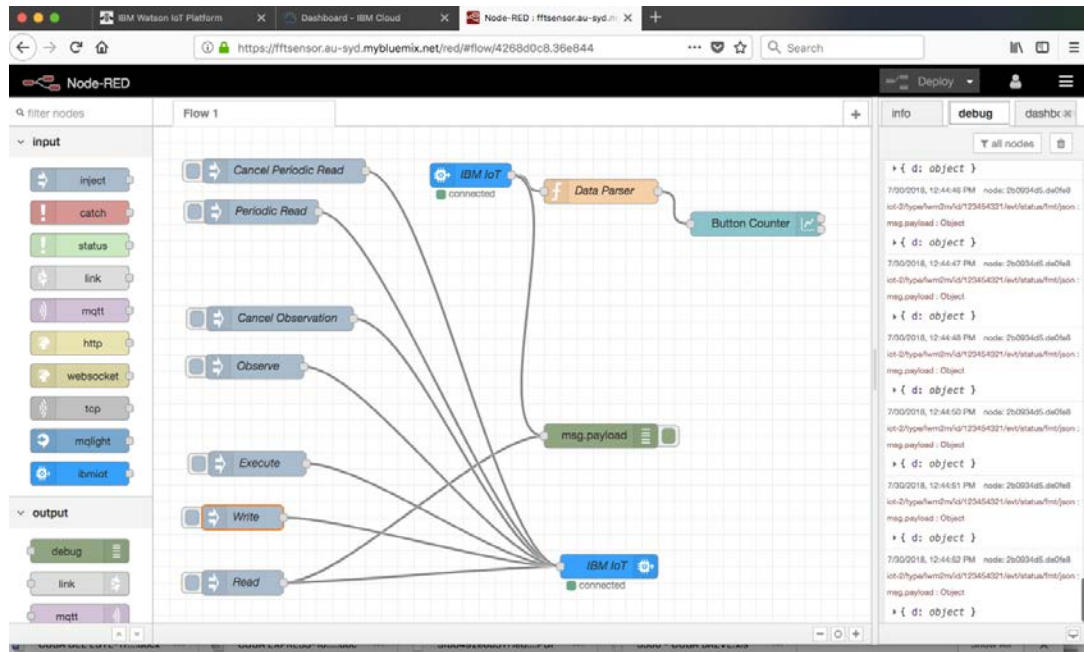


2.5.3.3.3 NodeRed

Once registered and connected to IBM Watson IoT cloud, the STM32 Nucleo or Discovery-based microsystem can send and receive data to and from IBM cloud applications and IoT services. More information about how to develop cloud applications and services can be found at <https://www.ng.bluemix.net/docs/starters/Node-RED/nodered.html>.

The following figure shows the screenshot of a possible Node-RED program in which the supported commands are shown.

Figure 22. Node-RED program blocks



The LWM2M to IBM application currently supports seven possible types of commands that can be sent from the IBM Bluemix application to the LWM2M gateway; the generic form is an MQTT message with a JSON payload containing:

- {"Node":nodeID
- "Path":"resourceUri"
- "Type":String/Int/Float
- "Operation":"Read/Write/Execute/Observe/Cancel/PRead/CRead"
- "Value": value }

In this way, it is possible to directly map five LWM2M standard operations (read, write, execute, observe and cancel observation) and to manage the periodic reads through the custom operations, *PRead* (to add a resource to this list) and *CRead* (to cancel this reporting), useful as the number of observable resources on each LWM2M client is limited.

The "*resourceUri*" path must be a valid resource path as defined by the LWM2M standard and "*nodeID*" is the index of the node the command is intended for.

The "*Type*" identifies the data type according to LWM2M / IPSO data model. The "*Value*" field is meaningful only for the Write operation.

The example implements the following operations:

1. Read the Float value of X axis (resource ID: 5702) of instance 0 of the magnetometer sensor (object ID: 3314), from node 0:
{"Node":0,"Path":"3314/0/5702","Operation":"Read","Type":"Float","Value":0}
2. Set to the Integer value=1 the On/Off (resource ID: 5850) attribute of instance 0 of the light control (object ID: 3311, that is used in this case to map the green LED), on node 0 (i.e. to switch the green LED on):
{"Node":0,"Path":"3311/0/5850","Operation":"Write","Type":"Int","Value":1}
3. Execute the Digital Input Counter Reset (resource ID: 5505) of instance 0 of the digital input (object ID: 3200, that is used in this case to map the user button):
{"Node":0,"Path":"3200/0/5505","Operation":"Execute","Type":"Int","Value":1}
4. Observe the Float value of Z axis (resource ID: 5704) of instance 0 of the magnetometer sensor (object ID: 3314), from node 0:
{"Node":0,"Path":"3314/0/5704","Operation":"Observe","Type":"Float","Value":0}

5. Cancel the observation of the Float value of Z axis (resource ID: 5704) of instance 0 of the magnetometer sensor (object ID: 3314), from node 0:
`{"Node":0,"Path":"3314/0/5704","Operation":"Cancel","Type":"Float","Value":0}`
6. Add to the periodic Read list the Float value of Y axis (resource ID: 5703) of instance 0 of the magnetometer sensor (object ID: 3314), from node 0:
`{"Node":0,"Path":"3314/0/5703","Operation":"PRead","Type":"Float","Value":0}`
7. Remove from the periodic Read list the Float value of Y axis (resource ID: 5703) of instance 0 of the magnetometer sensor (object ID: 3314), from node 0:
`{"Node":0,"Path":"3314/0/5703","Operation":"CRead","Type":"Float","Value":0}`

For instance, by executing the example described in point 6, this new datapoint is added to the other ones (pre-programmed via the template).

Note: In this case the datapoint name is actually the full resource path.

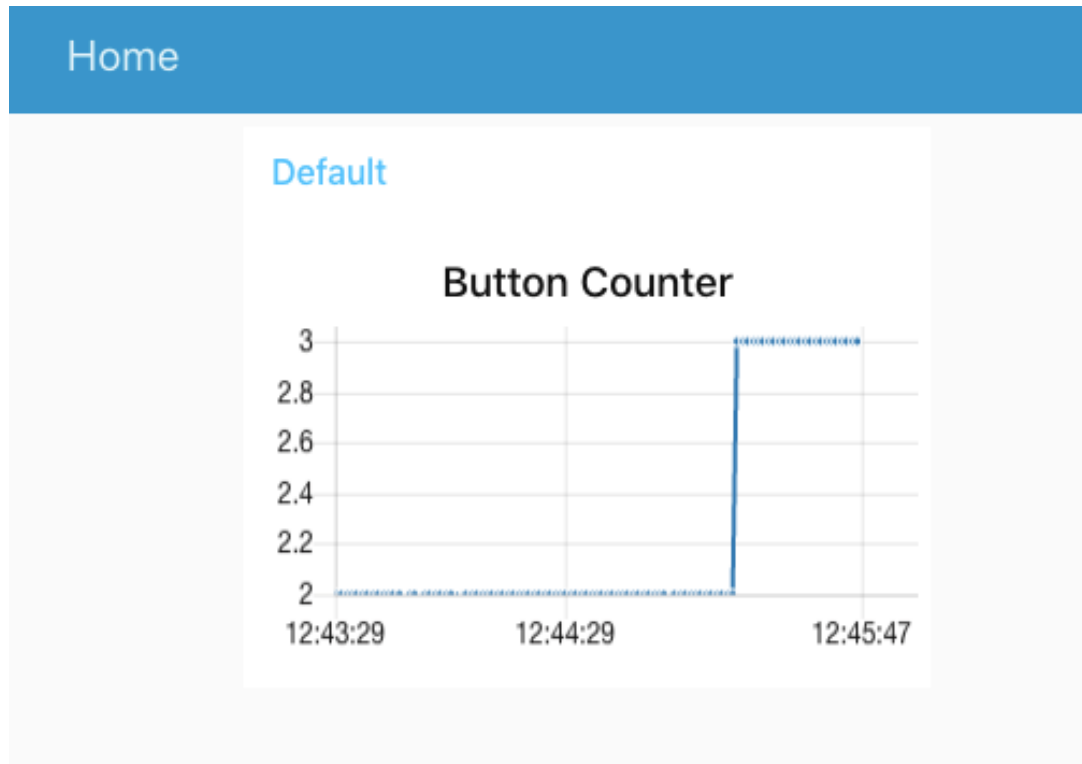
Figure 23. Sensor data visualization in the IBM web page for a registered device

status	d.myName	Node6LoWPAN	30 mar 2017 19:37:33
status	d.3314/0/5703_Node_0	0.51	30 mar 2017 19:37:30
status	d.Acceleration_Z_Node_0	0.96	30 mar 2017 19:37:33
status	d.Temperature_Node_0	28.7	30 mar 2017 19:37:27
status	d.Button_Counter_Node_0	3	30 mar 2017 19:37:12

Both the Node-RED program and the LWM2M to IBM code can be easily modified to address different needs (for example, reading different resources, addressing the nodes by means of IPv6 address, adding other commands for the gateway).

Figure 22. [Node-RED program blocks](#) shows a data parser and a dashboard that can be used to add a real-time web-based dashboard visualization as shown below.

Figure 24. Button counter on the web-based dashboard



The source code related to the gateway functionality can be found in the file `Projects/Multi/Applications/LWM2M_to_IBM/Src/lwm2m-simple-server.c`, inside the `PROCESS_THREAD(command_process, ...)` at the bottom of the file.

The Node-RED program source code can be found in `\Utilities\NodeRED\LWM2M.json` file. Simply copy and paste it into the Node-RED Flow Editor to be imported. You just have to adapt the application keys.

2.5.4 Wireless sensor node for LWM2M to IBM application

To enable user deployment and testing of end-to-end solutions between wireless sensor nodes and Internet based servers, some sample node applications are provided in the form of pre-compiled and ready-to-use binaries for STM32 Nucleo board equipped with [STM32 Nucleo](#) expansion boards. The applications export various resources, depending on the type of expansion boards stacked on the [NUCLEO-F401RE](#) board.

The supported hardware configurations and application examples for the wireless sensor nodes are the following (see [Section 3.3.1 Hardware setup](#) for more details):

1. NUCLEO-F401RE plus X-NUCLEO-IDS01A4 or X-NUCLEO-IDS01A5 (see example in `/Utilities/Binary/LWM2M_to_IBM/Ipso_Example_Nosensors/`): it exports common resources like LEDs and user buttons.
2. NUCLEO-F401RE plus X-NUCLEO-IDS01A4 or X-NUCLEO-IDS01A5 plus X-NUCLEO-IKS01A1 (see example in `/Utilities/Binary/LWM2M_to_IBM/Ipso_Example_Mems/`): it exports resources from the MEMS sensor expansion board (temperature, humidity and accelerometer sensors).
3. NUCLEO-F401RE plus X-NUCLEO-IDS01A4 or X-NUCLEO-IDS01A5 plus X-NUCLEO-6180XA1 (see example in `/Utilities/Binary/LWM2M_to_IBM/Ipso_Example_Proximity/`): it exports resources from the FlightSense expansion board (proximity sensor).

These applications demonstrate how a node can connect to a local server using the OMA lightweight M2M (LWM2M) standard. This technology uses CoAP (over UDP) to publish a node resources in a standard format (as specified by LWM2M) and make them available to be accessed from a LWM2M server.

In these examples, the nodes connect to a local, hard-coded address that is assigned to the LWM2M to IBM gateway: `aaaa::ff:fe00:1` (this address is selected regardless of the actual MAC address associated to the interface).

This address can be changed at the top of the file `Projects/Multi/Applications/LWM2M_to_IBM/Src/lwm2m-simple-server.c` by modifying the following macro: `#define LWM2M_SERVER_ADDRESS_v6 "aaaa::ff:fe00:1"`.

It must be kept synchronized with the same macro that is used to set the LWM2M server IPV6 address in the node firmware.

2.5.4.1 Connecting the remote node to the IBM cloud

- Step 1.** Power the **STM32 Nucleo** board using a Mini-B USB cable connected to the PC.
- Step 2.** Program the firmware on the STM32 Nucleo board: you can copy (or drag and drop) the binary file (in the /Utility/Binary folder) to the USB mass storage that is automatically created when you connect the STM32 Nucleo board to your PC.
- Step 3.** Open a serial line monitor utility, select the serial port name to which the board is connected and configure it as follow:
 - Baud Rate = 115200
 - Parity = None
 - Bits = 8
 - Stopbits = 1
- Step 4.** Press the RESET (black) button on the STM32 Nucleo board and wait for the node to complete the registration.

Note: *The whole process is similar to the one of the Wi-Fi bridge application node firmware, with the only difference that the wireless sensor node points to a different address for the LWM2M server.*

The LWM2M address, client endpoint and location path (assigned by the server resource directory to the client) are shown below.

Figure 25. Client node registered on the local LWM2M server (gateway for IBM cloud)

```

COM40 - Tera Term VT
File Edit Setup Control Window KanjiCode Help
RD Client process started.
Looking for LWM2M server: 'aaaa::ff:fe00:1'
LWM2M Server Address:
aaaa::ff:fe00:1
RD Client started with endpoint '?ep=STL1-mems-FAFFDA4C25B3'
Registering with [aaaa::ff:fe00:1]:5683 lwm2m endpoint '?ep=STL1-mems-FAFFDA4C25B3' (binding mode U): '<1/0>,<3/0>,<3303/0>,<3304/0>,<3315/0>,<3313/0>,<3314/0>,<3200/0>,<3311/0>,'
Request block 0 (size 57): '<1/0>,<3/0>,<3303/0>,<3304/0>,<3315/0>,<3313/0>,<3314/0>,'
LWM2M reply to Register: 85
Request block 1 (size 18): '<3200/0>,<3311/0>,'
Successfully registered!
Location-path: /rd/011d0
No format given: Assume text plain...
No Accept header, using TLV by default...
Context: 3303/0/5700 found: 3
GET Called Path:3303/0/5700 Format:1541 ID:3303 bsize:64
  
```


3 System setup guide

3.1 Hardware description

This section describes the hardware components needed for developing a sensors based application. The following sub-sections describe the individual components.

3.1.1 STM32 Nucleo platform

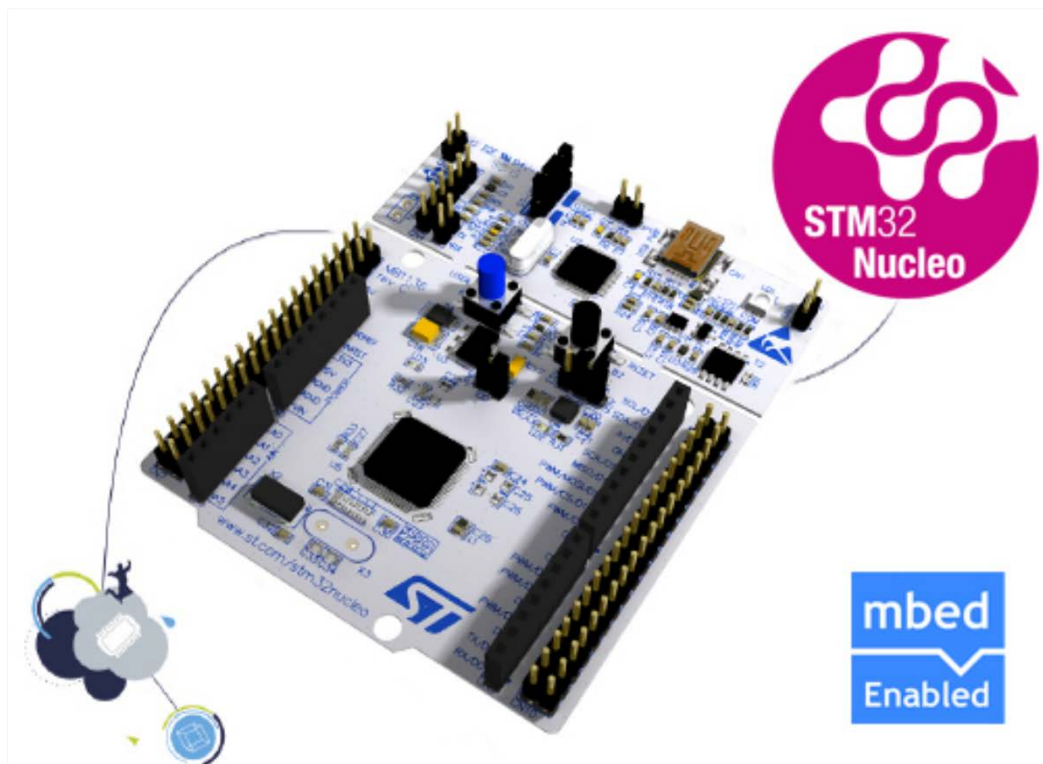
STM32 Nucleo development boards provide an affordable and flexible way for users to test solutions and build prototypes with any STM32 microcontroller line.

The Arduino™ connectivity support and ST morpho connectors make it easy to expand the functionality of the STM32 Nucleo open development platform with a wide range of specialized expansion boards to choose from.

The STM32 Nucleo board does not require separate probes as it integrates the ST-LINK/V2-1 debugger/programmer.

The STM32 Nucleo board comes with the comprehensive STM32 software HAL library together with various packaged software examples.

Figure 26. STM32 Nucleo board



Information regarding the STM32 Nucleo board is available at www.st.com/stm32nucleo

3.1.2 X-NUCLEO-IDS01A4 or X-NUCLEO-IDS01A5 expansion board

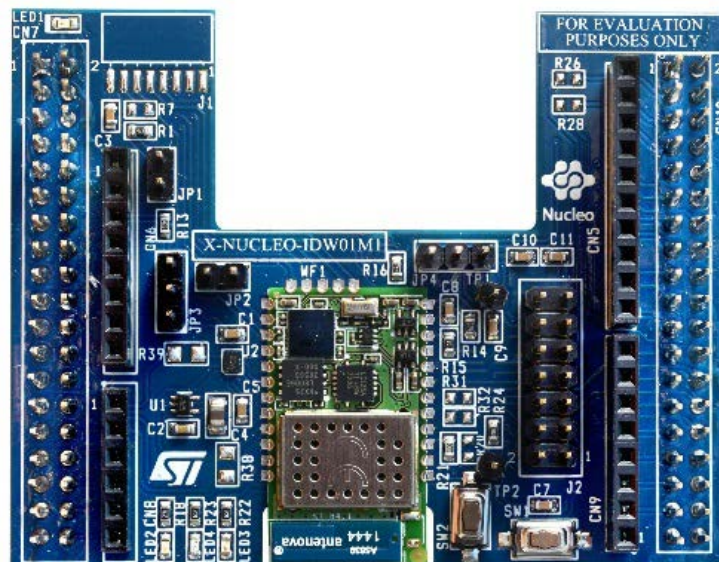
Figure 27. X-NUCLEO-IDS1A4 SPIRIT1 expansion board



Information about the X-NUCLEO-IDS01A4 and X-NUCLEO-IDS01A5 expansion board is available on www.st.com.

3.1.3 X-NUCLEO-IDW01M1 expansion board

Figure 28. X-NUCLEO-IDW01M1 Wi-Fi expansion board



Information about the X-NUCLEO-IDW01M1 expansion board is available on www.st.com.

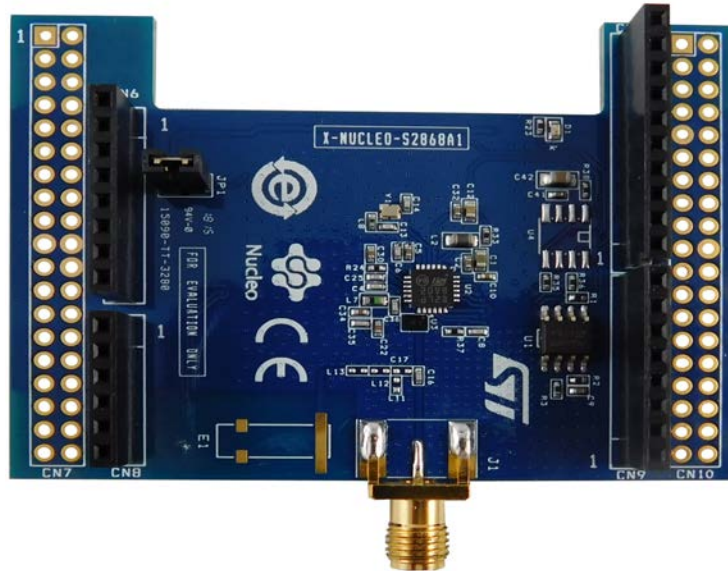
3.1.4 X-NUCLEO-S2868A1 expansion board

The X-NUCLEO-S2868A1 expansion board is based on the S2-LP radio and operates in the 868 MHz ISM frequency band.

The expansion board is compatible with ST morpho and Arduino UNO R3 connectors.

The X-NUCLEO-S2868A1 interfaces with the [STM32 Nucleo](#) microcontroller via SPI connections and GPIO pins. You can change some of the GPIOs by mounting or removing the resistors.

Figure 29. X-NUCLEO-S2868A1 expansion board



3.2 Software requirements

The following software components are required to set up a suitable development environment for creating applications for the [STM32 Nucleo](#) board with RF expansion board:

- [FP-NET-6LPWIFI1](#) software, available on www.st.com.
- Development tool-chain and Compiler: The [STM32Cube](#) expansion software supports the three following environments:
 - IAR Embedded Workbench for ARM® (EWARM) toolchain + ST-LINK
 - RealView Microcontroller Development Kit (MDK-ARM) toolchain + ST-LINK
 - System Workbench for STM32 (SW4STM32) + ST-LINK

3.3 Hardware and software setup

3.3.1 Hardware setup

For the implementation of the bridge, the following hardware components are required:

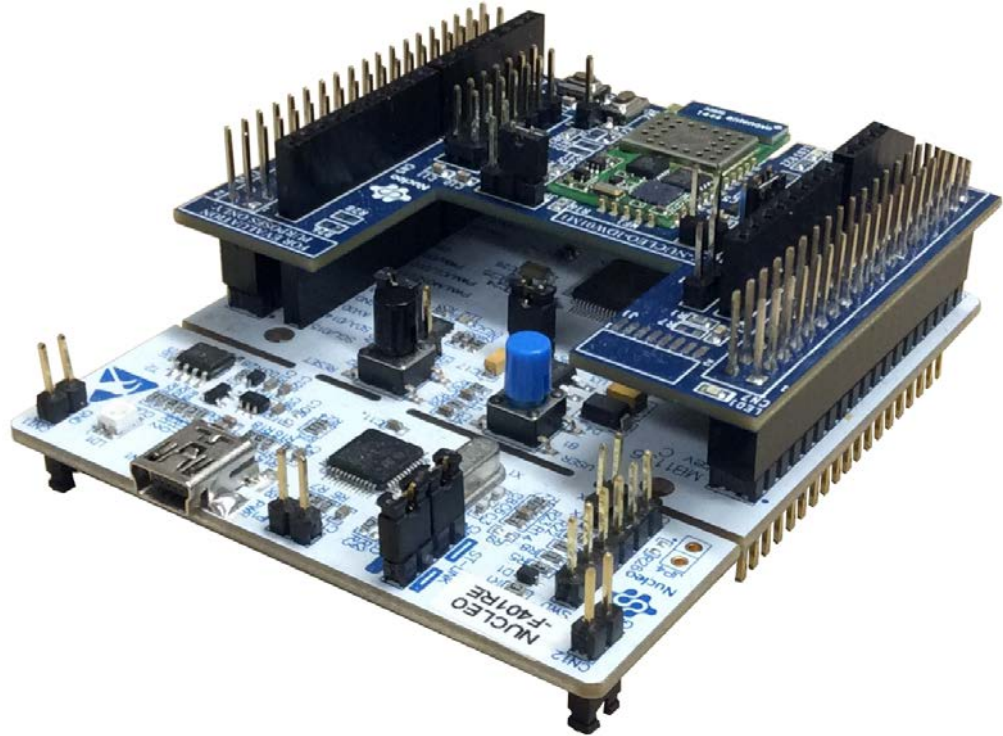
- One STM32 Nucleo development platform (order code: [NUCLEO-F401RE](#))
- One SPIRIT1 expansion board (order code: X-NUCLEO-IDS01A4 (for 868 MHz), or [X-NUCLEO-IDS01A5](#) (for 915MHz))
- One Wi-Fi expansion board (order code: X-NUCLEO-IDW01M1)
- One USB type A to Mini-B USB cable to connect the [STM32 Nucleo](#) to the PC

3.3.2 System setup guide

3.3.2.1 Bridge implementation based on STM32 Nucleo

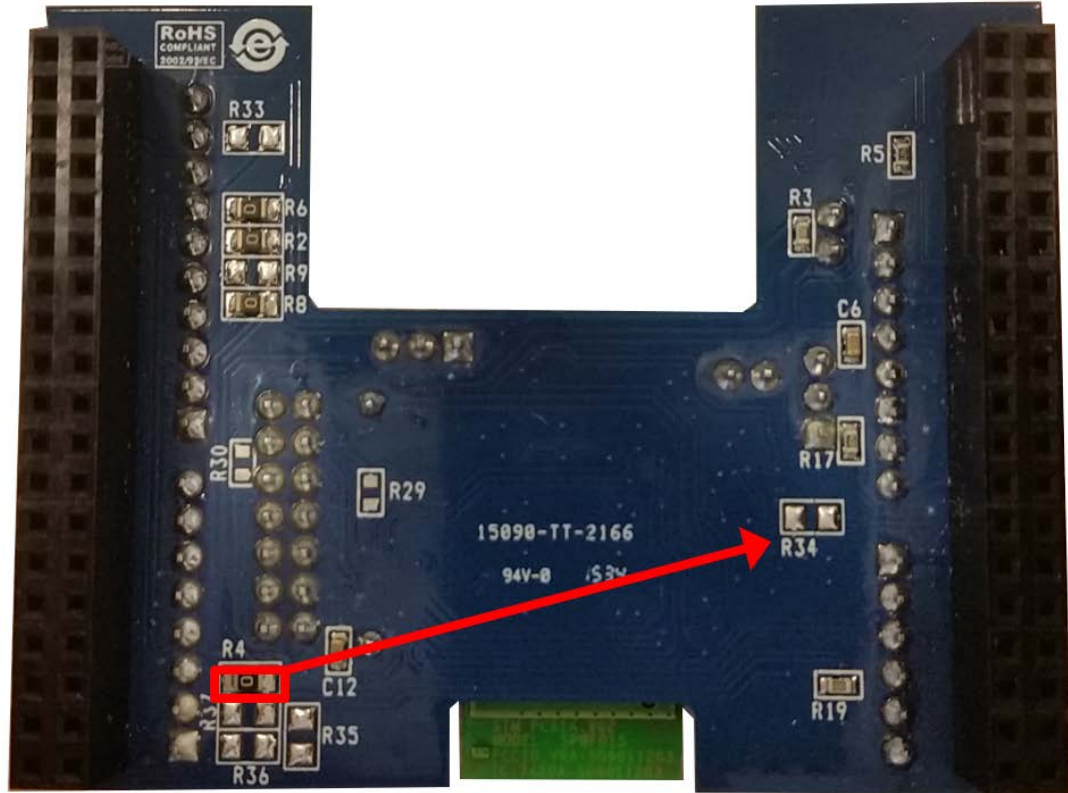
The [STM32 Nucleo](#) board integrates the ST-LINK/V2-1 debugger/programmer; you can download the ST-LINK/V2-1 USB driver by searching [STSW-LINK009](#) www.st.com.

The X-NUCLEO-IDW01M1 Wi-Fi expansion board is easily connected to the STM32 Nucleo board ST morpho connector, as shown below.

Figure 30. STM32 Nucleo plus the Wi-Fi expansion board


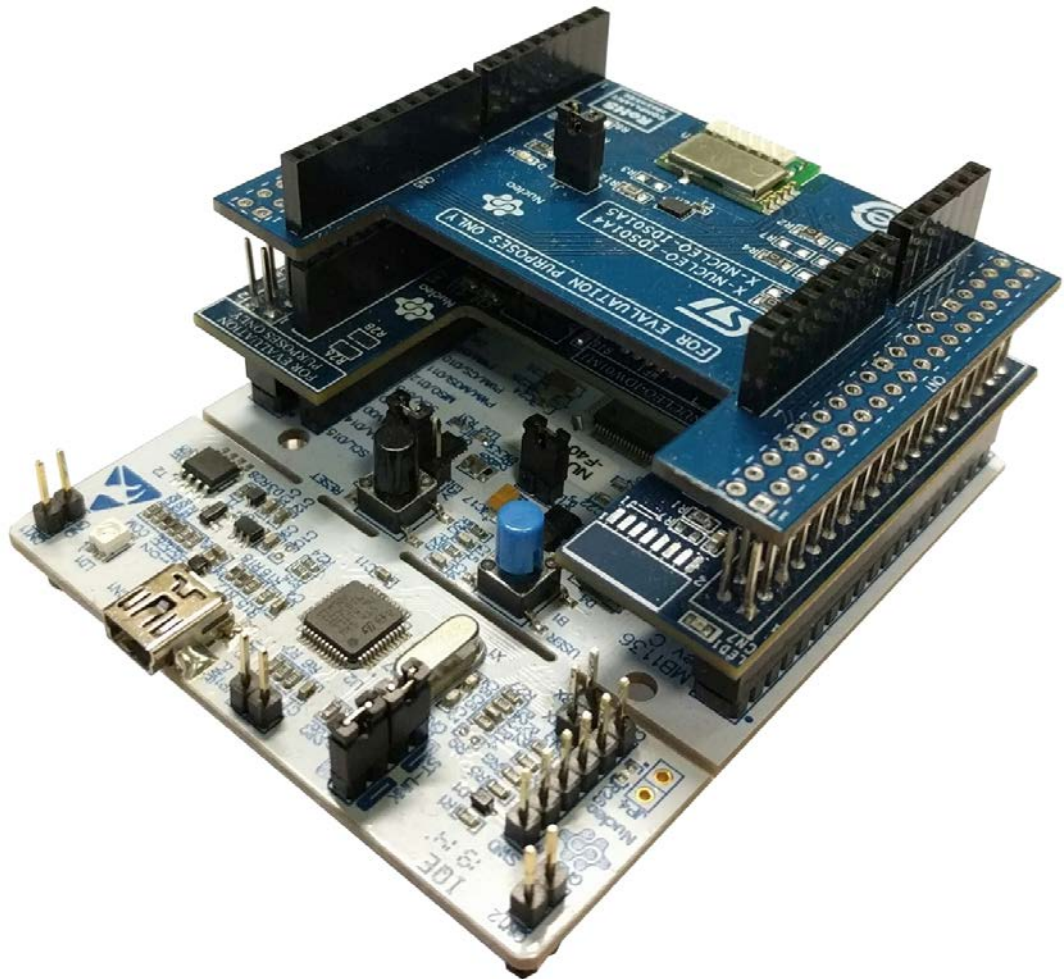
Note: To avoid a resource usage conflict between the X-NUCLEO-IDW01M1 expansion board and the X-NUCLEO-IDS1A4 or X-NUCLEO-IDS01A5 expansion board, the 0 Ω resistor at position R4 must be moved to R34, as shown below.

Figure 31. Modification required on the X-NUCLEO-IDW01M1 expansion board



The [SPIRIT1](#) expansion board X-NUCLEO-IDS01A4 or X-NUCLEO-IDS01A5 is easily connected to X-NUCLEO-IDW01M1 expansion board through the Arduino UNO R3 extension connector, as shown below.

Figure 32. STM32 Nucleo plus the Wi-Fi expansion board plus the SPIRIT1 expansion board

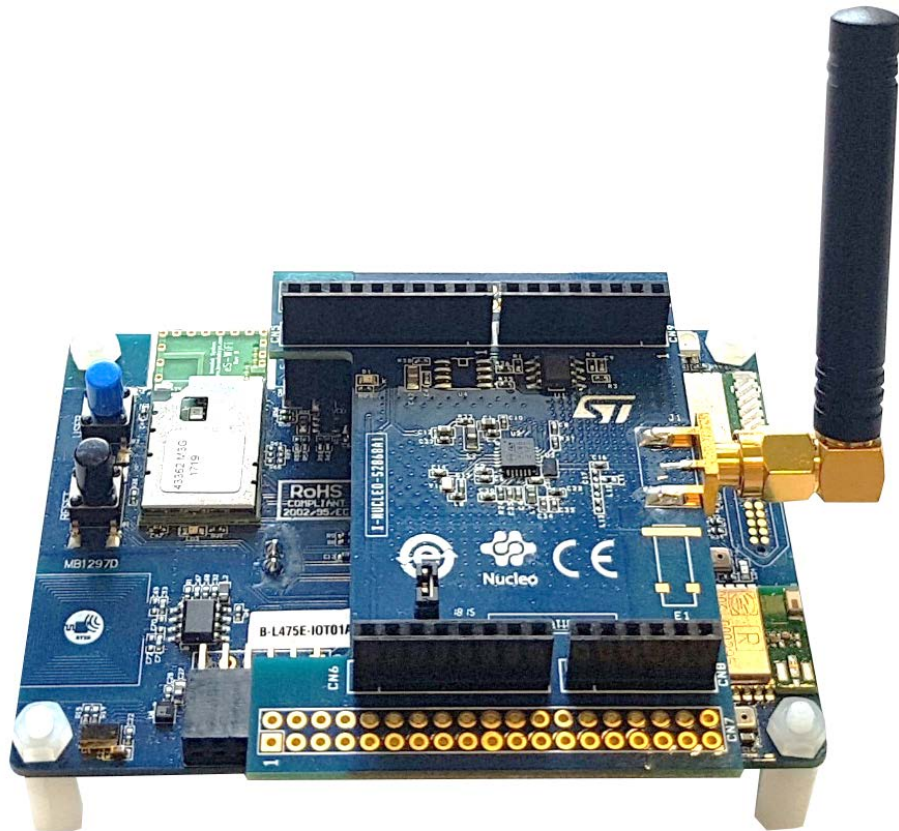


3.3.2.2 *Bridge implementation based on Discovery kit*

The Discovery kit IoT node (B-L475E-IOT01A) integrates the ST-LINK/V2-1 debugger/programmer; you can download the ST-LINK/V2-1 USB driver by searching [STSW-LINK009](http://www.st.com/STSW-LINK009) at www.st.com.

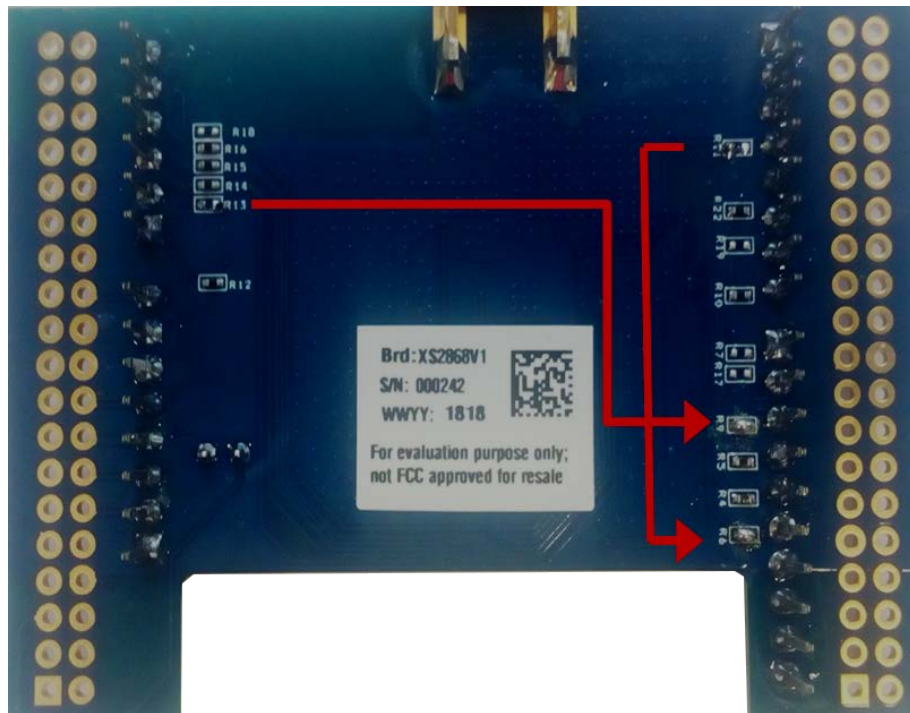
The [X-NUCLEO-S2868A1](#) sub-1GHz expansion board can be easily connect to the Discovery kit, as shown in the next figure.

Figure 33. Discovery kit plus the Sub-1GHz expansion board (X-NUCLEO-S2868A1)



Note: *A hardware modification is required on the X-NUCLEO-S2868A1 expansion board: in order to avoid a resource usage conflict between the X-NUCLEO-S2868A1 and the B-L475E-IOT01A board, the 0-Ohm resistances at position R11 and R13 must be unmounted and the 0-Ohm resistances must be mounted at positions R9 and R6, as shown below.*

Figure 34. Modification required on the X-NUCLEO-S2868A1 expansion board



Revision history

Table 2. Document revision history

Date	Version	Changes
18-Jun-2016	1	Initial release.
14-Apr-2017	2	Updated Section "Introduction", Section 1: "Acronyms and abbreviations", Section 2.1: "Overview ", Section 2.3: "Folder structure", and Section 2.5.2: "Wireless sensor node for Wi-Fi bridge application". Added Section 2.5.3: "LwM2M to IBM", Section 2.5.3.1: "Template", Section 2.5.3.2: "Sensor data visualization in IBM Quickstart", Section 2.5.3.3: "Connection to IBM Watson IoT cloud as registered device", Section 2.5.4: "Wireless sensor node for LWM2M to IBM application" and Section 2.5.4.1: "Connecting the remote node to the IBM cloud".
05-Sep-2018	3	Updated Section Introduction, Section 2.1 Overview, Section 2.2 Architecture. Added Section 2.5.3.2.1 Setup based on STM32 Nucleo, Section 2.5.3.2.2 Setup based on the Discovery kit IoT node (B-L475EIoT01A), Section 2.5.3.3.1 Setup based on STM32 Nucleo, Section 2.5.3.3.2 Setup based on the Discovery kit IoT node, Section 2.5.3.3.3 NodeRed, , Section 3.3.2.2 Bridge implementation based on Discovery kit.
11-Apr-2019	4	Updated Section 2.5.2.1 Connecting the remote node to the Internet .

Contents

1	Acronyms and abbreviations	2
2	FP-NET-6LPWIFI1 software expansion for STM32Cube	3
2.1	Overview	3
2.2	Architecture	3
2.3	Folder structure	4
2.4	APIs	5
2.5	Sample application description	5
2.5.1	Wi-Fi bridge	5
2.5.2	Wireless sensor node for Wi-Fi bridge application	7
2.5.3	LwM2M to IBM	10
2.5.4	Wireless sensor node for LWM2M to IBM application	21
3	System setup guide	23
3.1	Hardware description	23
3.1.1	STM32 Nucleo platform	23
3.1.2	X-NUCLEO-IDS01A4 or X-NUCLEO-IDS01A5 expansion board	23
3.1.3	X-NUCLEO-IDW01M1 expansion board	24
3.1.4	X-NUCLEO-S2868A1 expansion board	24
3.2	Software requirements	25
3.3	Hardware and software setup	25
3.3.1	Hardware setup	25
3.3.2	System setup guide	25
	Revision history	31

List of tables

Table 1.	List of acronyms	2
Table 2.	Document revision history	31

List of figures

Figure 1.	FP-NET-6LPWIFI1 software architecture	4
Figure 2.	FP-NET-6LPWIFI1 package folder structure	4
Figure 3.	Overall system architecture (example with bridge based on NUCLEO-F401RE)	5
Figure 4.	Wi-Fi bridge configuration	6
Figure 5.	Wi-Fi bridge custom credentials	6
Figure 6.	Wi-Fi bridge Contiki parameters	7
Figure 7.	Client node registered on the remote server	8
Figure 8.	Leshan server homepage	8
Figure 9.	Leshan server client homepage	9
Figure 10.	Leshan server, observing and reading resources.	10
Figure 11.	Template code for automatic sensor value retrieval	11
Figure 12.	Startup and Wi-Fi credential configuration (STM32 Nucleo)	12
Figure 13.	Startup and Wi-Fi credential configuration for Discovery kit.	13
Figure 14.	Startup and Wi-Fi credential configuration for Discovery kit (continued)	13
Figure 15.	Wireless node registration to the LWM2M-to-IBM gateway	14
Figure 16.	Sensors data in IBM Quickstart web page	14
Figure 17.	Configuration for registering the STM32 device on IBM Watson IoT cloud	15
Figure 18.	“LWM2M to IBM” setup with Discovery kit.	16
Figure 19.	“LWM2M to IBM” setup with Discovery kit (continued)	17
Figure 20.	Data sent to IBM cloud	17
Figure 21.	Dashboard example	18
Figure 22.	Node-RED program blocks.	19
Figure 23.	Sensor data visualization in the IBM web page for a registered device	20
Figure 24.	Button counter on the web-based dashboard	21
Figure 25.	Client node registered on the local LWM2M server (gateway for IBM cloud)	22
Figure 26.	STM32 Nucleo board	23
Figure 27.	X-NUCLEO-IDS1A4 SPIRIT1 expansion board.	24
Figure 28.	X-NUCLEO-IDW01M1 Wi-Fi expansion board	24
Figure 29.	X-NUCLEO-S2868A1 expansion board	25
Figure 30.	STM32 Nucleo plus the Wi-Fi expansion board.	26
Figure 31.	Modification required on the X-NUCLEO-IDW01M1 expansion board	27
Figure 32.	STM32 Nucleo plus the Wi-Fi expansion board plus the SPIRIT1 expansion board	28
Figure 33.	Discovery kit plus the Sub-1GHz expansion board (X-NUCLEO-S2868A1).	29
Figure 34.	Modification required on the X-NUCLEO-S2868A1 expansion board	30

IMPORTANT NOTICE – PLEASE READ CAREFULLY

STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST’s terms and conditions of sale in place at the time of order acknowledgement.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of Purchasers’ products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

ST and the ST logo are trademarks of ST. For additional information about ST trademarks, please refer to www.st.com/trademarks. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2019 STMicroelectronics – All rights reserved