

Contiki 6LoWPAN Quick Guide

Contiki on STM32 Nucleo plugged with Sub-1 GHz RF expansion board
(X-NUCLEO-IDS01A4, X-NUCLEO-IDS01A5)



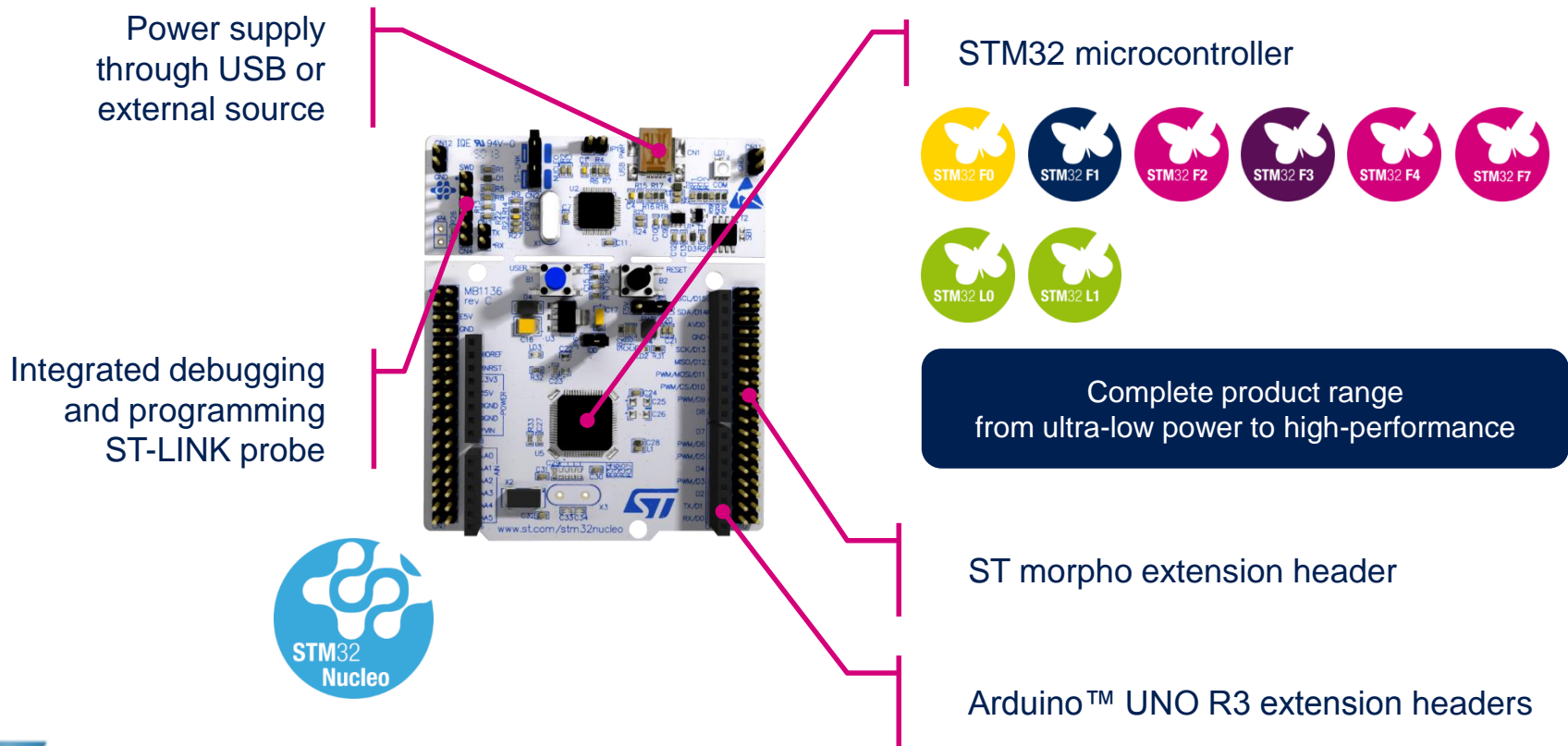
- Contiki (*) is an open source operating system (OS) for the Internet of Things (IoT)
- ST has developed a Contiki 3.x port for the STM32 Nucleo board (NUCLEO) plugged with the supported expansion boards (X-NUCLEO)
- The guide explains how to quickly get started with this platform

- The ST port allows running the Contiki OS, 6LoWPAN protocol stack and related applications on an STM32 Nucleo board plugged with one sub-1 GHz RF expansion board and, optionally, one motion MEMS and environmental sensors expansion board
- Software available for download on GitHub repository:
<https://github.com/STclab/contiki/tree/stm32nucleo-spirit1>
- Boards supported:
 - [NUCLEO-L152RE](#) based on the STM32L152RET6 ultra-low power microcontroller
 - [X-NUCLEO-IDS01A4](#) based on sub-1 GHz SPSGRF-868 SPIRIT1 module (operating at 868 MHz)
 - [X-NUCLEO-IDS01A5](#) based on sub-1 GHz SPSGRF-915 SPIRIT1 module (operating at 915 MHz)
 - [X-NUCLEO-IKS01A1](#) based on motion MEMS and environmental sensors (optional)
- License: BSD-3 (same as the Contiki distribution license)

STM32 Nucleo Development Boards (NUCLEO)

4

- A comprehensive range of affordable development boards for the entire STM32 microcontroller series, with unlimited unified expansion capabilities and integrated debugger/programmer functionality.



Description

- The X-NUCLEO-IDS01A4, X-NUCLEO-IDS01A5 are evaluation boards based on the SPIRIT1 RF modules SPSGRF-868 and SPSGRF-915
- The SPIRIT1 module communicates with the STM32 Nucleo board host microcontroller through an SPI link available on the Arduino UNO R3 connector.

Key products on board

SPSGRF

SPIRIT1 (Low data-rate, low-power sub-1GHz transceiver) module

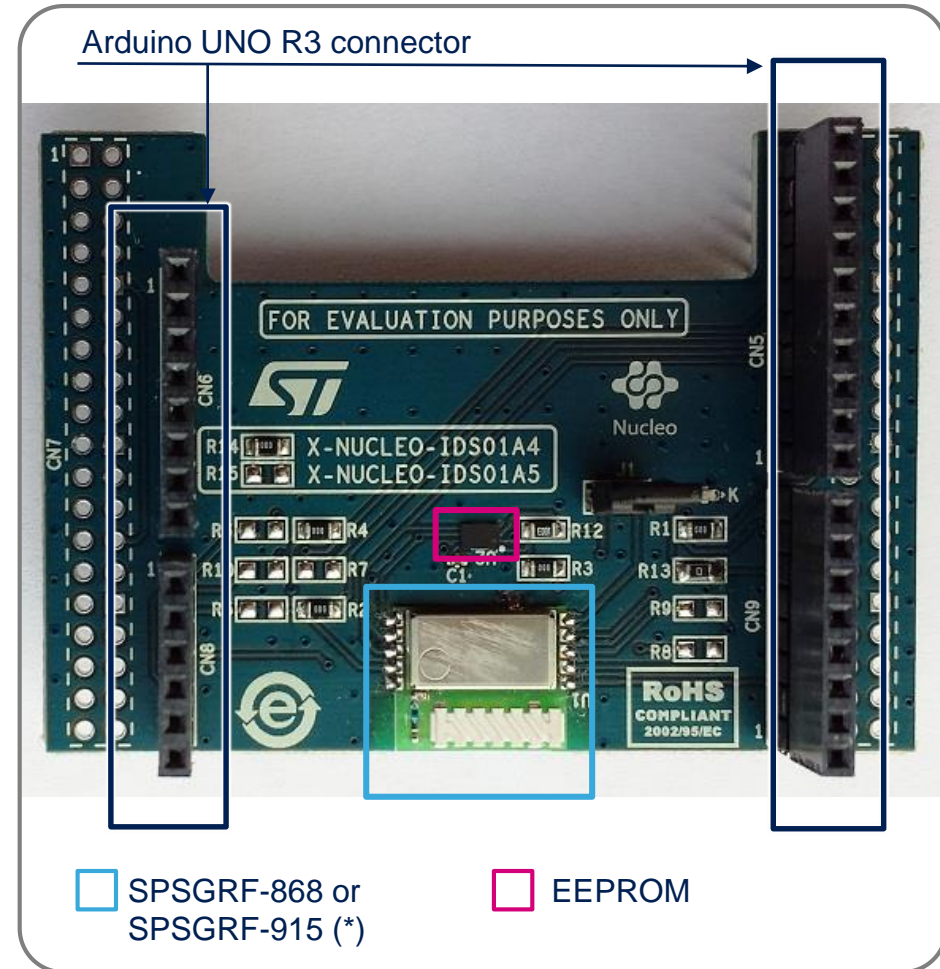
M95640-RMC6TG

64-Kbit serial SPI bus EEPROM

Latest info available at

[X-NUCLEO-IDS01A4](#)

[X-NUCLEO-IDS01A5](#)



Order code: **X-NUCLEO-IDS01A4, X-NUCLEO-IDS01A5**

(*) Identification of the operating frequency of the X-NUCLEO-IDS01Ax (x=4 or 5) is performed through two resistors (R14 and R15).

Motion MEMS and environmental sensor expansion board

Overview

6

Description

- The X-NUCLEO-IKS01A1 is a motion MEMS and environmental sensor evaluation board.
- It is compatible with the Arduino UNO R3 connector layout, and is designed around ST's sensors.

Key products on board

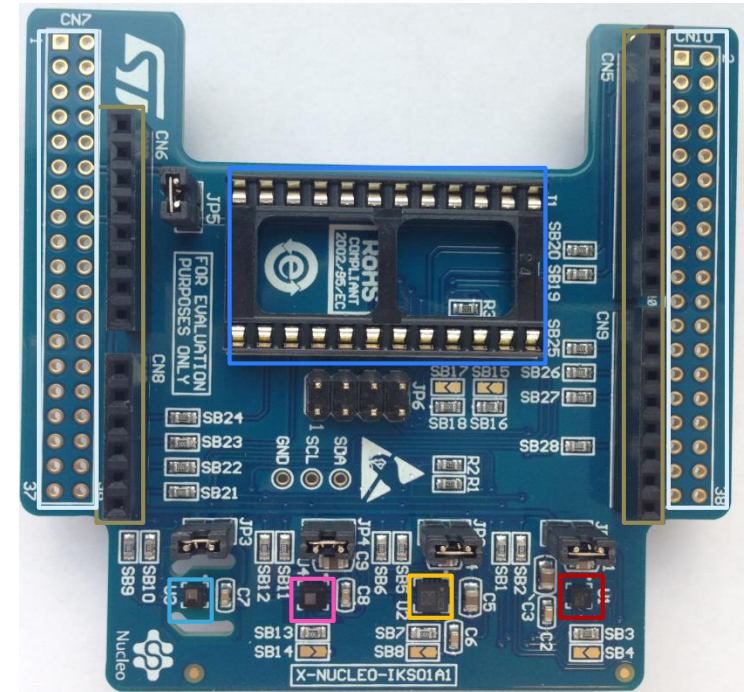
LSM6DS0: MEMS 3D accelerometer ($\pm 2/\pm 4/\pm 8$ g) + 3D gyroscope ($\pm 245/\pm 500/\pm 2000$ dps)

LIS3MDL: MEMS 3D magnetometer ($\pm 4/\pm 8/\pm 12/16$ gauss)

LPS25HB: MEMS pressure sensor, 260-1260 hPa absolute digital output barometer

HTS221: Capacitive digital relative humidity and temperature

DIL 24-pin: Socket available for additional MEMS adapters and other sensors (UV index)



- | | | |
|---|--|--|
|  HTS221 |  LSM6DS0 |  ST morpho connector** |
|  LPS25HB |  LIS3MDL |  Arduino UNO R3 connector |
| |  DIL 24-pin | |

Order code: X-NUCLEO-IKS01A1

** Connector for the STM32 Nucleo Board

Latest info available at
[X-NUCLEO-IKS01A1](https://www.st.com/en/evaluation-tools/x-nucleo-iks01a1.html)

Setup & demo examples

Hardware prerequisites

7

- 1 x NUCLEO-L152RE (STM32 Nucleo board)
- 1 x X-NUCLEO-IDS01A4 (Sub-1 GHz RF expansion board based on the SPSGRF-868 module) or 1 x X-NUCLEO-IDS01A5 (Sub-1 GHz RF expansion board based on the SPSGRF-915 module)
- (OPTIONAL) 1 x X-NUCLEO-IKS01A1 (Motion MEMS and environmental sensor expansion board)
- Laptop/PC with Windows 8/7 or Linux Ubuntu 15.4
- 1 x USB type A to Mini-B USB cable



NUCLEO-L152RE



X-NUCLEO-IDS01A4 or
X-NUCLEO-IDS01A5



Mini USB



X-NUCLEO-IKS01A1
(OPTIONAL)

Setup & demo examples

Software prerequisites (1/2)

8

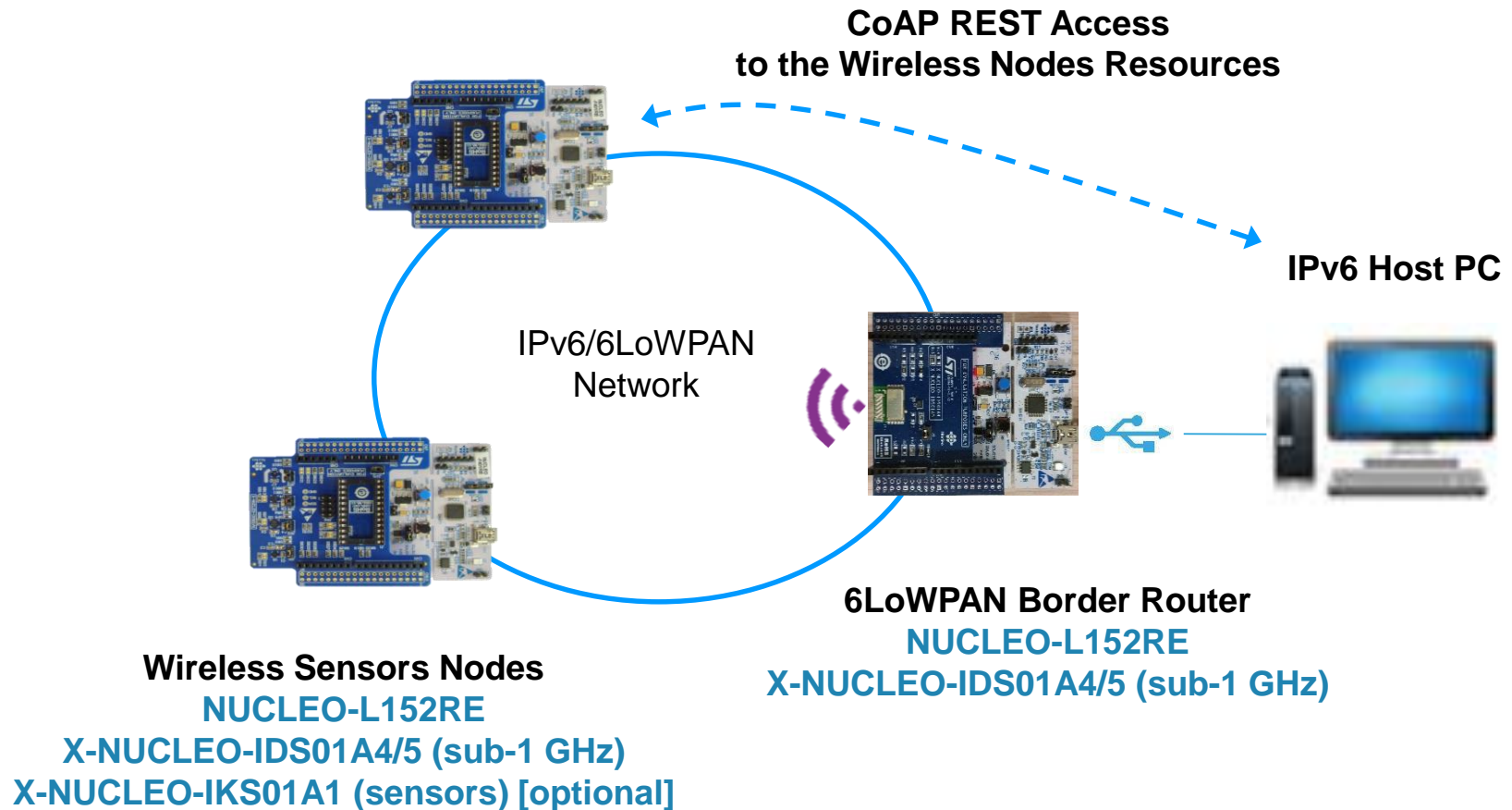
- Download the ST port from GitHub repository
(<https://github.com/STclab/contiki/tree/stm32nucleo-spirit1>)
- The ST port is installed automatically when the Contiki and sub-module repositories are cloned. It can be done using the following commands:
 - `git clone https://github.com/STclab/contiki.git`
`cd contiki/`
`git checkout stm32nucleo-spirit1`
`git submodule init`
`git submodule update`
- Contiki Platform name for ST port: stm32nucleo-spirit1

Setup & demo examples

Software prerequisites (2/2)

9

- PC software
 - Windows PC:
 - Linux environment on Windows using Cygwin ([Link](#))
 - GCC is provided in the System Workbench for STM32 (SW4STM32) ([Link](#))
 - Git package for Cygwin or Git for Windows ([Link](#))
 - WinPcaP (for demo purpose) ([Link](#))
 - Linux PC:
 - GNU Tools for ARM Embedded Processors ([Link](#))
- Firefox web browser ([Link](#))
- Firefox Copper plug-in (only for CoAP demo purpose) ([Link](#))



Contiki on STM32 Nucleo in a few steps (1/2)

11

1

Clone the online repository

```
git clone https://github.com/STclab/contiki.git
cd contiki/
git checkout stm32nucleo-spirit1
git submodule init
git submodule update
```

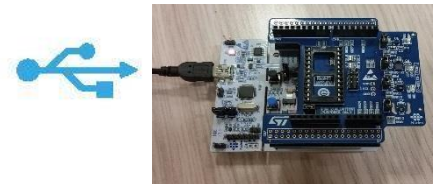
2

Compile the FW for a wireless node: REST example
(using the standard Contiki provided “er-rest-example”)

```
cd examples/er-rest-example
make TARGET=stm32nucleo-spirit1 USE_SUBGHZ_BOARD=IDS01A5
arm-none-eabi-objcopy -O binary er-example-server.stm32nucleo-
spirit1 er-example-server.bin
```

3

Connect the wireless sensor board to a
PC USB slot and program the device



Copy the “er-example-server.bin” file
(e.g. drag & drop) to the USB mass storage
corresponding to the STM32 Nucleo board

Contiki on STM32 Nucleo in a few steps (2/2)

12

4

Compile the FW for the Border Router node

```
cd examples/ipv6/rpl-border-router  
make TARGET=stm32nucleo-spirit1 USE_SUBGHZ_BOARD=IDS01A5  
arm-none-eabi-objcopy -O binary border-router.stm32nucleo-spirit1 br.bin
```

5

Connect the board to USB and
program the device



copy the “br.bin” file
(e.g. drag & drop) to the USB mass storage
corresponding to the STM32 Nucleo board

6

Setup the IPv6 Host PC
for IP traffic bridging between
host and 6LowPAN border Router

Contiki on STM32 Nucleo in a few steps

13

Windows PC setup (Win 7/8)
using "wpcapslip6" utility



Linux PC setup (Ubuntu)
using "tunslip6" utility

OR

1. wpcapslip6 needs a working network adapter:
The Microsoft loopback adapter can be installed via "Add legacy hardware" in the Windows Device Manager (reboot is needed after installation of the loopback adapter)
2. Copy "cygwin1.dll" from "contiki/tools/cygwin" to wpcapslip6 folder
3. Install WinPcap
4. run Cygwin as administrator

```
cd ./tools
cc tunslip6.c -o tunslip6
sudo ./tunslip6 -s /dev/ttyACM0 aaaa::1/64
```

wpcapslip6 utility can then be used with the rpl-border-router example

```
cd ./tools/stm32w/wpcapslip6
./wpcapslip6 -s /dev/ttyS21 -b aaaa:: -a aaaa::1/128 [addr]
```

Where [addr] is the MAC address of the local net adapter

```
~/workspace/contiki-stm32nucleo-spirit1/tools/stm32w/wpcapslip6
$ ./wpcapslip6.exe -s /dev/ttyS21 -b aaaa:: -a aaaa::1/128 02-00-4C-4F-4F-50
Using local network interface: Local Area Connection 5
10:10:56 netsh interface ipv6 add address "Local Area Connection 5" aaaa::1/128
10:10:58 wpcapslip6 started on "/dev/ttyS21"
10:10:58 Got request message of type M
10:10:58 *** Gateway's MAC address: 08-00-f7-ff-bd-42
10:10:58 Fictitious MAC-48: 0A-00-F7-BD-48-42
10:10:58 netsh interface ipv6 add route aaaa::/64 "Local Area Connection 5" aaaa::a00:f7ff:b7bd:4842
Ok.
10:10:58 netsh interface ipv6 add neighbor "Local Area Connection 5" aaaa::a00:f7ff:b7bd:4842 "0A-00-F7-BD-48-42"
10:10:58 Got configuration message of type O
10:10:58 *** Address:aaaa:: => aaaa:0000:0000:0000
10:10:58 Got configuration message of type P
10:10:58 Setting prefix aaaa::
10:10:59 Server IPv6 addresses:
10:10:59 aaaa:a00:f7ff:b7bd:4842
10:10:59 fc00:a00:f7ff:b7bd:4842
10:10:59 fe80::a00:f7ff:b7bd:4842
```

wpcapslip6 terminal window output

```
*****SLIP started on "/dev/ttyACM0"
opened tun device "/dev/tun0"
ifconfig tun0 inet 'hostname' up
ifconfig tun0 add aaaa::1/64
ifconfig tun0 inet 172.16.0.1 pointopoint 172.16.0.2
ifconfig tun0 add fe80::0:0:0:1/64
ifconfig tun0

tun0      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
          inet addr:172.16.0.1  P-t-P:172.16.0.2  Mask:255.255.255.255
          inet6 addr: fe80::1/64 Scope:Link
          inet6 addr: aaaa::1/64 Scope:Global
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

*** Address:aaaa::1 => aaaa:0000:0000:0000
Got configuration message of type P
Setting prefix aaaa::
Server IPv6 addresses:
aaaa::800:f5ff:eb3a:14c5
fc00::800:f5ff:eb3a:14c5
fe80::800:f5ff:eb3a:14c5
```

Tunslip6 terminal window output

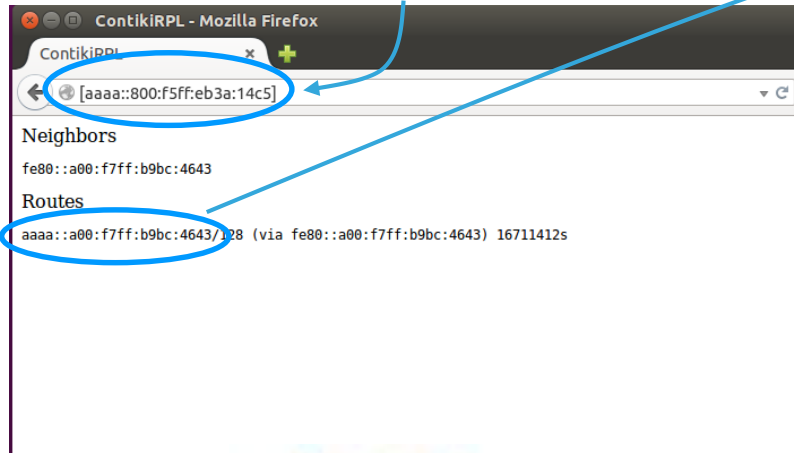
Contiki server address (used in the next step)

Contiki on STM32 Nucleo in a few steps

14

- 7 Open a Web browser (Firefox) to access the Contiki server providing the RPL neighbors and routes information.

Contiki server address (see previous step)
between brackets, e.g. [aaaa::800:f5ff:eb3a:14c5]



8

Ping the wireless
Node to test the
6LoWPAN
connectivity

`ping6 aaaa::a00:f7ff:b9bc:4643`

```
PING aaaa::a00:f7ff:b9bc:4643(aaaa::a00:f7ff:b9bc:4643) 56 data bytes
64 bytes from aaaa::a00:f7ff:b9bc:4643: icmp_seq=1 ttl=63 time=70.0 ms
64 bytes from aaaa::a00:f7ff:b9bc:4643: icmp_seq=2 ttl=63 time=70.7 ms
64 bytes from aaaa::a00:f7ff:b9bc:4643: icmp_seq=3 ttl=63 time=76.8 ms
64 bytes from aaaa::a00:f7ff:b9bc:4643: icmp_seq=4 ttl=63 time=65.8 ms
64 bytes from aaaa::a00:f7ff:b9bc:4643: icmp_seq=5 ttl=63 time=72.8 ms
64 bytes from aaaa::a00:f7ff:b9bc:4643: icmp_seq=6 ttl=63 time=67.8 ms
64 bytes from aaaa::a00:f7ff:b9bc:4643: icmp_seq=7 ttl=63 time=74.8 ms
64 bytes from aaaa::a00:f7ff:b9bc:4643: icmp_seq=8 ttl=63 time=68.9 ms
64 bytes from aaaa::a00:f7ff:b9bc:4643: icmp_seq=9 ttl=63 time=75.9 ms
64 bytes from aaaa::a00:f7ff:b9bc:4643: icmp_seq=10 ttl=63 time=64.9 ms
64 bytes from aaaa::a00:f7ff:b9bc:4643: icmp_seq=11 ttl=63 time=65.9 ms
64 bytes from aaaa::a00:f7ff:b9bc:4643: icmp_seq=12 ttl=63 time=72.9 ms
64 bytes from aaaa::a00:f7ff:b9bc:4643: icmp_seq=13 ttl=63 time=67.8 ms
64 bytes from aaaa::a00:f7ff:b9bc:4643: icmp_seq=14 ttl=63 time=74.8 ms
64 bytes from aaaa::a00:f7ff:b9bc:4643: icmp_seq=15 ttl=63 time=69.8 ms
64 bytes from aaaa::a00:f7ff:b9bc:4643: icmp_seq=16 ttl=63 time=70.8 ms
^C
--- aaaa::a00:f7ff:b9bc:4643 ping statistics ---
16 packets transmitted, 16 received, 0% packet loss, time 15017ms
rtt min/avg/max/mdev = 64.936/70.685/76.827/3.620 ms
fabien@marco-linux-HP:~$
```

9

Install the “Copper” CoAP plugin for Firefox
<https://addons.mozilla.org/en-US/firefox/addon/copper-270430>

Then access the CoAP Server on the wireless
node by typing the URL with the node IP address

`coap://[aaaa::a00:f7ff:b9bc:4643]:5683/`

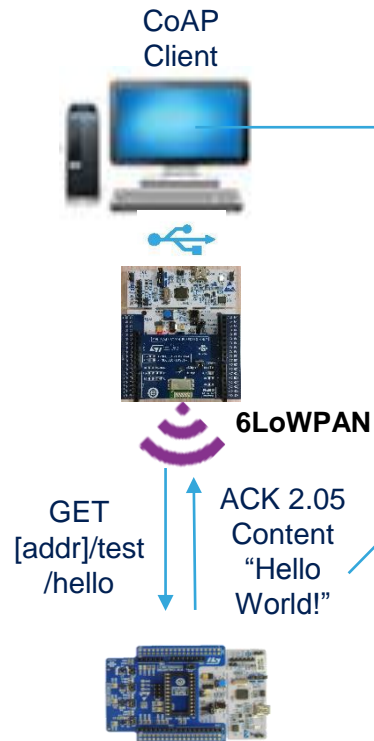
Contiki on STM32 Nucleo in a few steps

Example: “Hello World!” Resource Access using CoAP

15

(1) CoAP Resource Discovery

(2) CoAP GET Access to the “test/hello” resource



The screenshot shows a Firefox Browser window on a Linux PC with the “Copper” plugin (CoAP client). The address bar displays the CoAP URI: `coap://[aaaa::a00:f7ff:b9bc:4643]:5683/test/hello`. The browser interface includes a search bar, a “Discover” button, and a “Ping” button. The main content area shows the CoAP response: **2.05 Content (Blockwise) (Download finished)**. The response details are as follows:

H...	Value	Option	Value	Info
Type	Acknowledgment	ETag	0x0C	1 byte
Code	2.05 Content	Content-Format	text/plain	0
Mess...	26449	Block2	0 (32 B/block)	1 byte
Token	empty			

The payload (12) is displayed as “Hello World!”.

On the right side of the browser window, the “Debug Control” panel is visible, showing various request options and response options. The “Request Options” section includes fields for Token, Accept, Content-Format, Block1 (Req.), Block2 (Res.), Auto, Size1, Size2, Observe, ETag, If-Match, and Proxy-Uri. The “Response Options” section includes fields for Max-Age, Location-Path, Location-Query, and Custom Options.

- This demo requires an X-NUCLEO-IKS01A1 expansion board for STM32 Nucleo to be mounted on a wireless node
 - The X-NUCLEO-IKS01A1 should be plugged on top of X-NUCLEO-IDS01A4/5 and NUCLEO-L152RE
- To get the demo running, a modified version of the standard Contiki “er-rest-example” application needs to be used
 - The modification is needed to update the names of the sensors used in the “er-rest-example” application and match the names of the X-NUCLEO-IKS01A1 sensors
 - The modified application is available for download from the following GitHub repository: <https://github.com/STclab/stm32nucleo-spirit1-examples>
 - The step-by-step setup is identical to the one described in the previous “Hello World” demo, except for “step 2” in which the modified “sensor-er-rest-example” is used
- The next slide shows the result of a CoAP GET access to the “temperature” resource hosted by the CoAP server on the wireless node

Sensors Resource Access using CoAP

Example of temperature sensor reading

17

CoAP GET Access to the resource: "sensors/temperature"

CoAP Client



6LoWPAN

GET [addr]/sensors/temperature

ACK 2.05 Content "27.7"



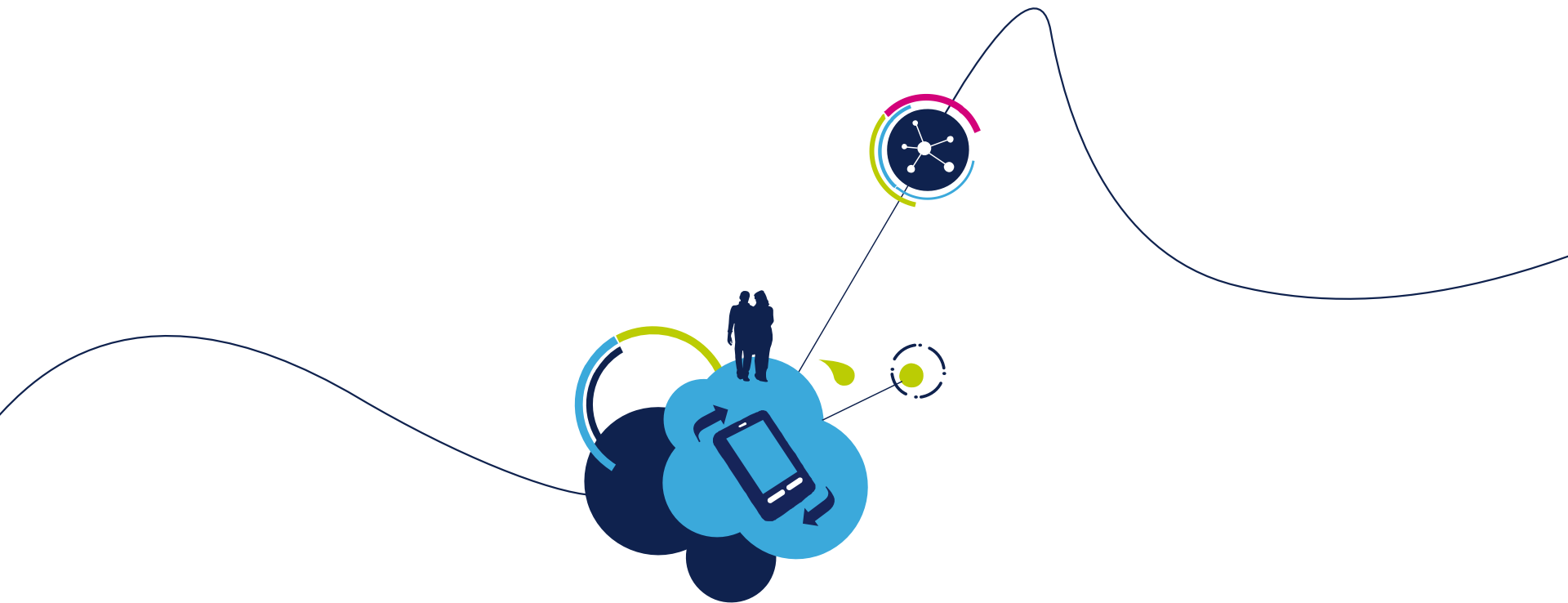
CoAP Server

The screenshot shows a CoAP client interface with the URL `coap://[aaaa::a00:f7ff:104e:efb1]:5683/sensors/temperature`. The interface includes a toolbar with various HTTP methods (Discover, Ping, GET, POST, PUT, DELETE, Observe) and a 'Payload' section. The main content area displays the response from the server, which is a 2.05 Content (Blockwise) message. The response details include:

Header	Value	Option	Value	Info
Type	Acknowledgment			
Code	2.05 Content			
Message	18306			
Token	empty			

The payload (4) is displayed as `27.7`. The right sidebar contains various configuration options for the CoAP client, including 'Debug Control', 'Token', 'Request Options', 'Content-Format', 'Observe', 'Etag', 'If-Match', 'Uri-Host', 'Uri-Port', 'Proxy-Uri', 'Response Options', and 'Custom Options'.

(*) Use of the X-NUCLEO-IKS01A1 sensors expansion board is required for this demo



www.st.com