# Wifi Training - Hands On

SPG Application

life.augmented

# Lab Prerequisites



- Hardware
  - STEVAL-IDW001V1 + STEVAL-PCC018V1, evaluation board for SPWF01SA.11
- Utility software
  - CP210x USB to UART Bridge VCP Drivers (available from http://www.silabs.com/products/mcu/Pages/USBtoUARTBridgeVCPDrivers.aspx)
  - Tera Term: terminal emulator (available from http://en.sourceforge.jp/projects/ttssh2/releases)
  - Text Editor
  - Web browser

# Hands on Agenda



- Lab 1: HW and SW setup
- Lab 2: Set the SPWF variables
- Lab 2a: FOTA update
- Lab 3: Access point connection (WPA or WEP)
- Lab 4: Web client mode
- Lab 5: Web server mode – files in RAM
- Lab 6: Web server mode – files in FLASH
- Lab 7: Socket interface
- Lab 8: MiniAP mode for the first set
- Lab 9: Socket interface in MiniAP mode
- Lab 10: Web server in MiniAP mode
- Lab 11: Remote control of GPIO interfaces
- Lab 12: Input demo
- Lab 13: Socket Server
- Lab 14: Low Power Modes
- Lab 15: HW switch from STA to MiniAP
- Lab 16: IBSS mode

# Lab 1: HW and SW setup



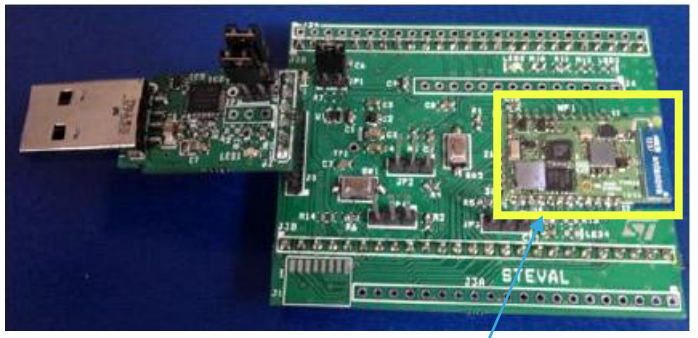
- Objective
  - Hardware set-up
  - Software set-up

- Prerequisites
  - Work alone

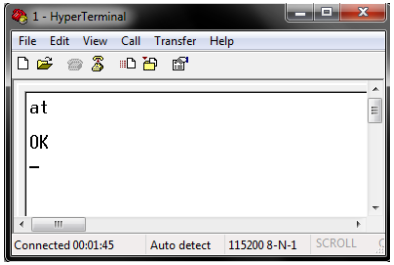
# Lab 1: EVAL of the SPWF01SA.11 module



- Plug&Play Solution
- AT command set
- Power Supplied via the USB interface
- UART/USB bridge from Silicon Lab requires to install the correspondent driver on your PC
- SMD antenna and reset button on-board




*HyperTerminal or similar*




AT Commands/Events

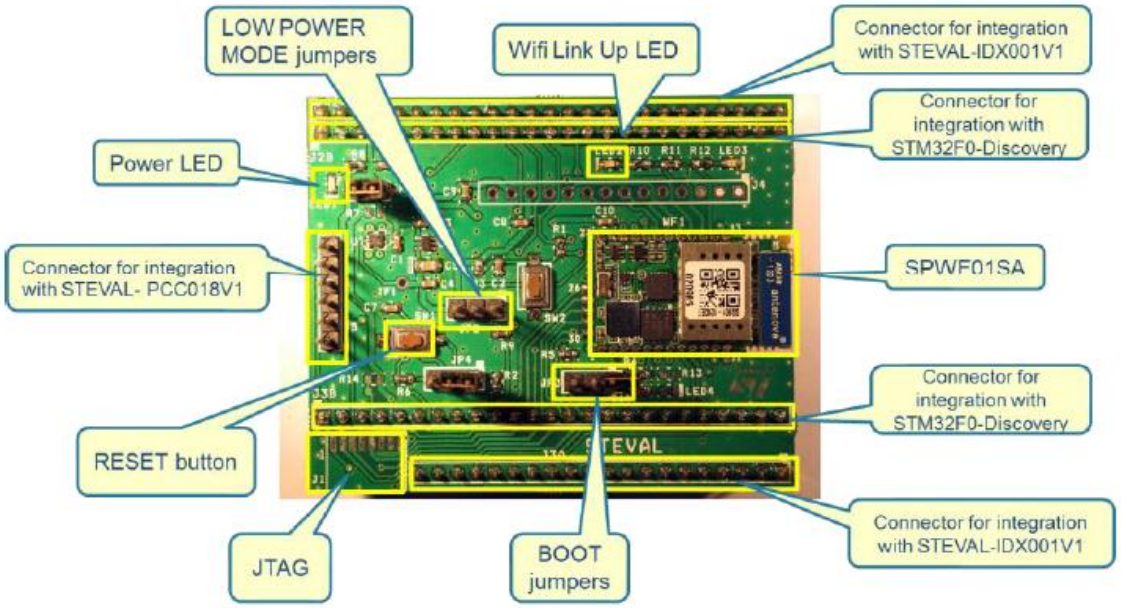
**VCOM**

802.11
b/g/n

# Lab 1: STEVAL-IDW001V1



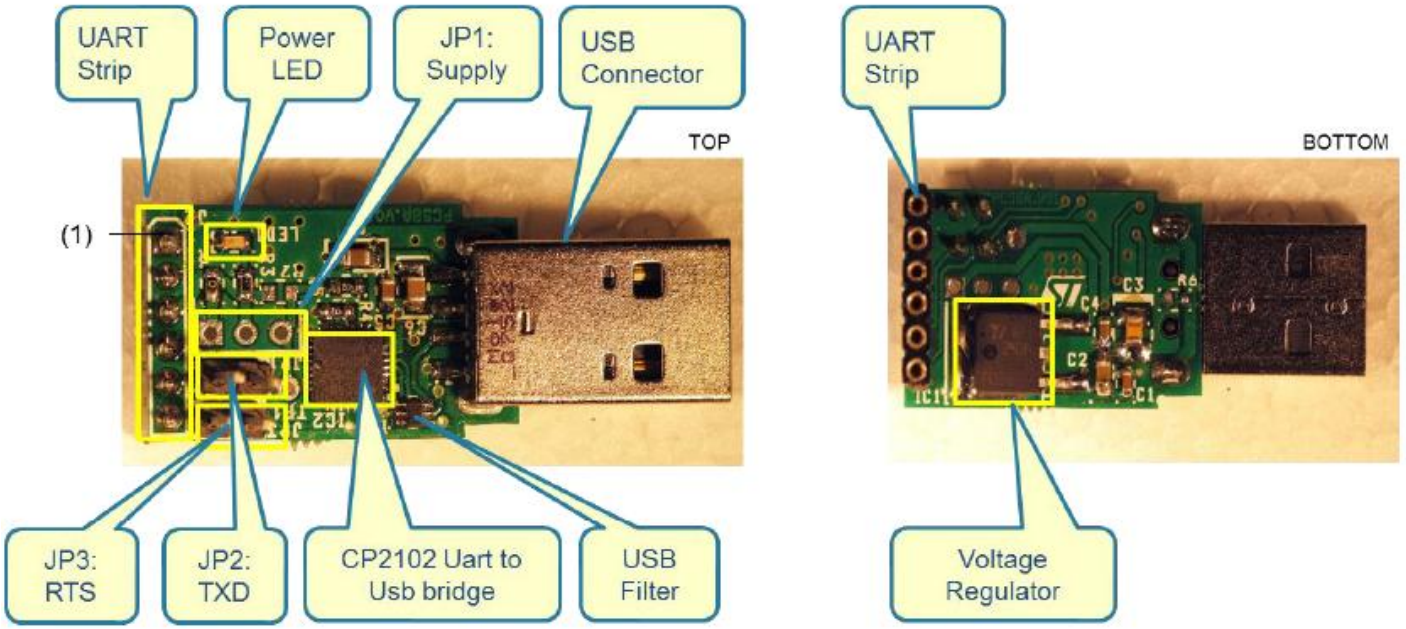
STEVAL-IDW001V1, evaluation board of the WiFi module SPWF01SA.11



# Lab 1: STEVAL-PCC018V1



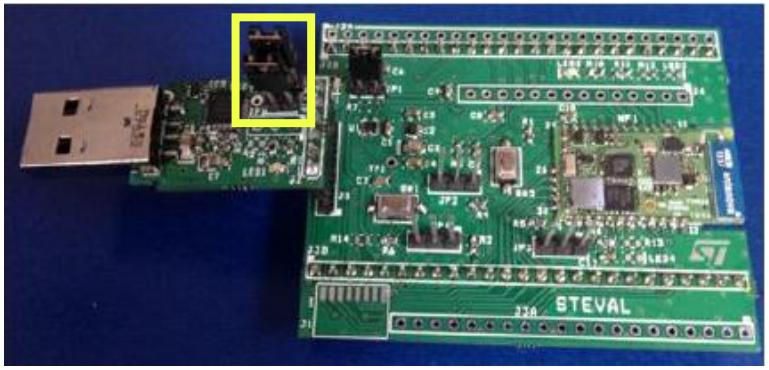
STEVAL-PCC018V1, USB to UART board



# Lab 1: Hardware setup



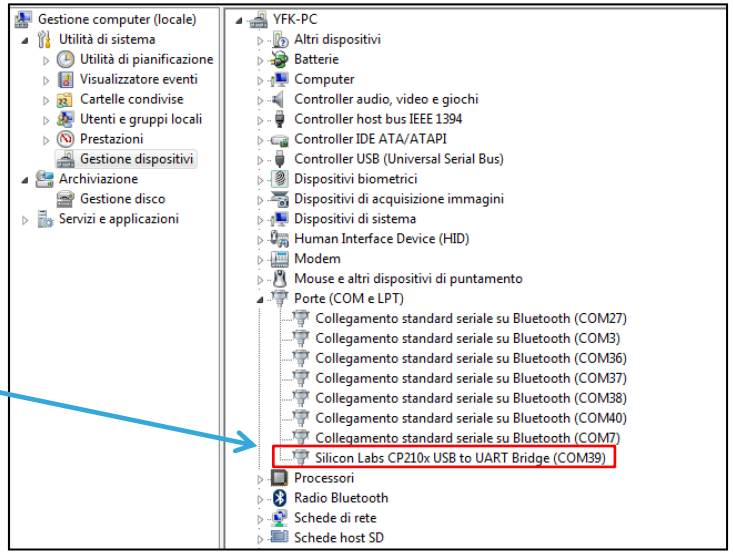
- Insert the jumpers in the USB to UART board as in the figure




- Connect the Evaluation board to the PC

- Use the Device manager to find the assigned COM port

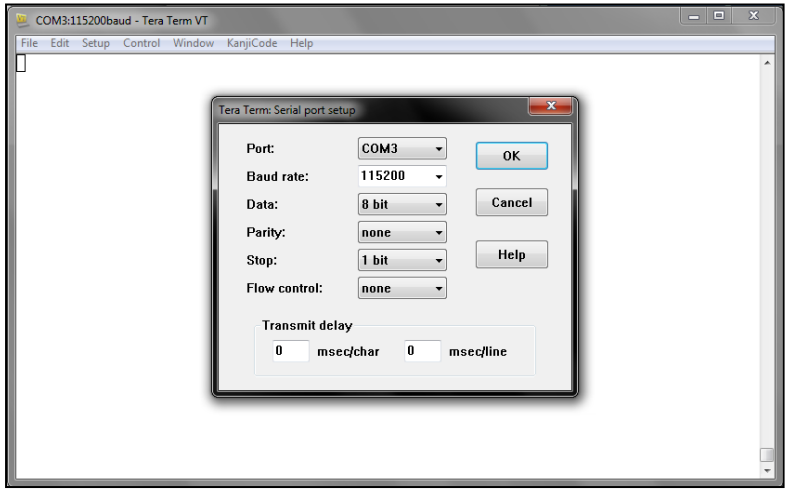
- ***Module automatically performs a reset***
- ***Module automatically performs a scan for available networks (if wifi_mode variable is ≠ 0)***
- ***Module automatically enters in command mode***
- ***STEVAL-PCC018V1: Yellow power LED will light up***
- ***STEVAL-IDW001V1: Yellow power LED will light up, orange wifi link up LED will light up***



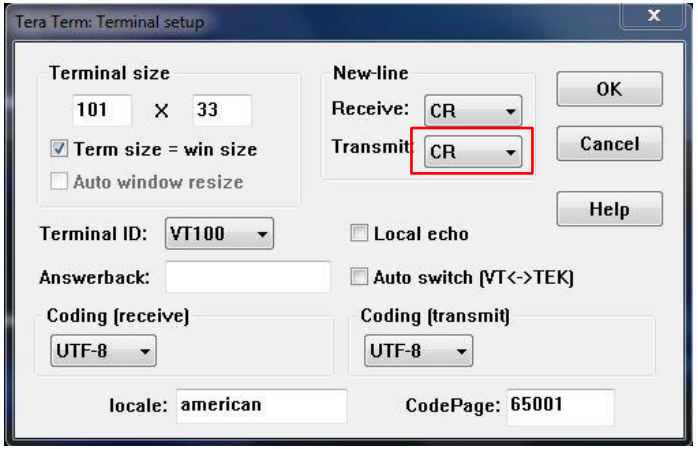
# Lab 1: Configuring the UART



- Open Tera Term
  - Run Tera Term (open Tera term folder and then run **ttermpro.exe**)
  - Open the assigned COM port
  - Tera Term: Setup → Serial port
    - Baud rate: 115200
    - Data: 8 bit
    - Parity: none
    - Stop: 1 bit
    - Flow control: none




- Set CR in the Terminal setup (default config)
  - Tera Term: Setup → Terminal → Transmit: CR

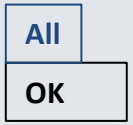


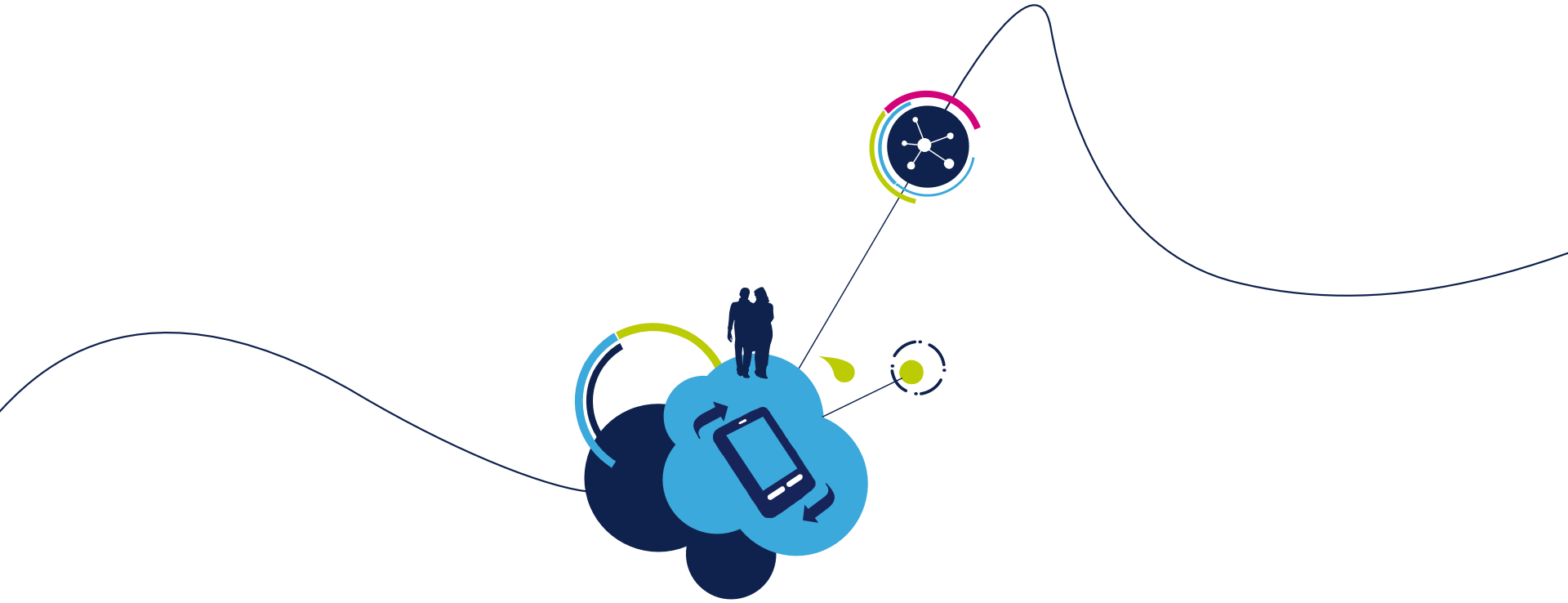
# Lab 1: Configuring the UART



- Open Tera Term
- Command Mode
  - Type **AT** followed by a carriage return (CR)

*Tera Term output*





# You are ready to use your Wifi EVAL board!

life.augmented

# Lab 2: Set the SPWF variables



- Objective
  - Run a command
  - Get the default configuration dump
  - Set host name
  - Set static IP parameters
  - Reset the module
- Prerequisites
  - Work alone

# Lab 2: Run a command



## Run a command - Syntax

- AT&x <CR>
- AT+S.[Command] <CR>

*AT Command Prefix*

***Not case sensitive***

| Utilities | |
|---|---|
| AT | Attention |
| AT+S.HELP | Display Help Text |
| AT+S.FWUPDATE | Perform a firmware update |
| AT+S.WIFI | Enable/Disable WiFi device |
| AT+CFUN | Comm Function (Reset) |
| AT+S.MFGTEST | Perform manufacturing tests |
| AT+S.PEMDATA | Configure certificate store |
| AT+S.ECHO | Send data out serial port |
| **Configuration** | |
| AT+S.GCFG | Get configuration value |
| AT+S.SCFG | Set configuration value |
| AT+S.SIDTXT | Set a textual SSID |
| AT&V | Display all configuration values |
| AT&F | Restore factory default settings |
| AT&W | Save current settings |
| AT+S.NVW | Write production settings |

| Network | |
|---|---|
| AT+S.PING | Send a ping to a specified host |
| AT+S.SCAN | Channels Scan |
| AT+S.HTTPGET | Issue an HTTP GET |
| AT+S.ROAM | Trigger WiFi reassociation sequence |
| **File Management** | |
| AT+S.FSC | Create a file |
| AT+S.FSA | Append to an existing file |
| AT+S.FSD | Delete an existing file |
| AT+S.FSL | List existing filename(s) |
| AT+S.FSP | Print the contents of an existing file |
| AT+S.HTTPDFSUPDATE | Update static HTTPD filesystem |
| **GPIO** | |
| AT+S.GPIOC | Configure General Purpose I/O |
| AT+S.GPIOR | Query General Purpose Input |
| AT+S.GPIOW | Set General Purpose Output |

## Response - Syntax

**xxxxxxxx**

- OK
- Command output followed by OK

- ERROR: Command not found
- ERROR: Unrecognized key

# Lab 2: Set the SPWF variables



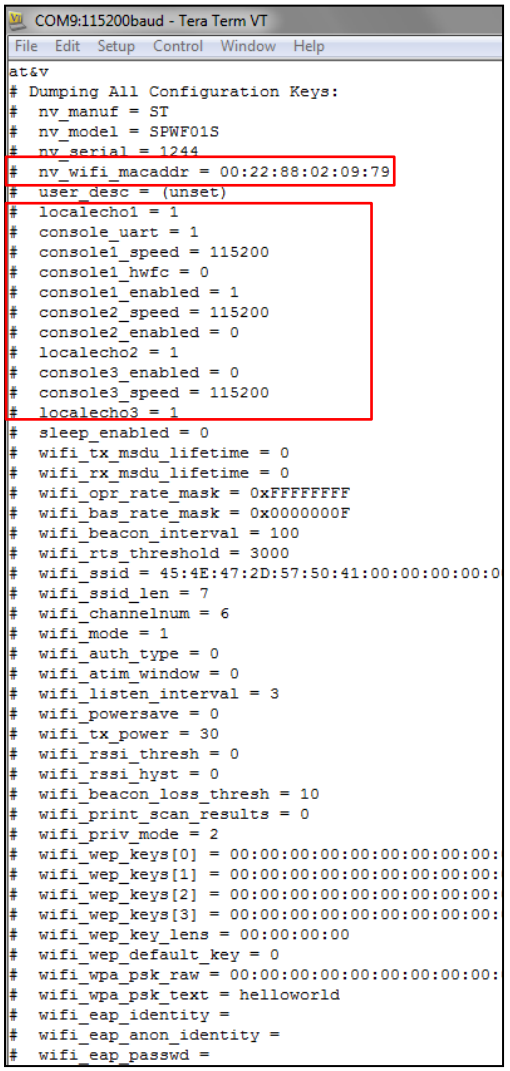
- Get the default configuration dump
  - Type **AT&V**
- Get a variable value
  - Type **AT+S.GCFG=ip_ipaddr**

## Config dump frame

| Variable Name | Variable Value |
|---|---|

How to change the variable value?

1. AT+S.SCFG=[var name],xxxx
2. AT&W



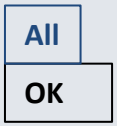
# Lab 2: Set the SPWF variables



- Get the default configuration dump
- Set host name
  - Type **AT+S.SCFG=ip_hostname,xxxxxxxx**

***Up to 31 characters (case sensitive), "spacebar" is allowed***

*Tera Term output*



# Lab 2: Set the SPWF variables



- Get the default configuration dump
- Set host name
- Set IP address, IP default gateway, IP DNS and IP netmask (for static usage)
  - Type **AT+S.SCFG=ip_ipaddr,192.168.0.1xx**
  - Type **AT+S.SCFG=ip_gw,192.168.0.1**
  - Type **AT+S.SCFG=ip_dns,192.168.0.1**
  - Type **AT+S.SCFG=ip_netmask,255.255.255.0**

*Tera Term output*

# Lab 2: Set the SPWF variables



- Get the default configuration dump
- Set host name
- Set IP address, IP default gateway, IP DNS and IP netmask (DHCP off)
- Save settings on the flash memory (mandatory after a variable change)
  - Type **AT&W**
- Reset the module
  - Type **AT+CFUN=1**

*Tera Term output*

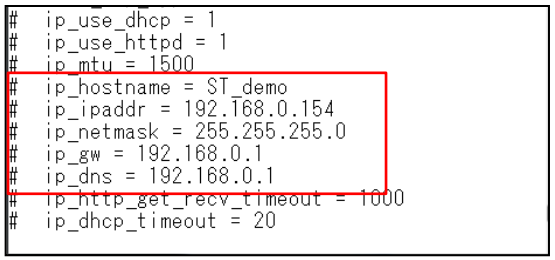
**All**

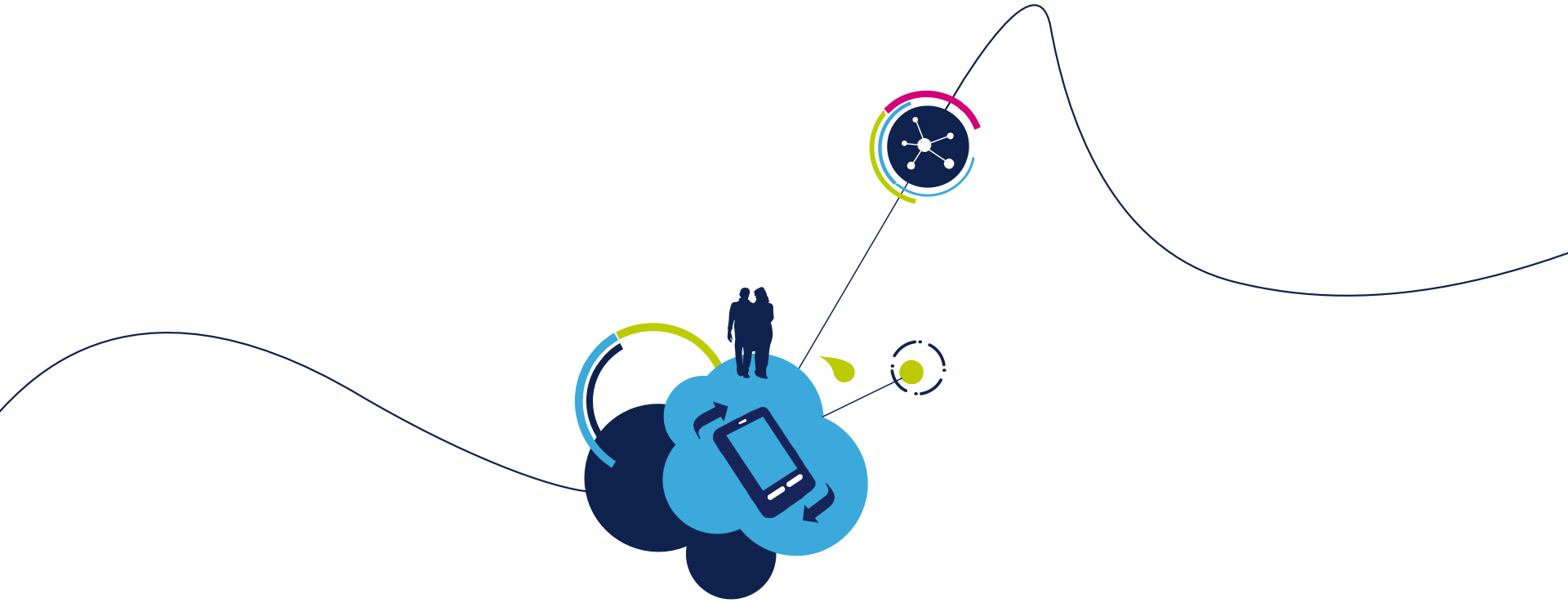
**+WIND:21:WiFi Scanning**
**+WIND:35:WiFi Scan Complete (0x0)**

# Lab 2: Set the SPWF variables



- Get the default configuration dump
- Set host name
- Set IP address, IP default gateway, IP DNS and IP netmask (DHCP off)
- Save settings on the flash memory (mandatory after a variable change) and reset the module
- Check the new configuration dump
  - Type **AT&V**





# Proceed to the next LAB!

# Lab 2a: FOTA update



- Objective
  - Upgrade the FW using the OTA file
- Prerequisites
  - OTA file (provided in the SPWF01S FW package)
  - External web server (i.e. Apache web server running on PC)

# Lab 2a: FOTA update



The SPWF01S module (with external flash on board) allows performing a Firmware Over-the-air update via a single HTTP GET.

The SPWF01S will validate the firmware image it downloads, load it into a staging area, then prompt  the user to issue a reset command in order to complete the update.

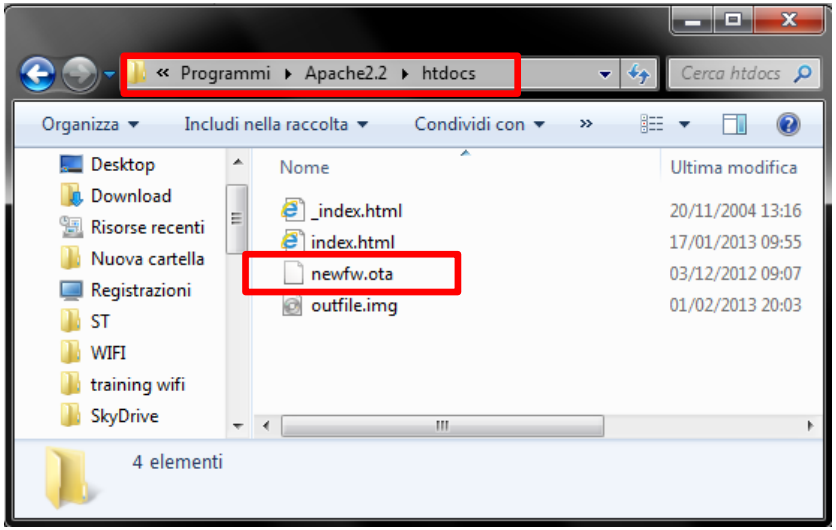
# Lab 2a: FOTA update



The **Apache Web Server** will be used in this LAB

(Apache Web Server is available at this link:
http://mirror.switch.ch/mirror/apache/dist//httpd/binaries/win32/httpd-2.2.25-win32-x86-openssl-0.9.8y.msi)

- Copy the OTA file (i.e. SPWF01S-140805-3f58d6b-RELEASE-main.ota) in the Apache 2.2 htdocs folder



# Lab 2a: FOTA update



The FWUPDATE command allows to perform a Firmware Over-the-air update via a single HTTP GET.

- Sintax
  - AT+S.FWUPDATE=<hostname>,<path>,<port>
- Configuration parameters
  - <hostname> Target host. DNS resolvable name or IP address
  - <path&queryopts> Document path and optional query arguments
  - <port> Target host port

# Lab 2a: FOTA update



- The module and the Apache Web server must be connected to the same network
- In Tera Term: type **AT+S.FWUPDATE=[Apache IP address],/[ota_file.ota]**

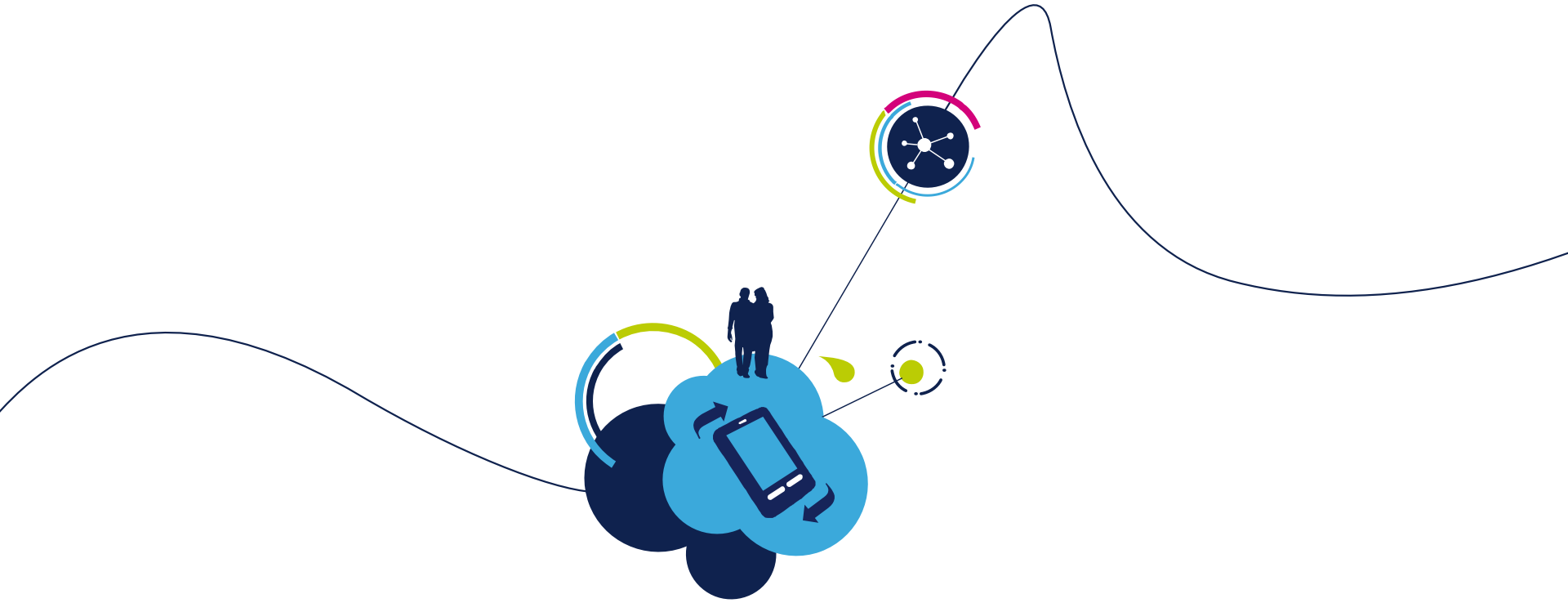
**i.e.** type **AT+S.FWUPDATE=192.168.x.yyy,/SPWF01S-140805-3f58d6b-RELEASE-main.ota**

- Reset the module to apply the new FW
  - Type **AT+CFUN=1**

*Tera Term output*

All

```
Staging F/W update for 'SPWF01SX.11' version '1203-120918_01'
F/W length 276824 @ 0x00002800 (offset 0x00000000, block len 4096)
Write len 4096 -> 0x0
Write len 4096 -> 0x1000
Write len 4096 -> 0x2000
Write len 4096 -> 0x3000
(note - deleted extra output for clarity)
Write len 4096 -> 0x41000
Write len 4096 -> 0x42000
Write len 2476 -> 0x43000 (final)
Wrote 276904 bytes
Complete! Update will be applied on next reboot.
(at+cfun=1)
```



# Proceed to the next LAB!

# Lab 3: Access point connection



- Objective
  - Scan for available networks
  - Join a network
  - Check the status/statistics variables

- Prerequisites
  - USB dongle and computer are set up as described in Lab 2
  - Work alone

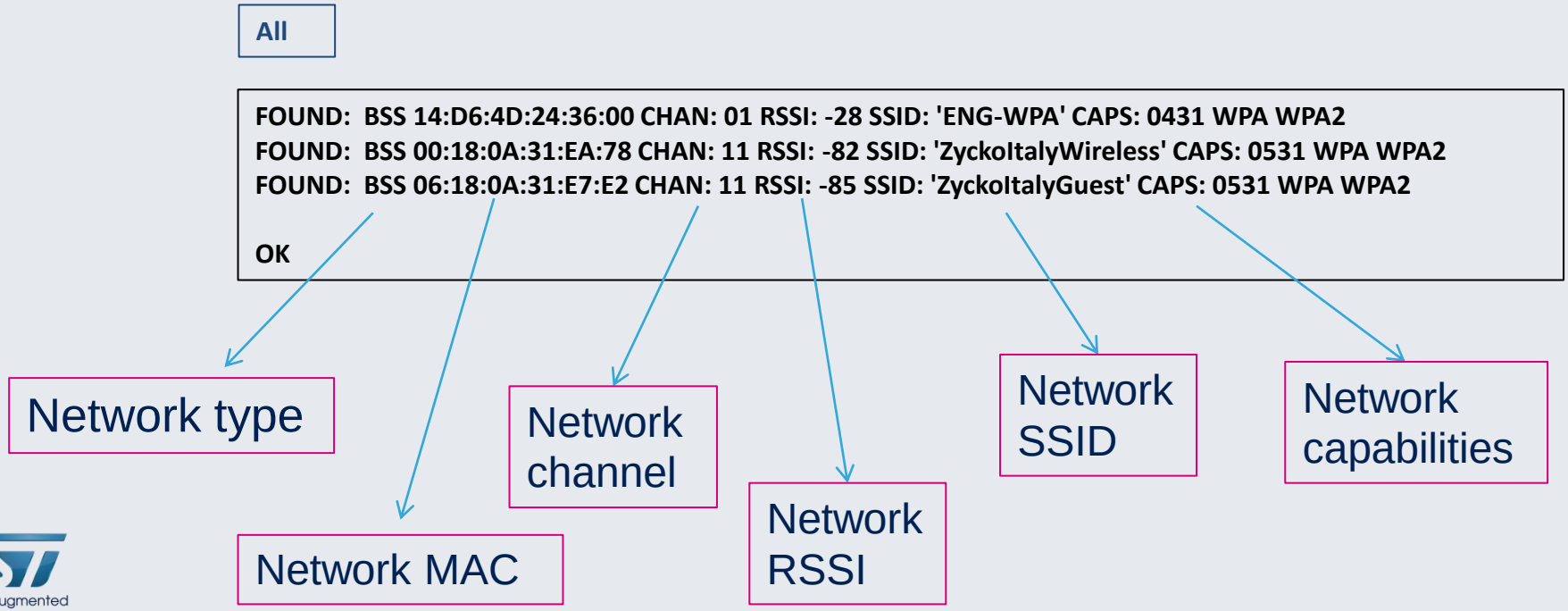
# Lab 3: Scan for available networks



The SCAN command performs an immediate scan for available networks. Infrastructure (AP) and IBSS (Ad-Hoc) networks are both reported. Network type, Channel, BSSID, SSID, Signal strength (RSSI), and 802.11 capabilities are all reported.

- Type **AT+S.SCAN** **(AT+S.SCAN=a,r to perform a non-filtered scan)**

*Tera Term output*



# Lab 3: Joining a network (WPA Key)



In order to be connected to an available Wifi network, the AP parameters setting is needed.

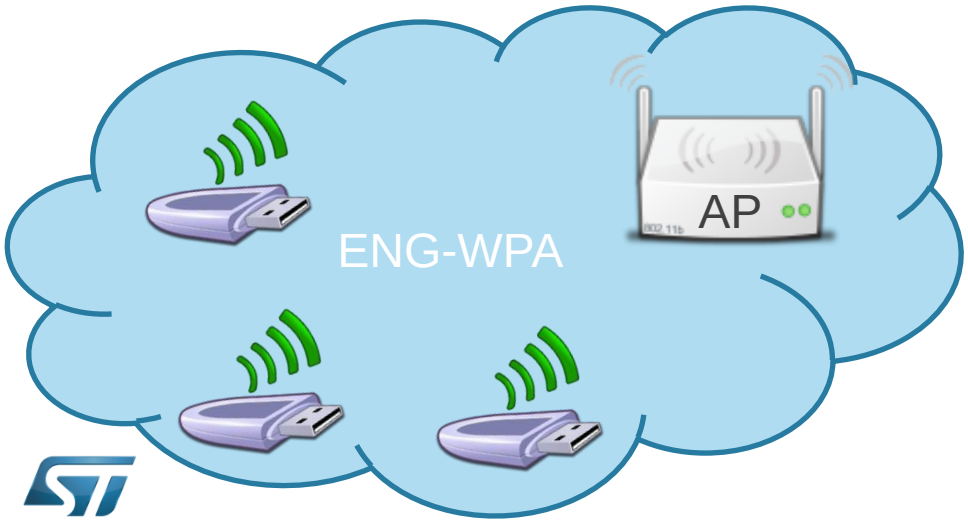
- Set the SSID
  - Type **AT+S.SSIDTXT=ENG-WPA**
- Set the password
  - Type **AT+S.SCFG=wifi_wpa_psk_text,helloworld**
- Set the network privacy mode (0=none, 1=WEP, 2=WPA-Personal (TKIP/AES) or WPA2-Personal (TKIP/AES))
  - Type **AT+S.SCFG=wifi_priv_mode,2**
- Set the network mode (1 = STA, 2 = IBSS, 3 = MiniAP)
  - Type **AT+S.SCFG=wifi_mode,1**

*Tera Term output*

# Lab 3: Joining a network (WPA Key)



- Save the settings on the flash memory and reset the module
  - Type **AT&W**
  - Type **AT+CFUN=1**




*Tera Term output*

All

```
+WIND:0:Console active
+WIND:46:WPA: Crunching PSK...
+WIND:32:WiFi Hardware Started
+WIND:21:WiFi Scanning
+WIND:35:WiFi Scan Complete (0x0)
+WIND:19:WiFi Join: 14:D6:4D:24:36:00
+WIND:25:WiFi Association with 'ENG-WPA' successful
+WIND:51:WPA Handshake Complete
+WIND:24:WiFi Up: 192.168.0.1xx
```

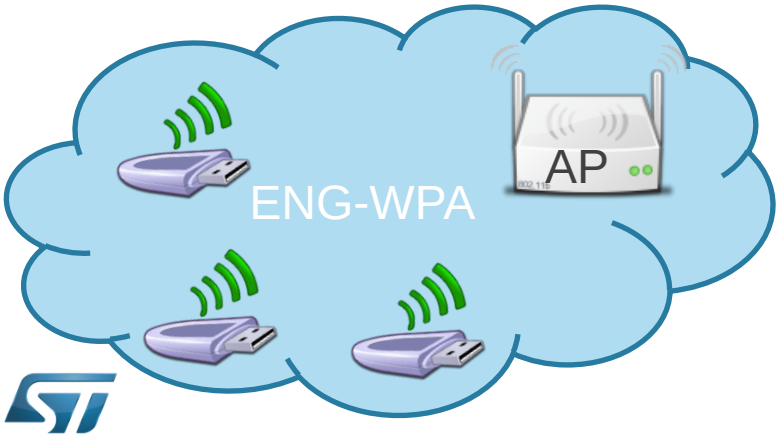
# Lab 3: Joining a network (WPA Key)



- Check the status/statistics variables
  - Type **AT+S.STS**

```
#  ip_ipaddr = 192.168.0.103
#  ip_netmask = 255.255.255.0
#  ip_gw = 192.168.0.1
#  ip_dns = 192.168.0.1
#  free_heap = 28256
#  min_heap = 26456
#  current_time = 39

OK
```

- Send a ping to the gateway (ip_gw)
  - Type **AT+S.PING=192.168.0.1**




*Tera Term output*

All

OK

# Lab 3: Joining a network (WEP Key)



Configure the module using the WEP key (4 possible combinations available)

## Sample table:

| AP configuration | AT command to be used |
|---|---|
| **Security Mode: WEP<br>WEP Key Length: 64 bit (10 hex digits)<br>Authentication: Open<br>Wep Key 1: 1234567890** | at+s.ssidtxt=ENG-WEP<br>AT+S.SCFG=wifi_wep_keys[0],1234567890<br>AT+S.SCFG=wifi_wep_key_lens,05<br>AT+S.SCFG=wifi_auth_type,0<br>AT+S.SCFG=wifi_priv_mode,1<br>AT+S.SCFG=wifi_mode,1<br>at&w<br>at+cfun=1 |
| **Security Mode: WEP<br>WEP Key Length: 64 bit (10 hex digits)<br>Authentication: Shared Key<br>Wep Key 1: 1234567890** | at+s.ssidtxt=ENG-WEP<br>AT+S.SCFG=wifi_wep_keys[0],1234567890<br>AT+S.SCFG=wifi_wep_key_lens,05<br>AT+S.SCFG=wifi_auth_type,1<br>AT+S.SCFG=wifi_priv_mode,1<br>AT+S.SCFG=wifi_mode,1<br>at&w<br>at+cfun=1 |

| AP configuration | AT command to be used |
|---|---|
| **Security Mode: WEP<br>WEP Key Length: 128 bit (26 hex digits)<br>Authentication: Open<br>Wep Key 1: 12345678901234567890123456** | at+s.ssidtxt=ENG-WEP<br>AT+S.SCFG=wifi_wep_keys[0],12345678901234567890123456<br>AT+S.SCFG=wifi_wep_key_lens,0D<br>AT+S.SCFG=wifi_auth_type,0<br>AT+S.SCFG=wifi_priv_mode,1<br>AT+S.SCFG=wifi_mode,1<br>at&w<br>at+cfun=1 |
| **Security Mode: WEP<br>WEP Key Length: 128 bit (26 hex digits)<br>Authentication: Shared Key<br>Wep Key 1: 12345678901234567890123456** | at+s.ssidtxt=ENG-WEP<br>AT+S.SCFG=wifi_wep_keys[0],12345678901234567890123456<br>AT+S.SCFG=wifi_wep_key_lens,0D<br>AT+S.SCFG=wifi_auth_type,1<br>AT+S.SCFG=wifi_priv_mode,1<br>AT+S.SCFG=wifi_mode,1<br>at&w<br>at+cfun=1 |

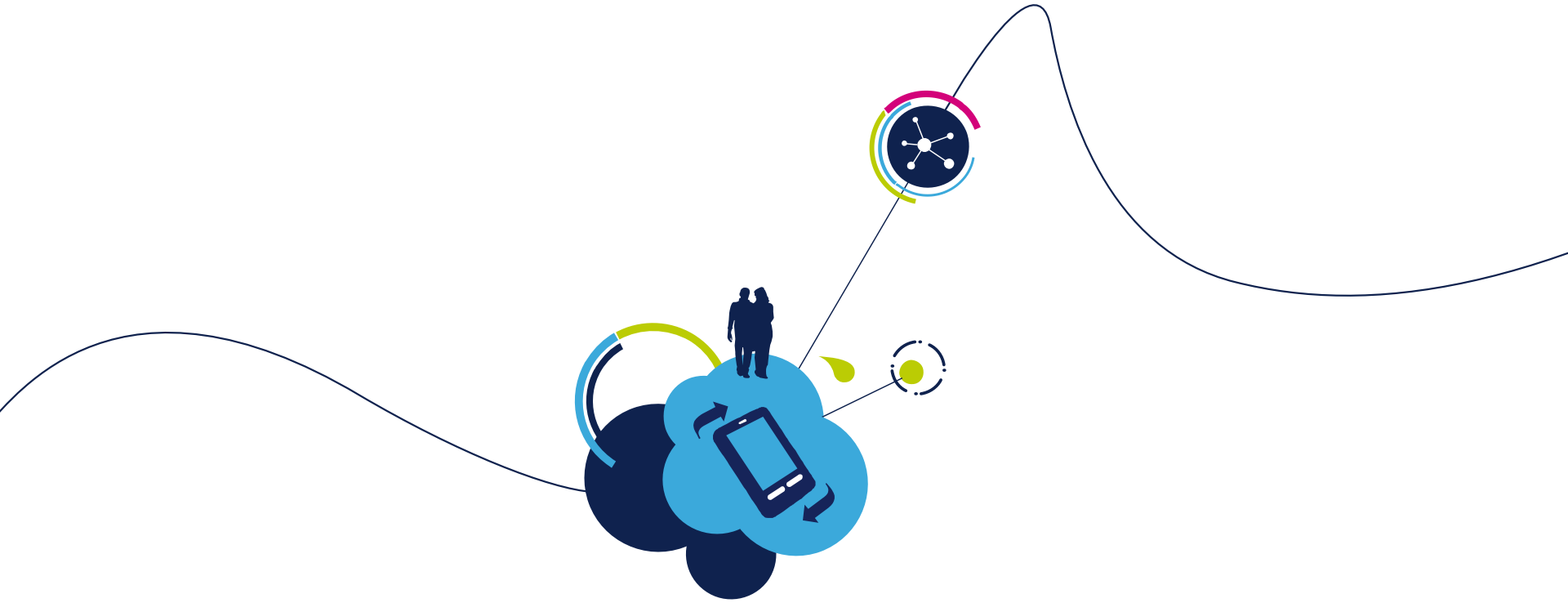
# Lab 3: Joining a network (WEP Key)



## Notes:

- “wifi_wep_key_lens” variable values: 05 and 0D
- It’s possible to enter any text string into a WEP key box in the AP, in which case it will be converted into a hexadecimal key using the ASCII values of the characters. A maximum of 5 text characters can be entered for 64 bit keys, and a maximum of 13 characters for 128 bit keys.
  In this case, it needs to manually convert your ASCII password to HEX and complete the wifi_wep_keys[0] variable with the HEX value.

- i.e. AP WEP key: **test1**
  ASCII to HEX: **74:65:73:74:31**
  So, the AT command is: **AT+S.SCFG=wifi_wep_keys[0],7465737431**



# Proceed to the next LAB!

# Lab 4: Web client mode

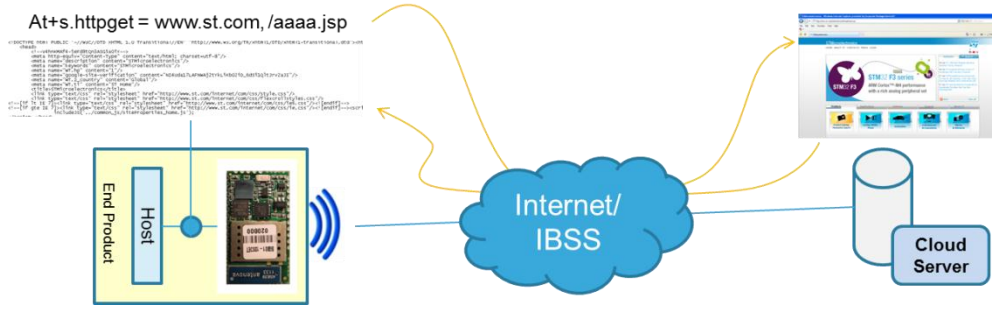


- Objective
  - HTTP GET
  - HTTP POST
  - PUSH DATA ON SERIAL PORT
- Prerequisites
  - USB dongle and computer are set up as described in Lab 2
  - Work in couple

# Lab 4: HTTP GET



The HTTP GET feature performs a single HTTP request to the specified host and path. The server response is printed on the UART enabled.




- Sintax
  - AT+S.HTTPGET=<hostname>,<path>,<port>
- Configuration parameters
  - <hostname>: target host. DNS resolvable name or IP address
  - <path>: document path
  - <port>: target port

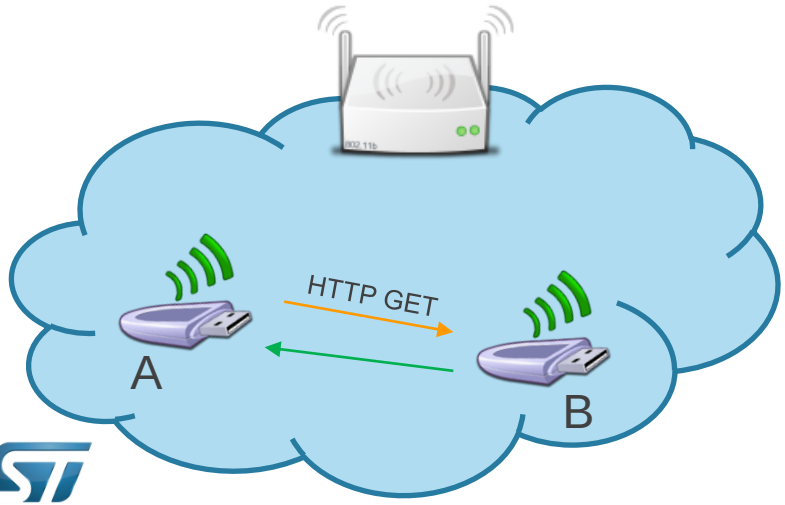
# Lab 4: HTTP GET



- Device A performs an HTTP GET to the Device B
  - Device A:

    AT+S.HTTPGET=<Device B IP addr>,/index.html

    Type **AT+S.HTTPGET=192.168.0.1xx,/index.html**




*Tera Term output*

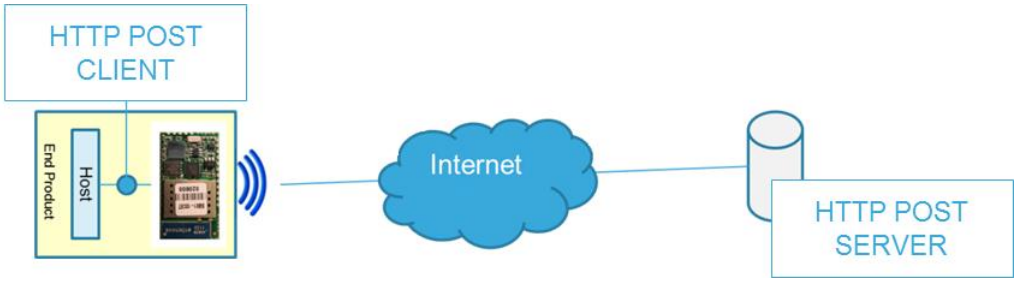
**Device A**

```
GET /index.html HTTP/1.0
User-Agent: SPWF01S
……….
<h1>ST SPWF01Sx.11 WiFi Module</h1>
<p>
Welcome to the ST SPWF01Sx.11 WiFi Module.
</p>
<p>
This page was delivered from the SPWF01Sx.11
internal HTTP server.
…..
<a href=/status.shtml>SPWF01Sx.11 Status Page</a>
</p>
</body>
</html>

OK
```

# Lab 4: HTTP POST



The HTTP POST performs a post of the given path to the specified host. The module can be only used as an HTTP POST client.



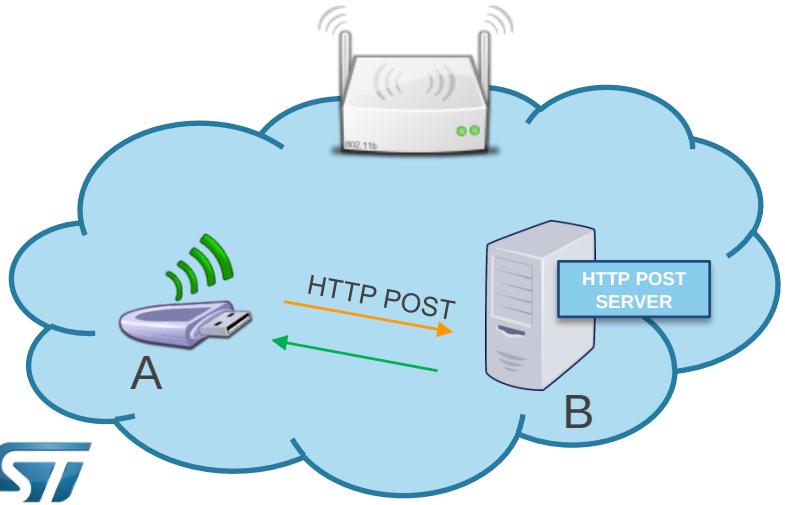

- Sintax
  - AT+S.HTTPPOST =<hostname>,<path&queryopts>,<formcontent>,<port>
- Configuration parameters
  - <hostname>: target host. DNS resolvable name or IP address
  - <path&queryopts>: document path
  - <formcontent>: form to be submitted
  - <port>: target port

# Lab 4: HTTP POST



- The SPWF01S performs an HTTP POST to an HTTP Post Test Server
  - Type:
    **at+s.httppost=posttestserver.com,/post.php,name=demo&email=mymail&subject=subj&body=message**

- The HTTP Post Test Server replies as displayed in the Tera Term output if the HTTP POST successfully




*Tera Term output*

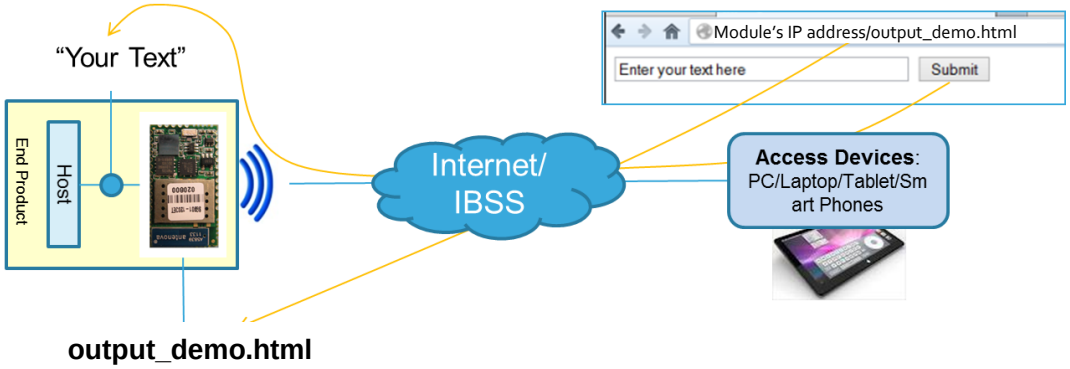
**Device A**

```
HTTP/1.1 200 OK
Date: Fri, 10 Jan 2014 13:24:14 GMT
Server: Apache
Access-Control-Allow-Origin: *
Vary: Accept-Encoding
Content-Length: 139
Connection: close
Content-Type: text/html

Successfully dumped 4 post variables.
View it at
http://www.posttestserver.com/data/2014/01/10/05.24.1443192628
Post body was 0 chars long.

OK
```

# Lab 4: Push data on serial port



A built-in html page “output_demo.html” allows to remotely push characters on the serial port from a remote browser.



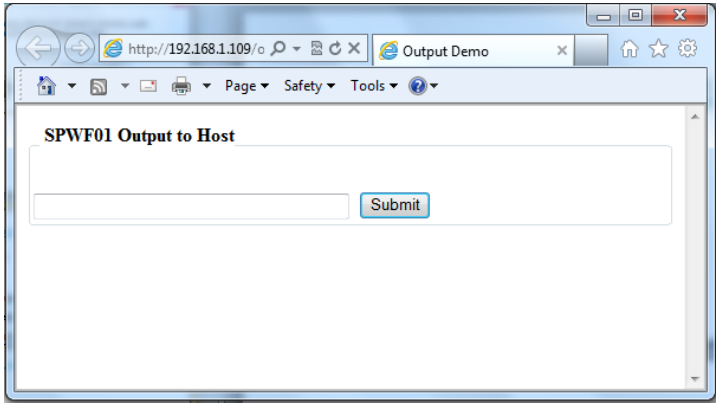
# Lab 4: Push data on serial port



- Find your IP address
  - Type **AT+S.STS**

```
#  ip_ipaddr = 192.168.0.103
#  ip_netmask = 255.255.255.0
#  ip_gw = 192.168.0.1
#  ip_dns = 192.168.0.1
#  free_heap = 28256
#  min_heap = 26456
#  current_time = 39
```

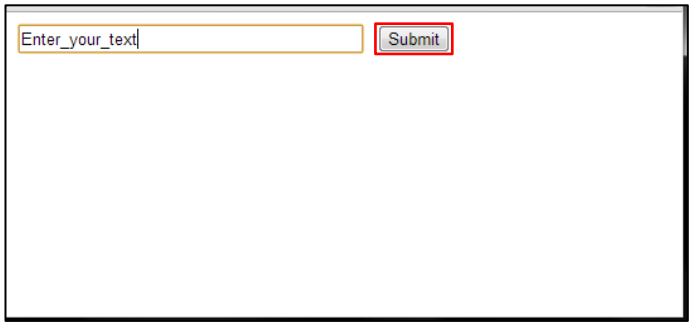
- Associate your computer with the AP
- Open your web browser
- In the address bar, type <module's IP Address>/output_demo.html
  - Type 192.168.0.1xx/output_demo.html



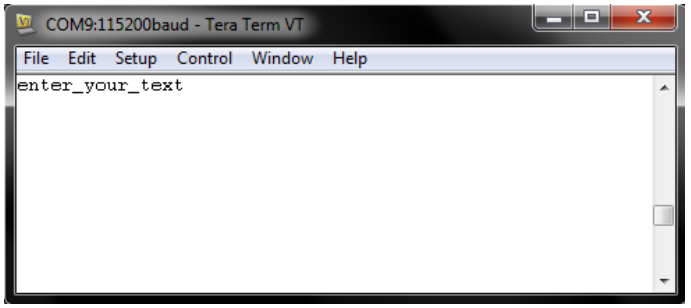
# Lab 4: Push data on serial port

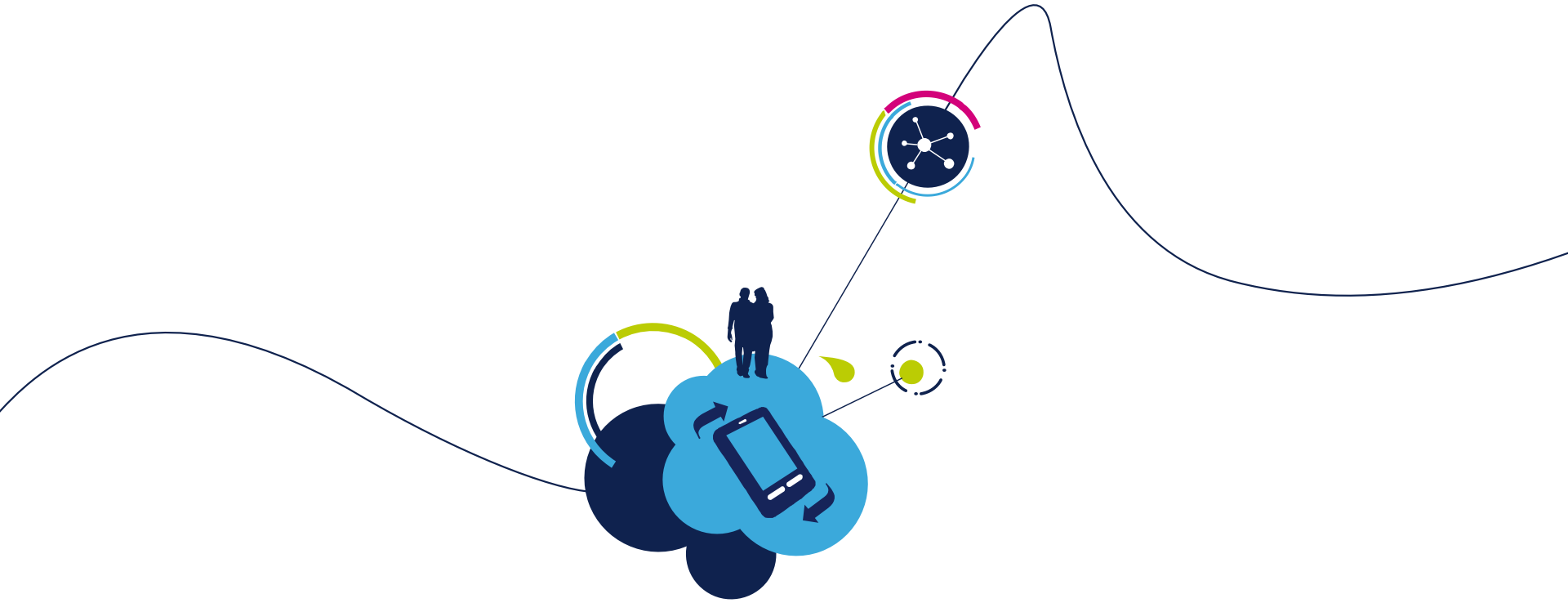


- Enter the text
- Submit




- The text will be sent to the serial port of the module





# Proceed to the next LAB!

# Lab 5: Web Server use mode



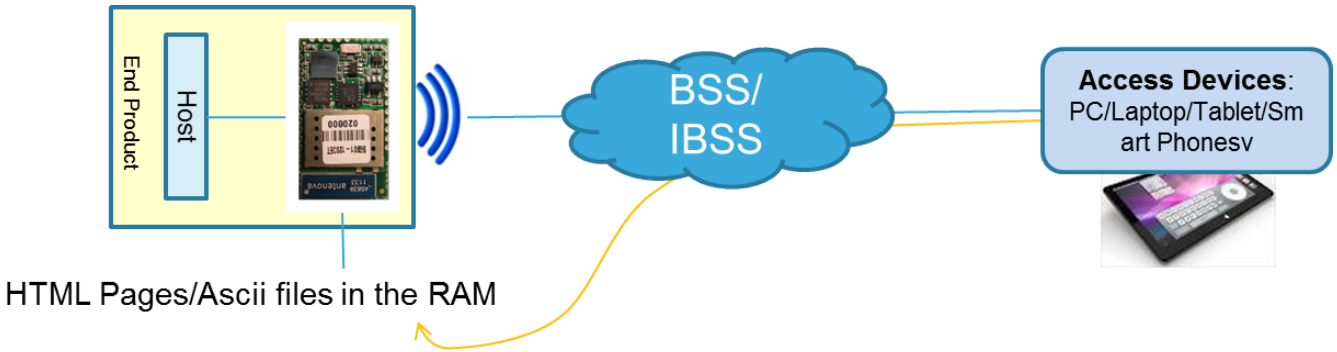
- Objective
  - List existing files
  - Print a file
  - Create a file
  - Append to an existing file
  - Delete an existing file

- Prerequisites
  - USB dongle and computer are set up as described in Lab 2
  - Work alone

# Lab 5: Web server



The Web Server feature allows to create, print and delete ASCII files in the **RAM** memory of the module.



# Lab 5: List existing files



The FSL command lists type, sizes and name of all the existing files.

- Type **AT+S.FSL**

*Tera Term output*

**All**

```
I 461 /input_demo.shtml
I 180 /message.shtml
I 384 /output_demo.html
I 614 /index.html
I 157 /peers.shtml
I 193 /config.shtml
I 174 /status.shtml
I 212 /404.html
I 2022 /firstset.html
I 2898 /remote.html

OK
```

Files stored in the STM32 FLASH memory

# Lab 5: Print a file



The FSP command prints the content of an existing file.

- Type **AT+S.FSP=/index.html**

*Tera Term output*

**All**

```
<html>
<head><title>ST SPWF01S</title></head>
<body bgcolor="white" text="black">
<h1>ST SPWF01S Intelligent WiFi Module</h1>
<p>Welcome to the ST SPWF01S intelligent Wi-Fi module.</p>
<p>This page was delivered from the SPWF01S internal HTTP
server.</p>
<p>
<a href=/config.shtml>SPWF01S Configuration Settings
Page</a>
</p>
<p>
<a href=/status.shtml>SPWF01S Status Page</a>
</p>
<p>
<a href=/peers.shtml>SPWF01S Peers Page</a>
</p>
<p>More information about the SPWF01S and other ST
products can be found at ST's website:
<a
href="http://www.ST.com/">http://www.ST.com/</a>.</p>
</body>
</html>

OK
```

***Tip: How to use the offset and length parameters:***
***AT+S.FSP=/index.html,[offset],[length]***

***i.e. Type AT+S.FSP=/index.html,5,20***

***Length parameter is mandatory if is used the offset.***

# Lab 5: Create a file

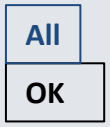


The FSC command allows to create a file inside the SPWF for delivery by the SPWF HTTP server.

- Sintax
  - AT+S.FSC=/<filename>,<max_len>
- Configuration parameters
  - < filename >: filename to create
  - < max_len >: amount of space to allocate for file, max = 4096 bytes

Type **AT+S.FSC=/wifidemo.html,1965**

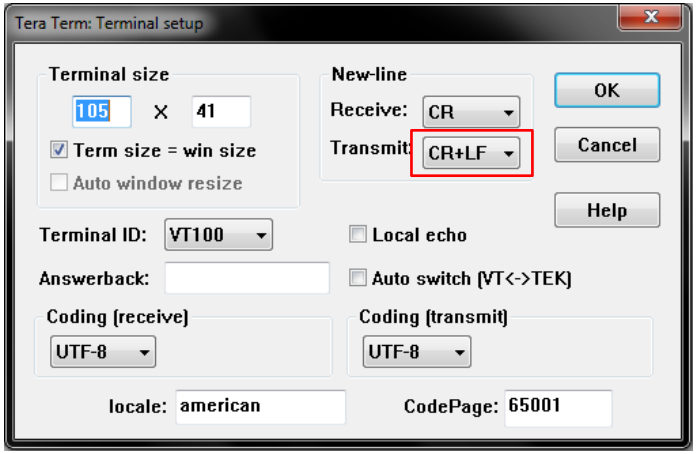
*Tera Term output*



# Lab 5: Configuring the UART



- Set CR+LF in the Terminal setup (to properly paste text in the terminal):
  - Tera Term: Setup → Terminal → Transmit: CR+LF



# Lab 5: Append to an existing file



The FSA command allows to append blocks of data to an existing file.

This command accepts data after the <cr> at the end of the command line.

- Sintax
  - AT+S.FSA=/<filename>,<datalen>
- Configuration parameters
  - < filename >: filename pre-created
  - < datalen >: number of characters to append to the file

Type **AT+S.FSA=/wifidemo.html,1965**

**Tips**

**Now SPWF accepts data**

# Lab 5: Append to an existing file



- Open wifidemo.txt in the following directory: Hands_on folder/HTML_demo_page
- Check you have set up CR+LF in the Terminal setup (for more information, **refer to the slide 48**)
- Copy the file content into Tera term (paste with ALT+V or mouse right button)

*Tera Term output*

# Lab 5: Append to an existing file



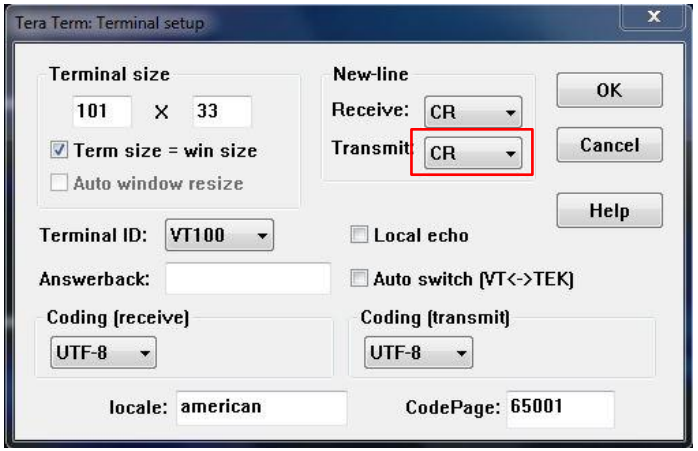
- Open wifidemo.html
  - Open your Web browser (suggested Google Chrome for HTML5 test)
  - In address bar, type <SPWF IP addr>/wifidemo.html

    Type **192.168.0.1xx/wifidemo.html**

# Lab 5: Configuring the UART



- Re-set CR in the Terminal setup as by default
  - Tera Term: Setup → Terminal → Transmit: CR



# Lab 5: List existing files



Let's list the files

- Type **AT+S.FSL**

File stored in the RAM memory

*Tera Term output*

```
All
D 1965 /wifidemo.html
I 461 /input_demo.shtml
I 180 /message.shtml
I 384 /output_demo.html
I 614 /index.html
I 157 /peers.shtml
I 193 /config.shtml
I 174 /status.shtml
I 212 /404.html
I 2022 /firstset.html
I 2898 /remote.html

OK
```

# Lab 5: Delete an existing file

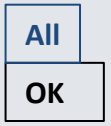


The FSD command allows to delete an existing file by name. Static files may not be deleted.

- Sintax
  - AT+S.FSD=/<filename>

Type **AT+S.FSD=/wifidemo.html**

*Tera Term output*

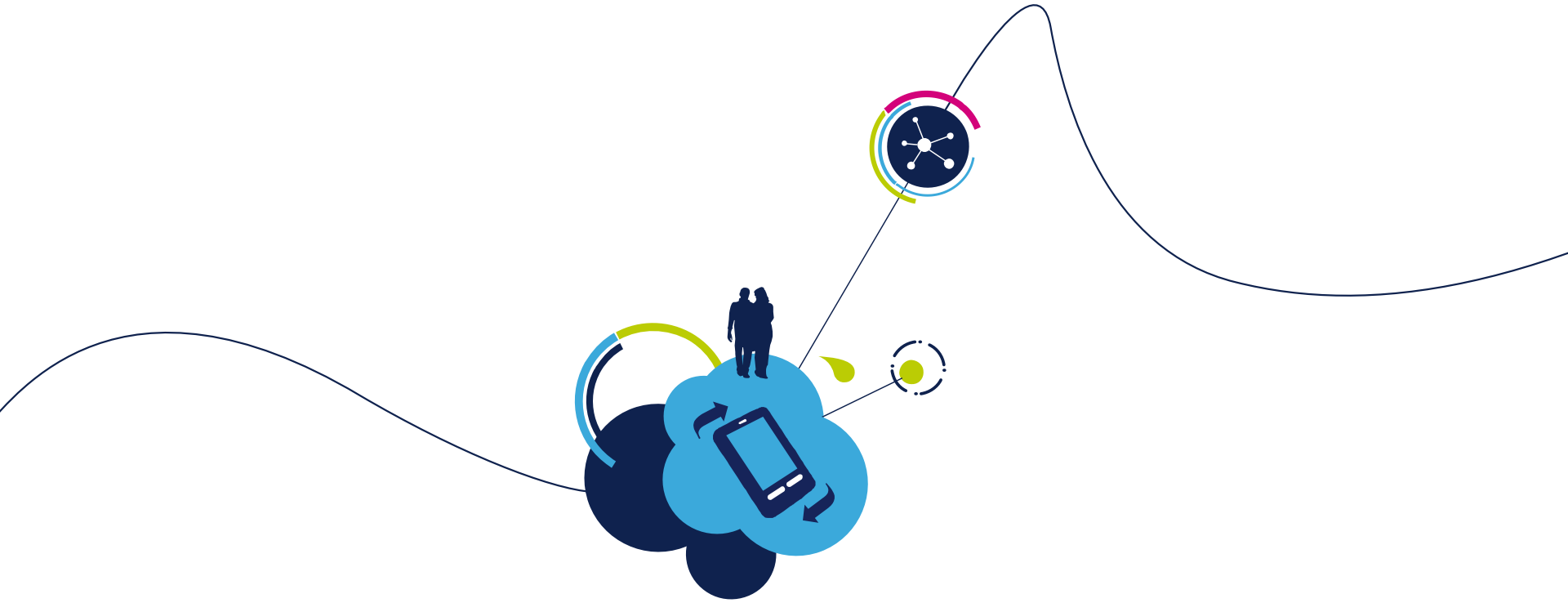


# Lab 5: Delete an existing file



- Re-open wifidemo.html
  - Open your Web browser
  - In address bar: <SPWF IP addr>/wifidemo.html

    Type **192.168.0.1xx/wifidemo.html**



# Proceed to the next LAB!

# Lab 6: Web Server use mode



- Objective
  - Create an image file
  - Filesystem update over-the-air

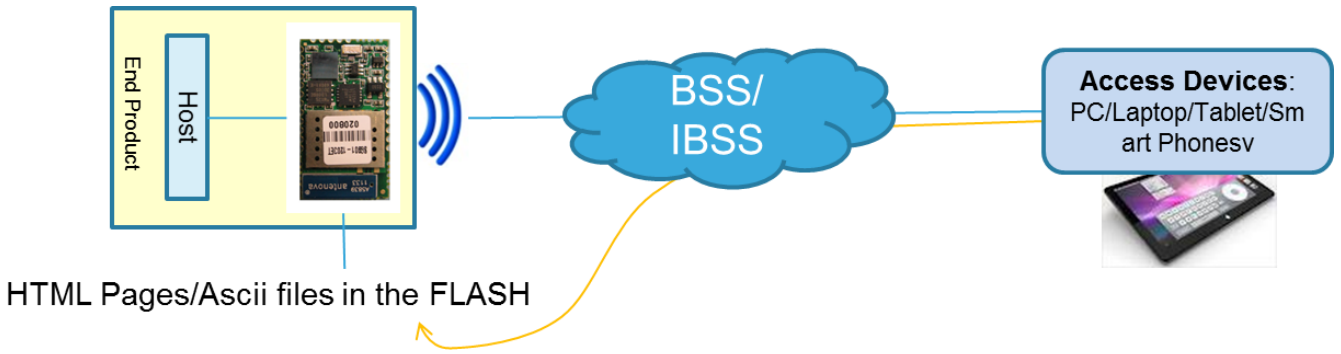
- Prerequisites
  - Work alone

# Lab 6: Web server



The Web Server feature also allows to upload generic files (PDF, HTML, DOC, JPEG and so on) in the **EXTERNAL FLASH** memory of the module. It needs to follow the below steps:

- create an image file (IMG) using the tool provided in the Hands-on package (the IMG contains the files to upload)
- Put the image in a generic and accessible Web Server (Apache Web server will be used in the Hands-on as example)
- Using the proper AT command, the IMG will be retrieved over-the-air and the static files will be created in the external flash memory



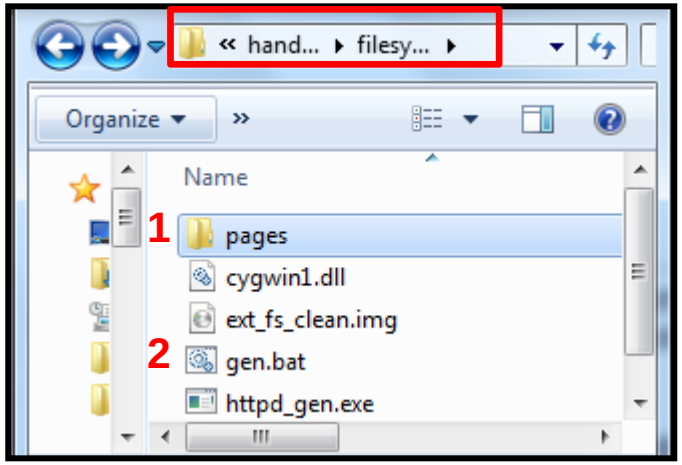
# Lab 6: Create an image file



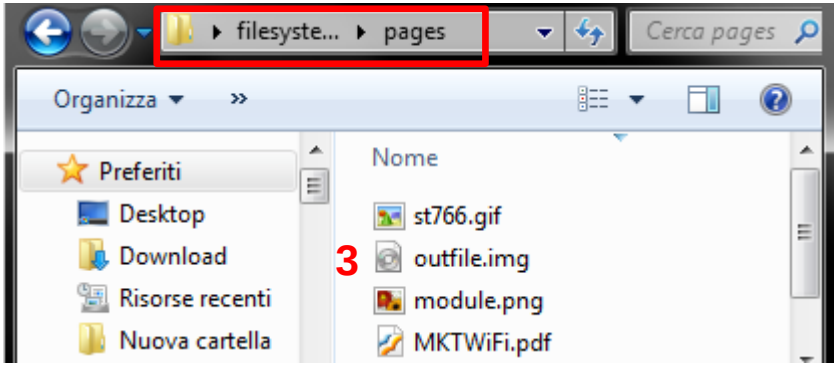
- Open the following folder and extract all the files:

  ../hands_on_wifi/filesystem_update

- Put your own files to upload in the **pages** folder (max 512 Kbytes)
- Run gen.bat




- Open the **pages** folder and copy outfile.img in your external Web Server (Apache)



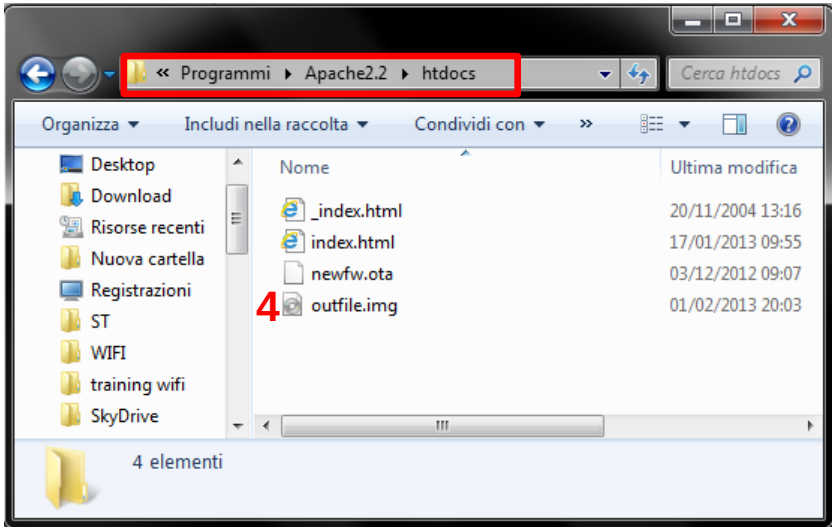
# Lab 6: Create an image file



The **Apache Web Server** will be used in this tutorial

(Apache Web Server is available at this link:
http://mirror.switch.ch/mirror/apache/dist//httpd/binaries/win32/httpd-2.2.25-win32-x86-openssl-0.9.8y.msi)

- Copy outfile.img in the Apache 2.2 *htdocs folder* (this is the default root server directory)



# Lab 6: Filesystem update



The HTTPDFSUPDATE command allows to create static files inside the SPWF for delivery by the SPWF HTTP server.

- Sintax
  - AT+S.HTTPDFSUPDATE=<hostname>,<path>
- Configuration parameters
  - <hostname>: external web server. DNS resolvable name or IP address
  - <path>: document path
  - <port>: target host port

# Lab 6: Filesystem update



- In Tera Term: type
  **AT+S.HTTPDFSUPDATE=192.168.x.10x,/outfile.img**

**Warning**

**The file system will be permanently deleted.**
**The new IMG will overwrite the existent files in the file system**

*Tera Term output*

**All**

```
Image length 357957 (offset 0x00080000, block len 4096)
Write len 4096 -> 0x80000
Write len 4096 -> 0x81000
Write len 4096 -> 0x82000
…….
……..
……
Write len 4096 -> 0xD6000
Write len 1612 -> 0xD7000 (final)
Wrote 357964 bytes
Complete! Please reboot. (at+cfun=1)

OK
```

- Reset the module
  - Type **AT+CFUN=1**

# Lab 6: Filesystem update



Let's list the files

- Type **AT+S.FSL**

*Tera Term output*

**All**

```
E 2615 /st766.gif
E 112374 /module.png
E 2430 /index1.html
E 419 /index.html
E 239887 /MKTWiFi.pdf
I 461 /input_demo.shtml
I 180 /message.shtml
I 384 /output_demo.html
I 614 /index.html
I 157 /peers.shtml
I 193 /config.shtml
I 174 /status.shtml
I 212 /404.html
I 2022 /firstset.html
I 2898 /remote.html

OK
```

Files stored in the EXT FLASH memory

# Lab 6: Filesystem update



- Open index1.html page
  - Open your web browser
  - In the address bar: <SPWF IP addr>/index1.html

    Type **192.168.x.1xx/index1.html**

# Lab 6: Filesystem erase



Erase the external httpd filesystem

- Type **AT+S. HTTPDFSERASE**

**The files stored in the EXT FLASH memory will be erased**

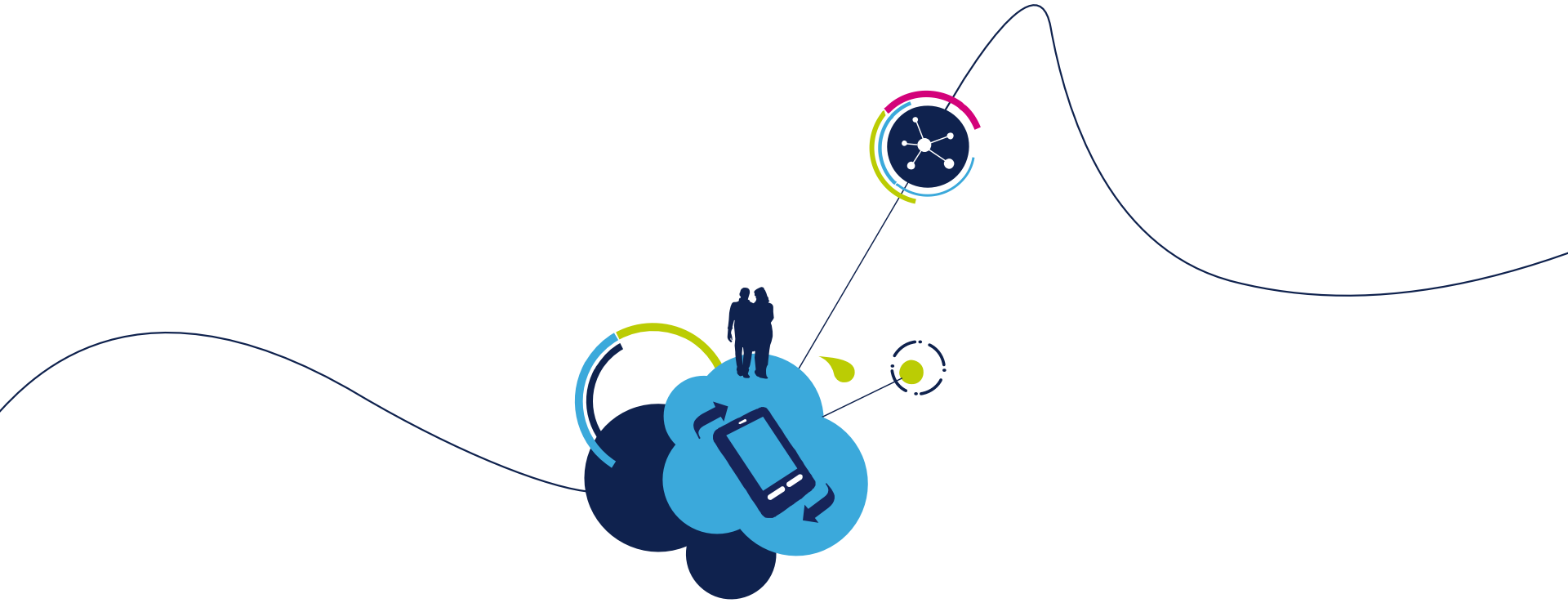
Let's list the files

- Type **AT+S.FSL**

*Tera Term output*

All

```
I 461 /input_demo.shtml
I 180 /message.shtml
I 384 /output_demo.html
I 614 /index.html
I 157 /peers.shtml
I 193 /config.shtml
I 174 /status.shtml
I 212 /404.html
I 2022 /firstset.html
I 2898 /remote.html

OK
```



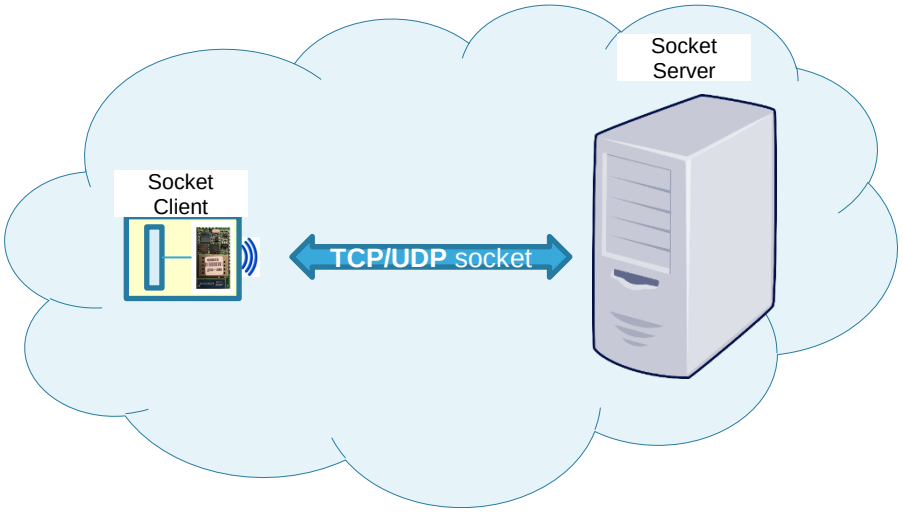
# Proceed to the next LAB!

# Lab 7: Socket interface



- Objective
  - Open TCP/UDP connection
  - Write data to socket
  - Read data from socket
  - Close socket

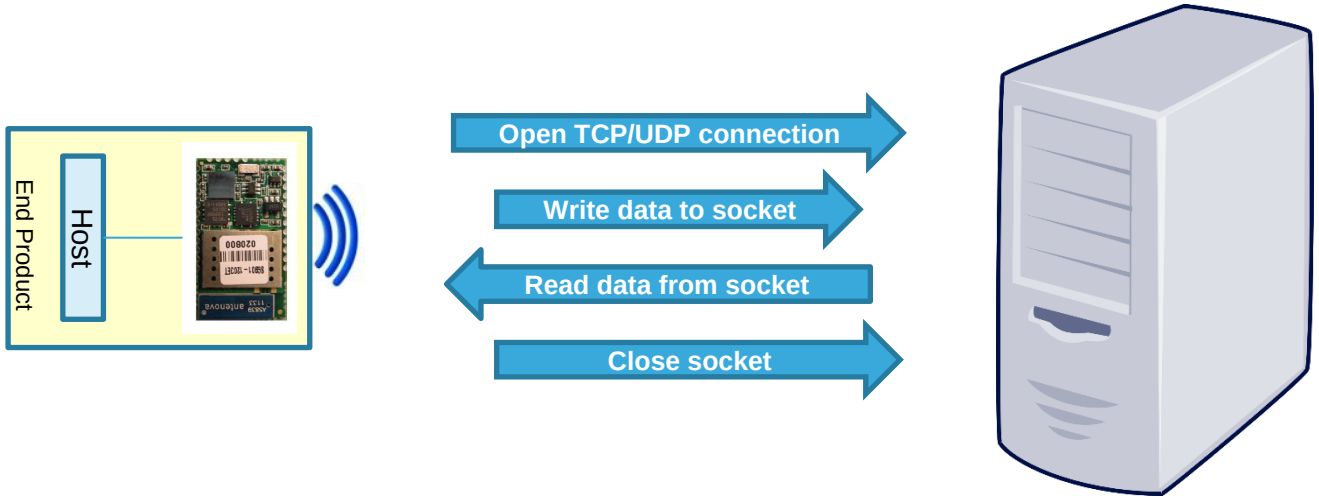
- Prerequisites
  - Work alone



# Lab 7: TCP/UDP/UART socket interface



The Socket interface allows communication via TCP, UDP and UART. The SPWF is both a client and a server socket. In this LAB, will be detailed the socket client feature.



# Lab 7: Open TCP/UDP connection



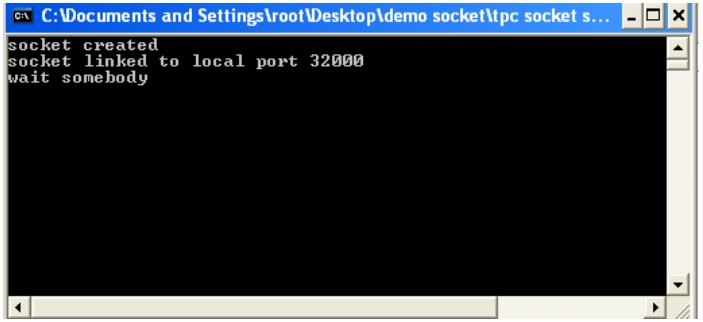
The SOCKON command allows to open a TCP/UDP connection to a specified host (up to 8 socket connections at same time).

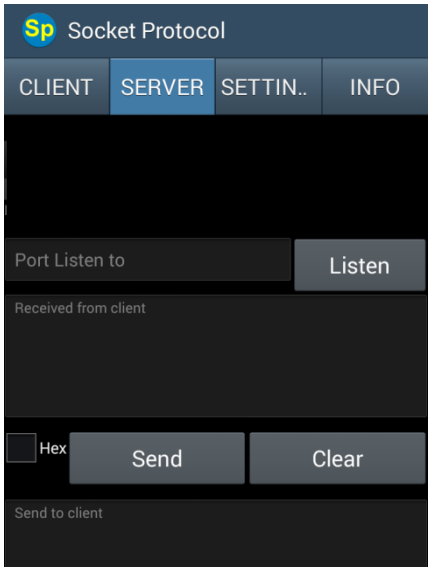
- Sintax
  - AT+S.SOCKON=<hostname>,<port>,<protocol>,ind
- Configuration parameters
  - <hostname>: target host. DNS resolvable name or IP address
  - <port>: TCP/UDP socket port
  - <protocol>: ***t*** for TCP socket, ***u*** for UDP socket
  - ind: indicate when data has arrived (optional). Up to 4 consecutive “Pending data” message (without any AT+S.SOCKR) are guaranteed. It is suggested to empty the buffer using the **AT+S.SOCKR** command and to avoid exceeding 4 indications.

# Lab 7: Open TCP/UDP connection



- Open the TCP socket server (disable the firewall to properly run it)
  - Folder ../hands_on_wifi/tcp socket server/server.exe
- The TCP server listens for incoming connections on the port 32000. It sends back all data received.



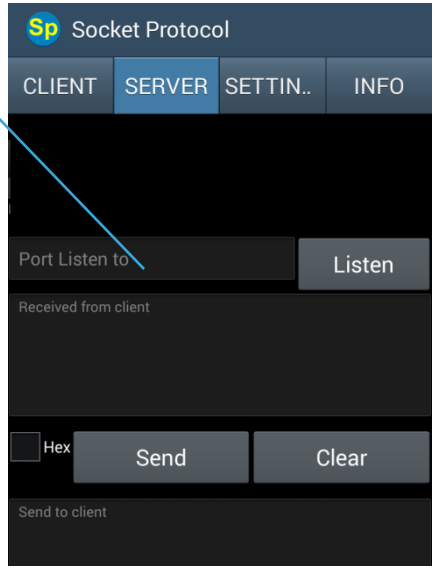

life.augmented

# **Optional** - Lab 7: Open TCP/UDP connection



- Using an Android platform, the socket server can be opened using a specific APP (Socket Protocol, available on Play Store)

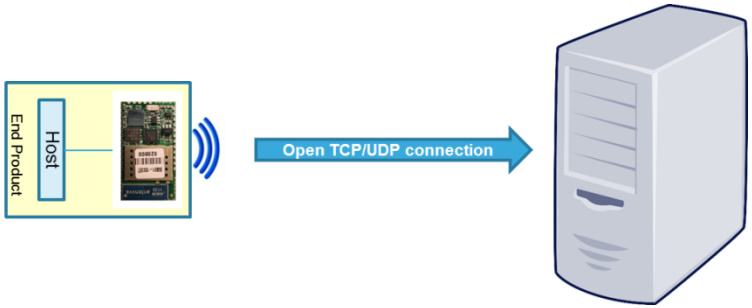
- The TCP socket server can be configured to listen for incoming connections on the port 32000.
  - Type 32000 on «Port Listen to» box
  - Click on Listen



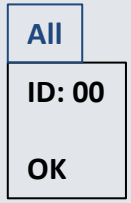
# Lab 7: Open TCP/UDP connection



- Type **AT+S.SOCKON=192.168.0.10x,32000,t,ind**
- The client and the server use the socket identifier (ID) displayed




*Tera Term output*



# Lab 7: Write data to socket



The SOCKW command allows to write data to the specified ID socket.

This command accepts data after the <cr> at the end of the command line.

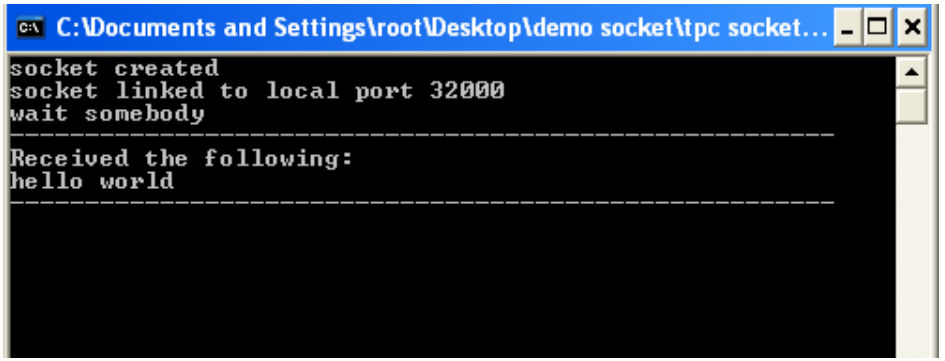
- Sintax
  - AT+S.SOCKW=<ID>,<len>
- Configuration parameters
  - <ID>: socket identifier
  - <len>: data length to send

# Lab 7: Write data to socket



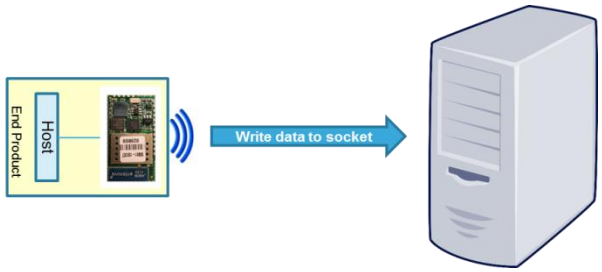
- Write data
  - Type **AT+S.SOCKW=00,13**

***Note: the module is waiting 13 bytes to be written to the socket. As soon as 13 bytes (or characters) are sent from the terminal, the module is going to write them to the socket and will be ready to receive new commands.***

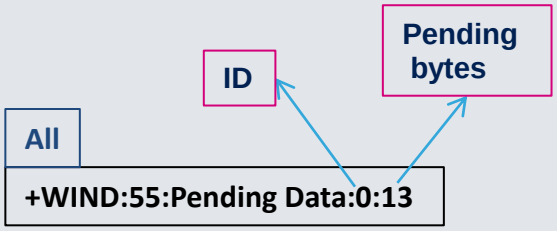



  - Type **hello world<CR>**

- SPWF shows that there are <pending data> and their amount




*Tera Term output*



# Lab 7: Read data from socket



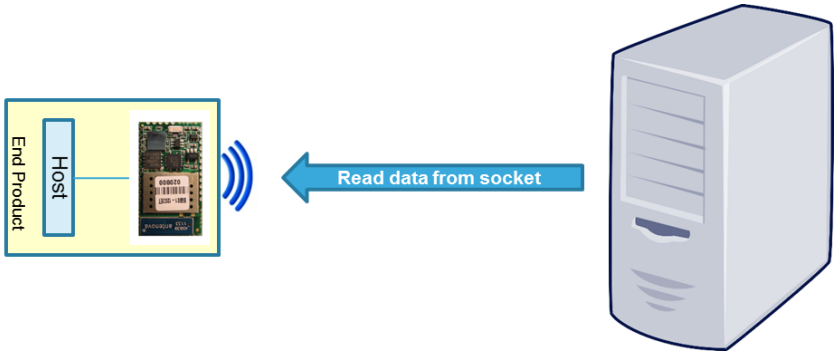
The SOCKR command allows to read data from socket.

- Sintax
  - AT+S.SOCKR=<ID>,<len>
- Configuration parameters
  - <ID>: socket identifier
  - <len>: data length to read

# Lab 7: Read data from socket



- Read data
  - Type **AT+S.SOCKR=00,13**




*Tera Term output*

# Lab 7: Close socket



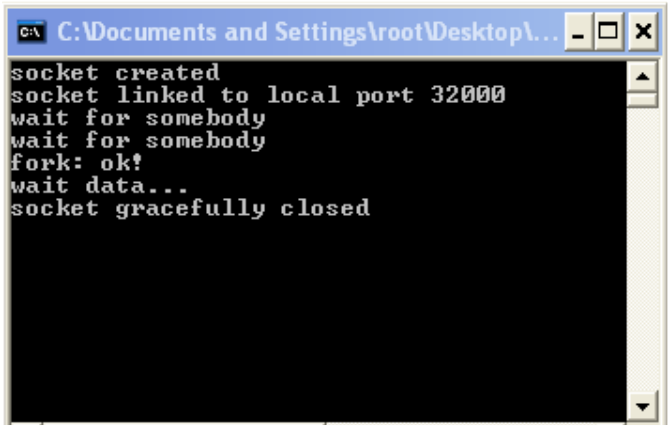
The SOCKC command allows to close socket.

- Sintax
  - AT+S.SOCKC=<ID>
- Configuration parameters
  - <ID>: socket identifier

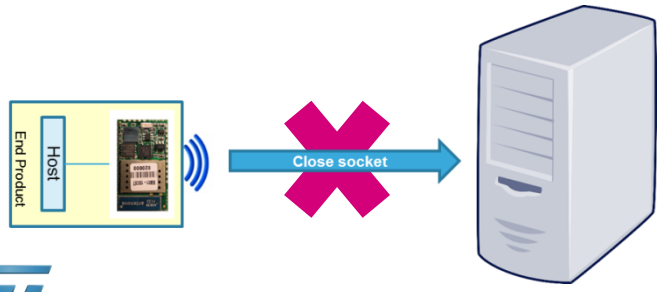
# Lab 7: Close socket

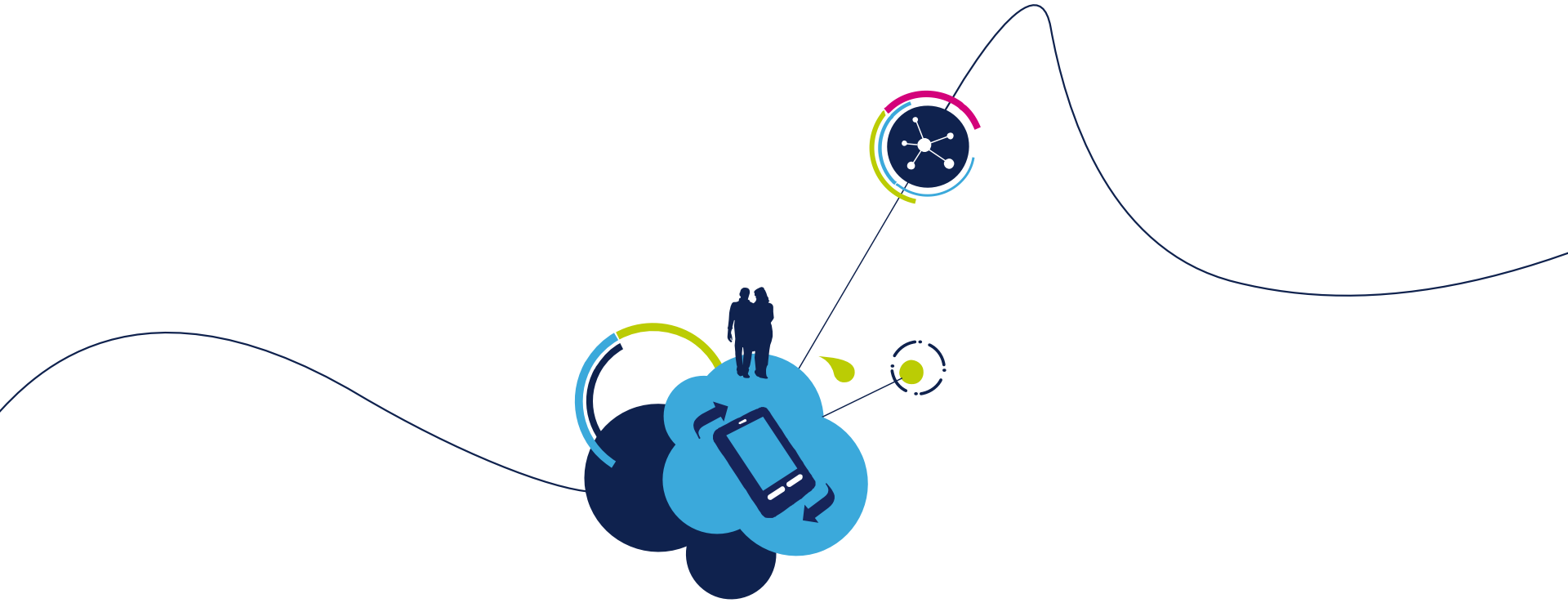


- Close socket
  - Type **AT+S.SOCKC=00**




*Tera Term output*



# Proceed to the next LAB!

# Lab 8: MiniAP mode for the first set



- Objective
  - Create a direct connection between the module and an end device
  - First set of the module in order to enable the connection between the module and a generic AP

- Prerequisites
  - Work alone

# Lab 8: Configuring the module in MiniAP mode



The Mini AP mode is available starting from the following FW release: ***SPWF01S-131115-de4568d-RELEASE***

In order to set the module in Mini AP mode, the following parameters are needed

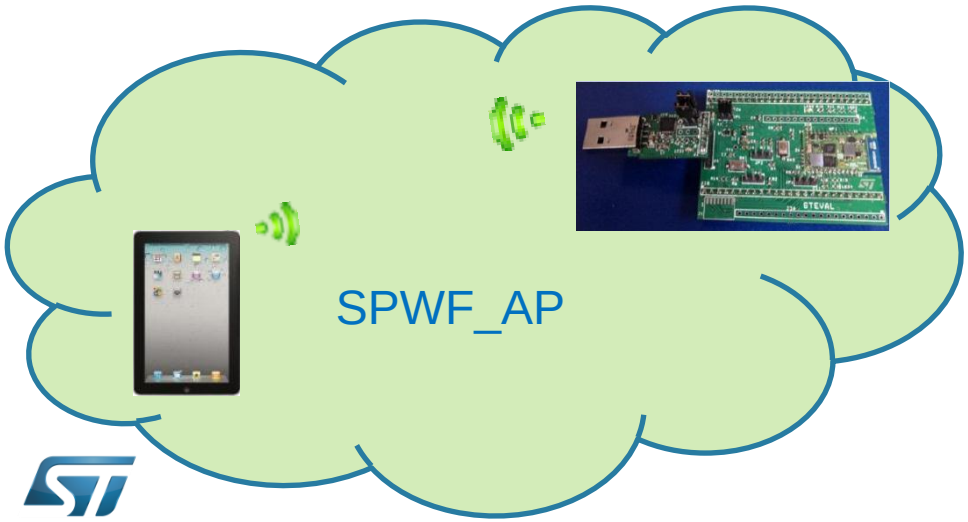
- Set the SSID
  - Type **AT+S.SSIDTXT=SPWF_AP**
- Set the network privacy mode (0=OPEN or 1=WEP are supported)
  - Type **AT+S.SCFG=wifi_priv_mode,0**
- Set the network mode (1 = STA, 2 = IBSS, 3 = MiniAP)
  - Type **AT+S.SCFG=wifi_mode,3**

*Tera Term output*

# Lab 8: Configuring the module in MiniAP mode



- Save the settings on the flash memory and reset the module
  - Type **AT&W**
  - Type **AT+CFUN=1**




*Tera Term output*

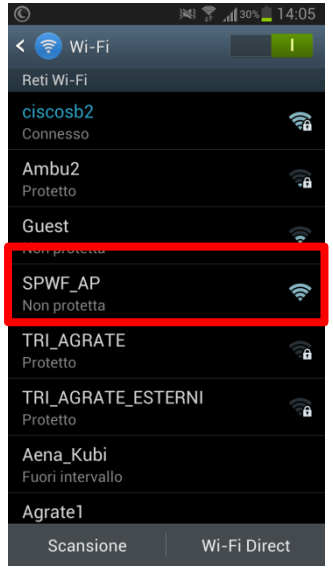
**All**

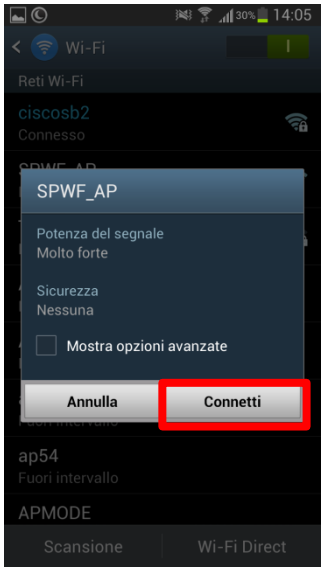
```
+WIND:0:Console active
+WIND:32:WiFi Hardware Started
+WIND:26:WiFi Started AP
+WIND:24:WiFi Up: 172.18.151.1
```

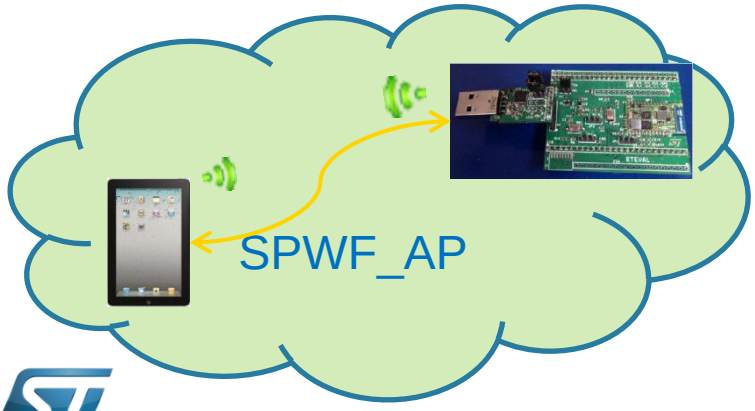
# Lab 8: Mini AP mode



- Associate your end device to the SPWF_AP network
  - Find the SPWF_AP network and connect the end device to the module




*Tera Term output*

**All**

**+WIND:28:90:18:7C:96:0D:0B Associated**
**+WIND:29:DHCP reply for 172.18.151.2/90:18:7C:96:0D:0B to 172.18.151.2**
**+WIND:29:DHCP reply for 172.18.151.2/90:18:7C:96:0D:0B to 172.18.151.2**

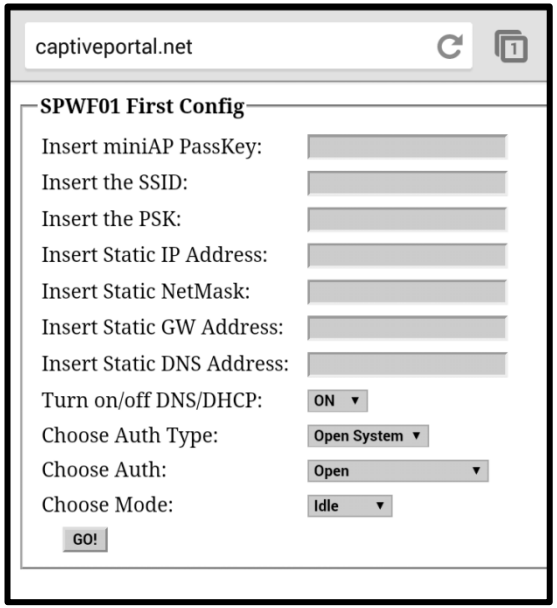
life.augmented

# Lab 8: Mini AP mode

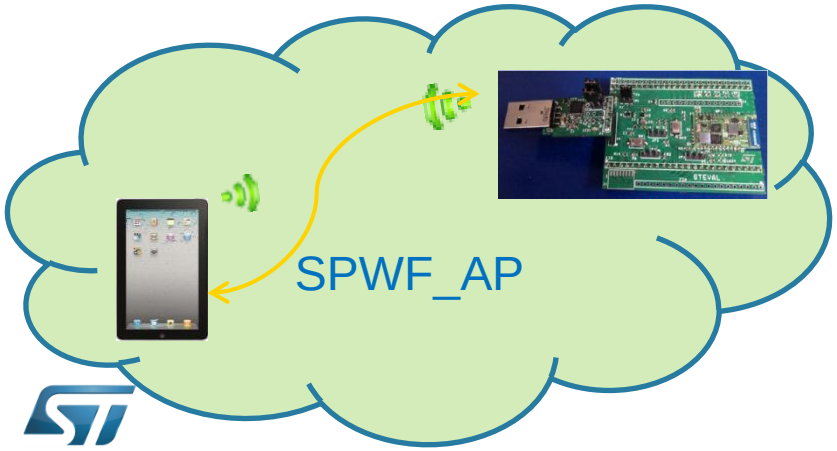


- Open your web browser

- In the address bar, type captiveportal.net




***Tip: If the AP domain name is not quickly opened, it's suggested to turn off an eventual proxy server (check the connection settings or browser preferences)***

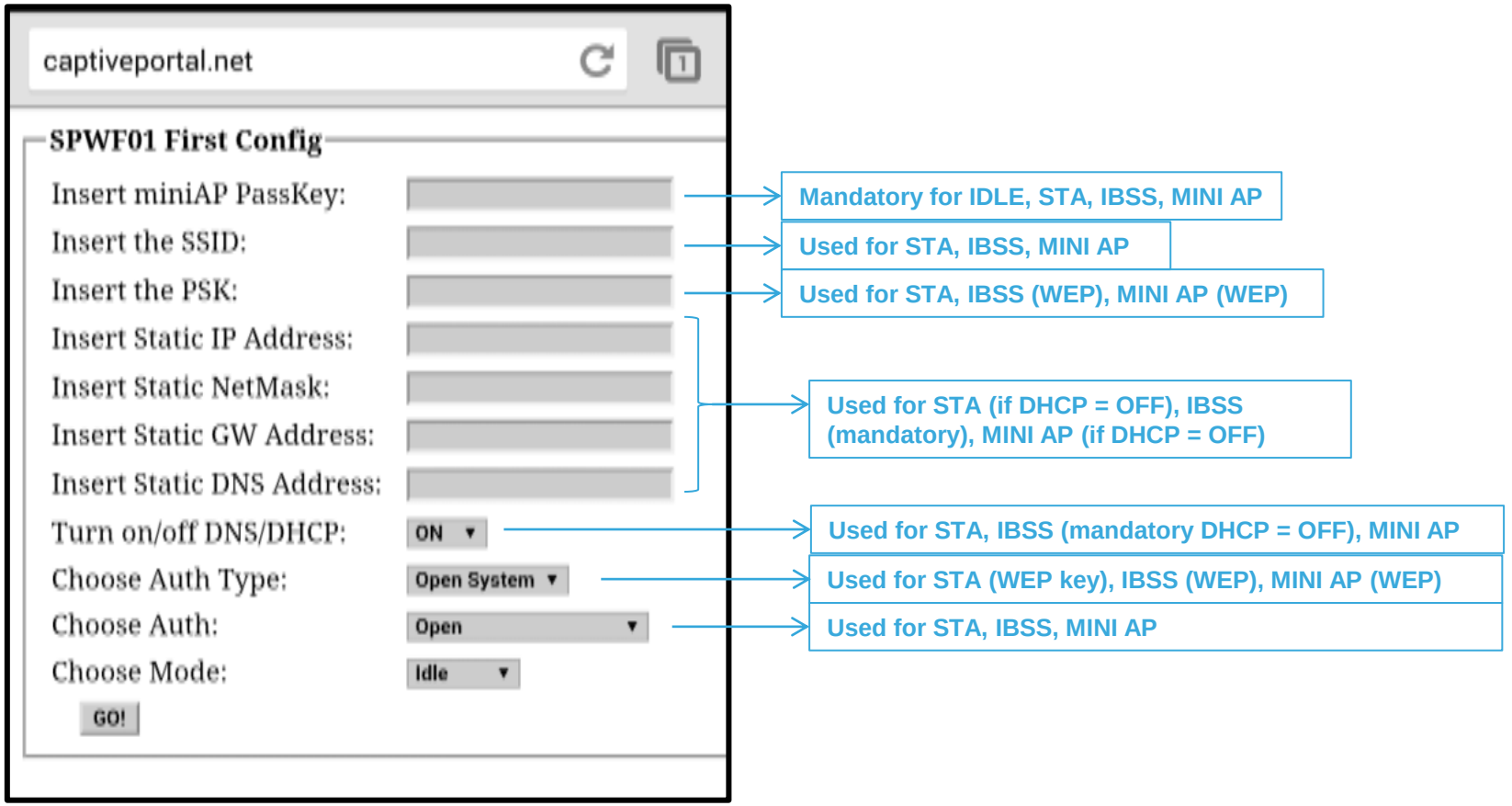



***Tip: The Mini AP domain can be set using the variable ip_apdomainname.***
***The default value is "captiveportal.net".***

# Lab 8: First Set Page



- The first set page allows to configure the module in IDLE mode, STATION mode, IBSS mode and MINI AP mode.



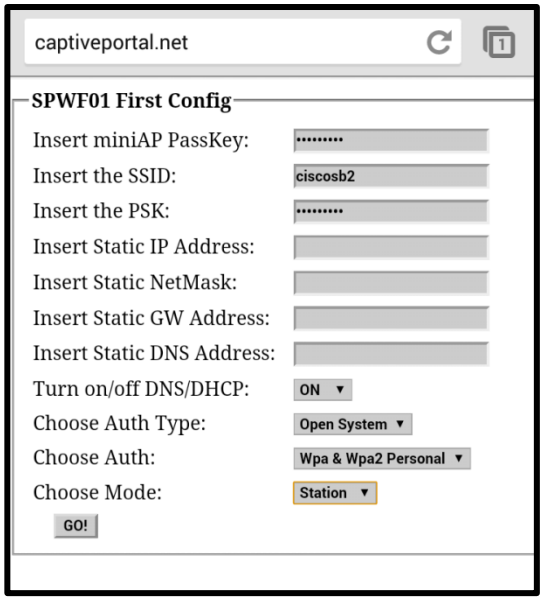
# Lab 8: Mini AP mode



- Set all the parameters required in order to enable the connection between the module and a generic AP

  i.e. AP configured in WPA/WPA2 mode

  - MiniAP PassKey: anonymous (by default)
  - SSID of the access point
  - Password of the access point
  - Authentication type of the access point
  - Use mode of the module



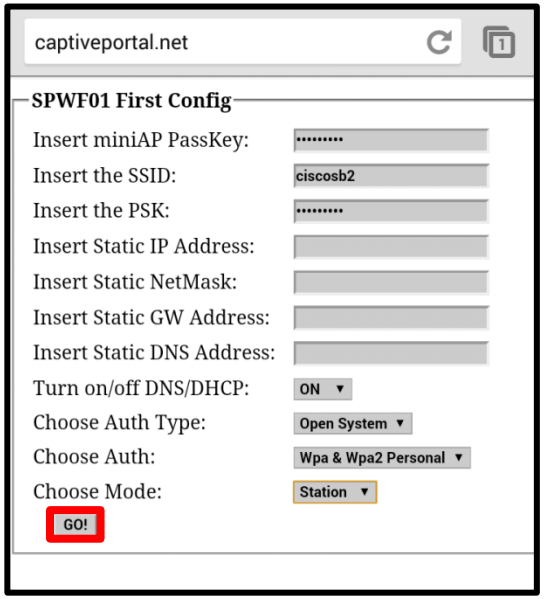

Note: It's suggested to configure SSID and password without special characters. If it needs to configure SSID and password containing special chars, bypass the "First Config" page and directly insert the whole URL in the browser URL bar. Please note that also using this trick, some special chars will not be correctly sent (it may depend on the web browser used).
For example if the password is **{helloworld}**, use directly the following URL:
**http://captiveportal.net/firstset.cgi?KEY=anonymous&SSID=ciscosb2&PWD={helloworld}&Auth=2&Mode=1&confirm=GO**

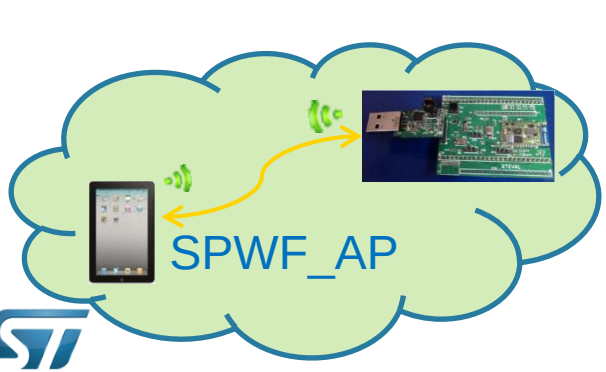
***Tip: The Mini AP PassKey can be set using the variable "user_desc".***
***The default value is "anonymous".***

# Lab 8: Mini AP mode

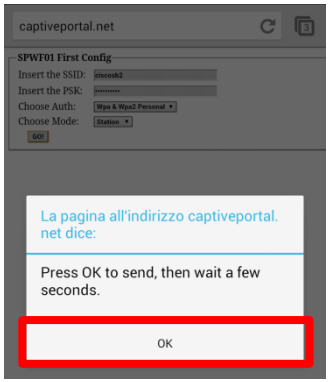


- Click on GO! button and then send the parameters confirming with OK





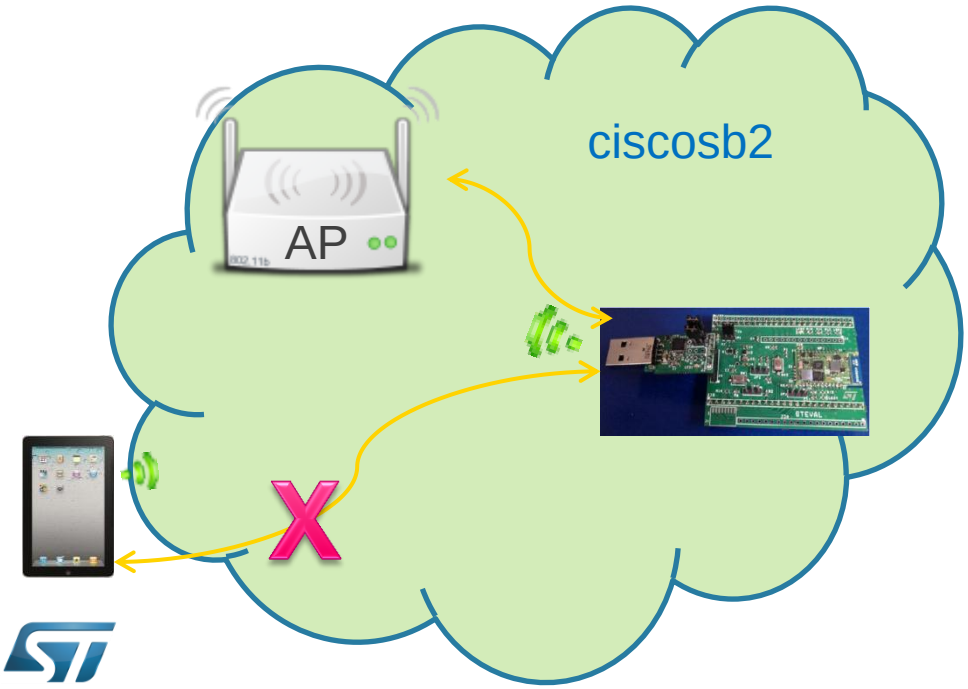



life.augmented

# Lab 8: Mini AP mode



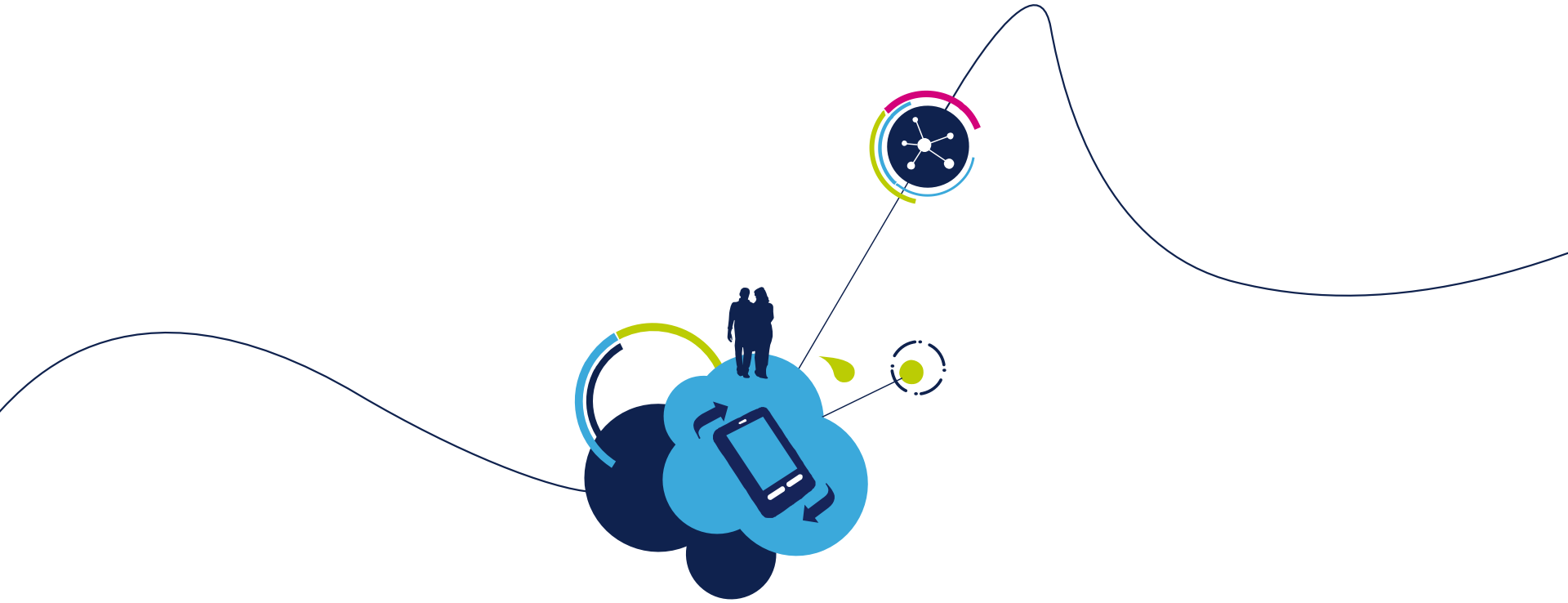
- The module will receive the parameters and will automatically connect to the access point required




*Tera Term output*

**All**

```
+WIND:57:Received SSID is ciscosb2
+WIND:57:Received PWD is **********
+WIND:57:Received Auth mode is 2
+WIND:57:Received Mode is 1
+WIND:2:RESET
+WIND:1:Poweron (******-*******-SPWF01S)
+WIND:13:ST IWM: Copyright (c) 2012-2013 STmicroelectronics
+WIND:3:Watchdog Running
+WIND:0:Console active
+WIND:46:WPA: Crunching PSK...
+WIND:32:WiFi Hardware Started
+WIND:21:WiFi Scanning
+WIND:35:WiFi Scan Complete (0x0)
+WIND:19:WiFi Join: 02:62:1F:51:8F:0B
+WIND:25:WiFi Association with 'ciscosb2' successful
+WIND:51:WPA Handshake Complete
+WIND:24:WiFi Up: 192.168.1.106
```



# Proceed to the next LAB!

# Lab 9: Socket interface - MiniAP mode



- Objective
  - Create a direct connection between the module and an end device
  - Socket interface used in MiniAP mode

- Prerequisites
  - Work alone

# Lab 9: Configuring the module in MiniAP mode



The Mini AP mode (available starting from the ***SPWF01S-131115-de4568d-RELEASE***) also allows to use the socket interface. The procedure is the same as in Lab 7.

In order to set the module in Mini AP mode, the following parameters are needed

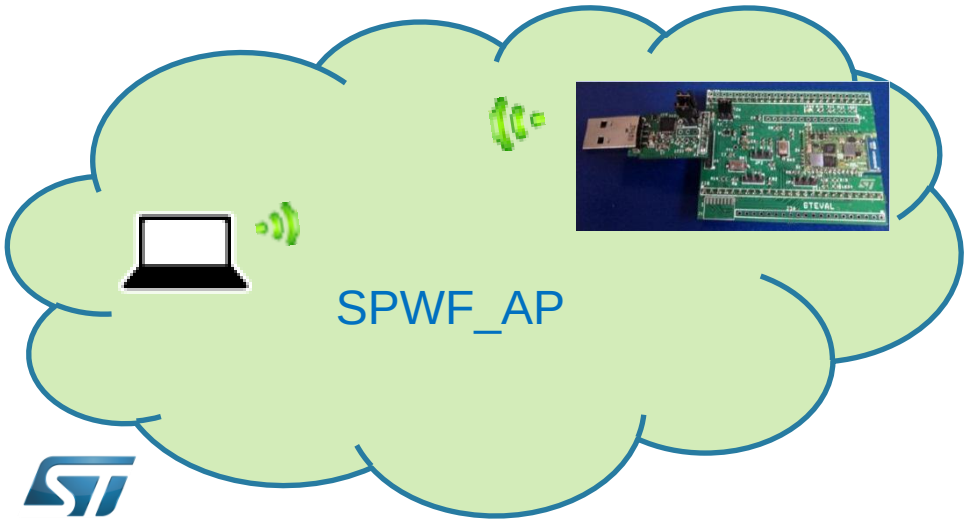
- Set the SSID
  - Type **AT+S.SSIDTXT=SPWF_AP**
- Set the network privacy mode
  - Type **AT+S.SCFG=wifi_priv_mode,0**
- Set the network mode (1 = STA, 2 = IBSS, 3 = MiniAP)
  - Type **AT+S.SCFG=wifi_mode,3**

*Tera Term output*

# Lab 9: Configuring the module in MiniAP mode



- Save the settings on the flash memory and reset the module
  - Type **AT&W**
  - Type **AT+CFUN=1**




*Tera Term output*

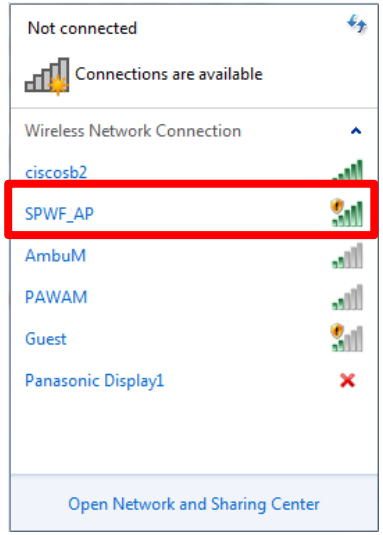
**All**

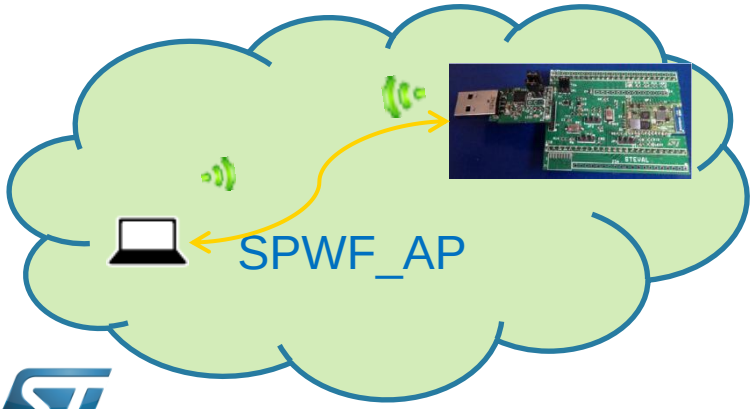
```
+WIND:0:Console active
+WIND:32:WiFi Hardware Started
+WIND:26:WiFi Started AP
+WIND:24:WiFi Up: 172.18.151.1
```

# Lab 9: Mini AP mode



- Connect your PC to the SPWF_AP network
  - Find the SPWF_AP network and connect the end device to the module




*Tera Term output*

**All**

```
+WIND:28:90:18:7C:96:0D:0B Associated
+WIND:29:DHCP reply for 172.18.151.2/90:18:7C:96:0D:0B to 172.18.151.2
+WIND:29:DHCP reply for 172.18.151.2/90:18:7C:96:0D:0B to 172.18.151.2
```

# Lab 9: Open TCP/UDP connection



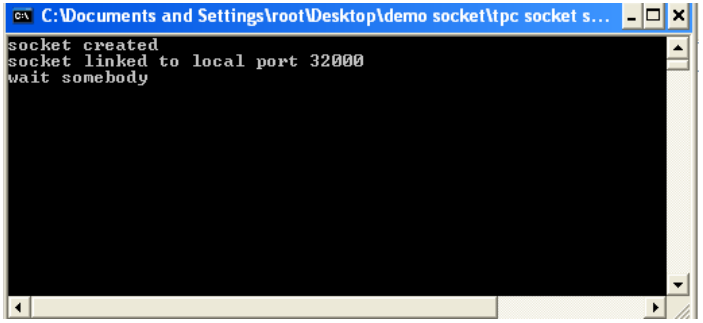
As described in the Lab 7, the SOCKON command allows to open a TCP/UDP connection to a specified host.

- Sintax
  - AT+S.SOCKON=<hostname>,<port>,<protocol>,ind
- Configuration parameters
  - <hostname>: target host. DNS resolvable name or IP address
  - <port>: TCP/UDP socket port
  - <protocol>: ***t*** for TCP socket, ***u*** for UDP socket
  - ind: indicate when data has arrived (optional). Up to 4 consecutive “Pending data” message (without any AT+S.SOCKR) are guaranteed. It is suggested to empty the buffer using the **AT+S.SOCKR** command and to avoid exceeding 4 indications.

# Lab 9: Open TCP connection



- Open the TCP socket server (disable the firewall to properly run it)
  - Folder ../hands_on_wifi/tcp socket server/server.exe
- The TCP server listens for incoming connections on the port 32000. It sends back all data received.

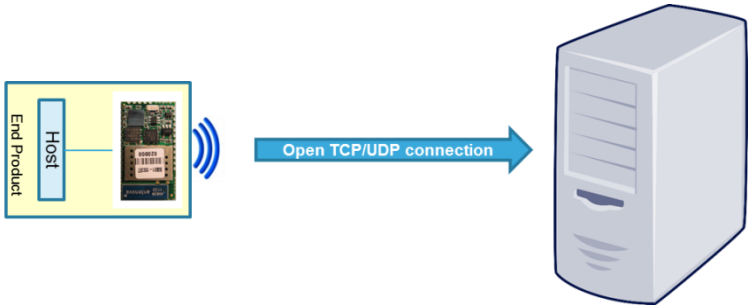


# Lab 9: Open TCP connection

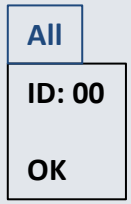


- Type **AT+S.SOCKON=172.18.151.2,32000,t,ind**

- The client and the server use the socket identifier (ID) displayed




*Tera Term output*



# Lab 9: Write data to socket



The SOCKW command allows to write data to the specified ID socket.

This command accepts data after the <cr> at the end of the command line.

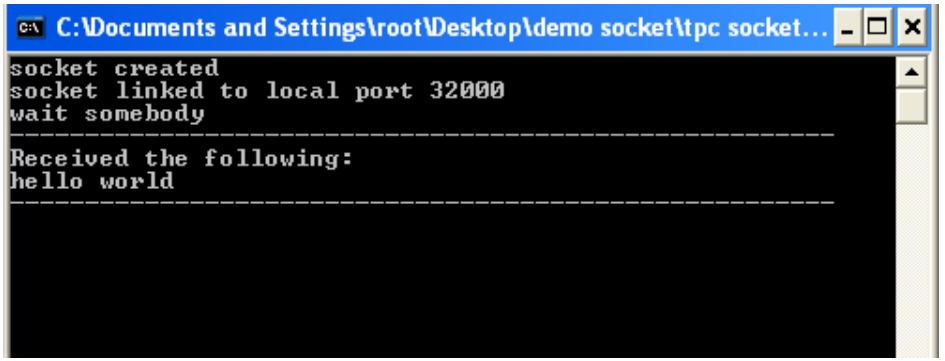
- Sintax
  - AT+S.SOCKW=<ID>,<len>
- Configuration parameters
  - <ID>: socket identifier
  - <len>: data length to send

# Lab 9: Write data to socket



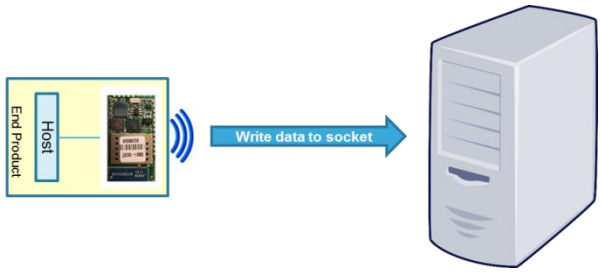
- Write data
  - Type **AT+S.SOCKW=00,13**

***Note: the module is waiting 13 bytes to be written to the socket. As soon as 13 bytes (or characters) are sent from the terminal, the module is going to write them to the socket and will be ready to receive new commands.***

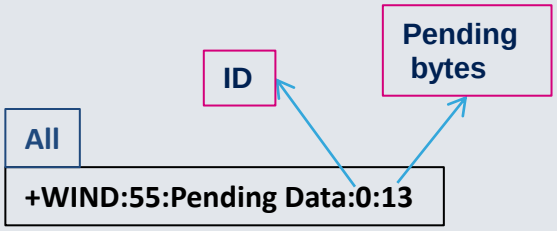



  - Type **hello world<CR>**

- SPWF shows that there are <pending data> and their amount




*Tera Term output*



# Lab 9: Read data from socket



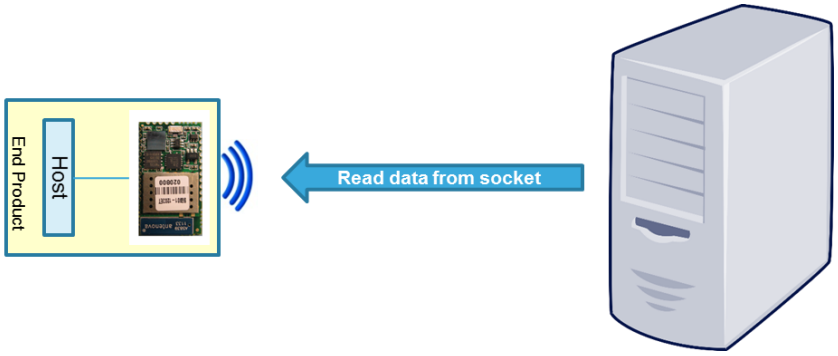
The SOCKR command allows to read data from socket.

- Sintax
  - AT+S.SOCKR=<ID>,<len>
- Configuration parameters
  - <ID>: socket identifier
  - <len>: data length to read

# Lab 9: Read data from socket



- Read data
  - Type **AT+S.SOCKR=00,13**




*Tera Term output*

# Lab 9: Close socket



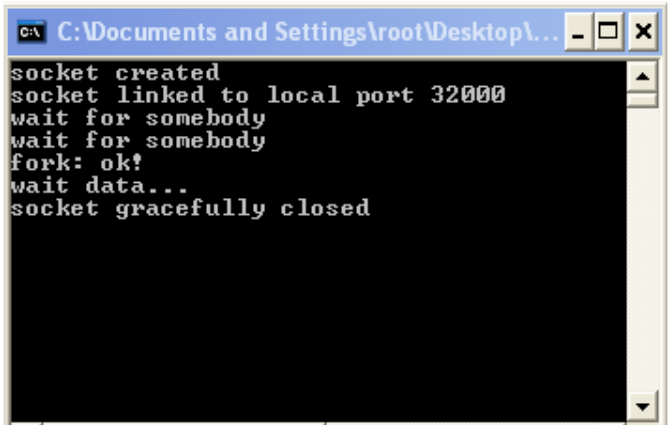
The SOCKC command allows to close socket.

- Sintax
  - AT+S.SOCKC=<ID>
- Configuration parameters
  - <ID>: socket identifier

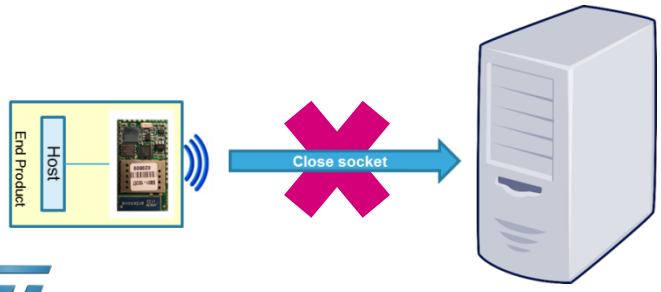
# Lab 9: Close socket

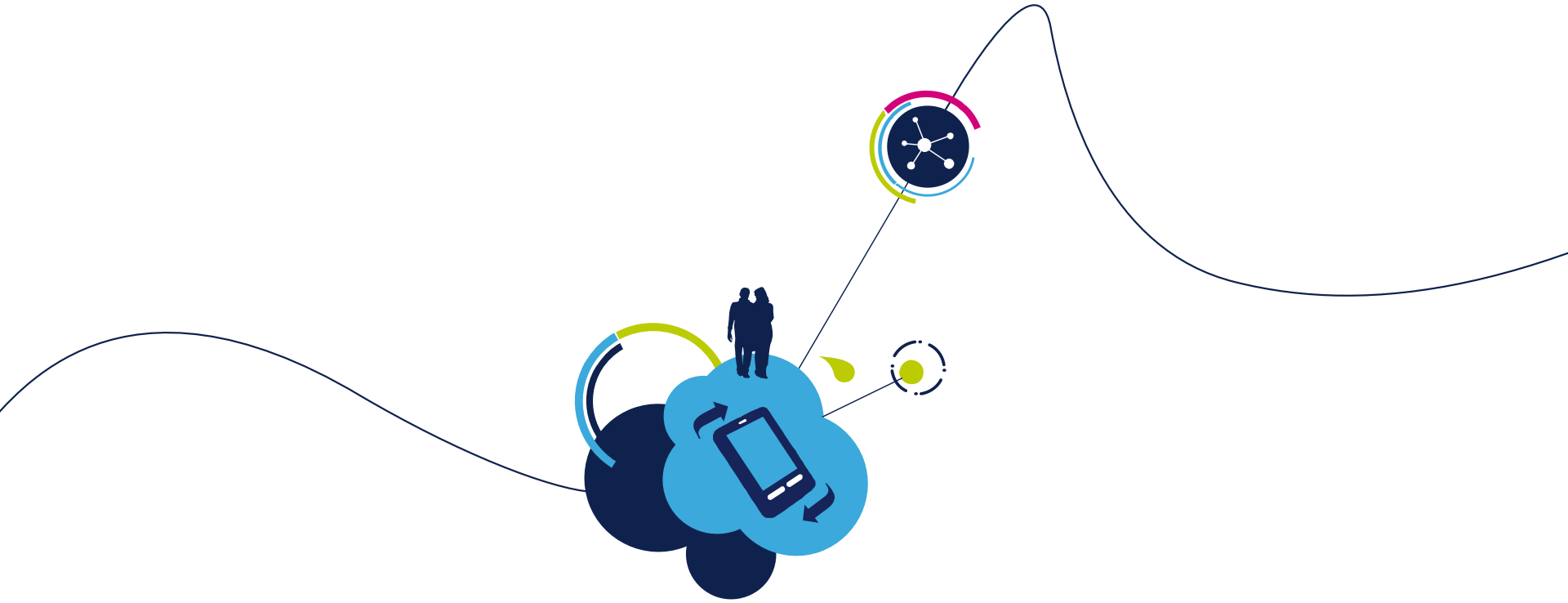


- Close socket
  - Type **AT+S.SOCKC=00**




*Tera Term output*



# Proceed to the next LAB!

# Lab 10: Web server - MiniAP mode



- Objective
  - Create a direct connection between the module and an end device
  - Web server used in MiniAP mode

- Prerequisites
  - Work alone

# Lab 10: Configuring the module in MiniAP mode



The Mini AP mode (available starting from the ***SPWF01S-131115-de4568d-RELEASE***) also allows to use the module file system (both RAM and Int/Ext FLASH).

In order to set the module in Mini AP mode, the following parameters are needed

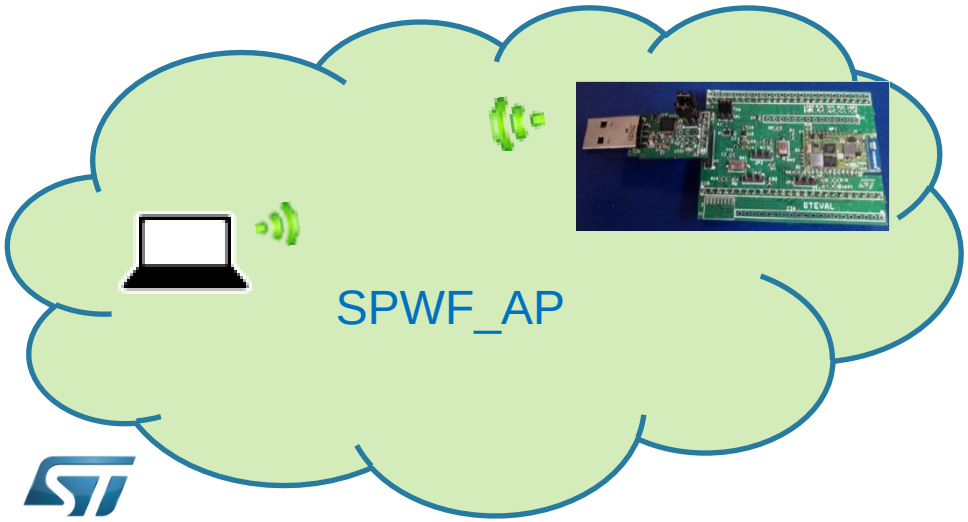
- Set the SSID
  - Type **AT+S.SSIDTXT=SPWF_AP**
- Set the network privacy mode
  - Type **AT+S.SCFG=wifi_priv_mode,0**
- Set the network mode (1 = STA, 2 = IBSS, 3 = MiniAP)
  - Type **AT+S.SCFG=wifi_mode,3**

*Tera Term output*

# Lab 10: Configuring the module in MiniAP mode



- Save the settings on the flash memory and reset the module
  - Type **AT&W**
  - Type **AT+CFUN=1**




*Tera Term output*

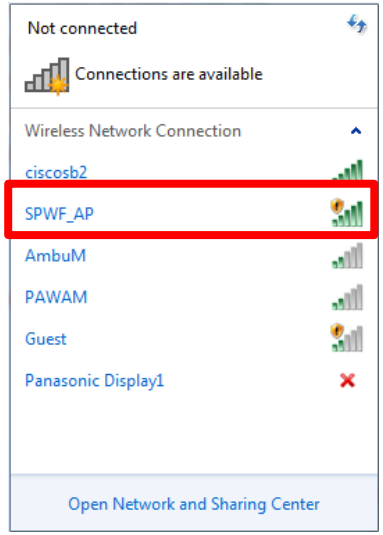
**All**

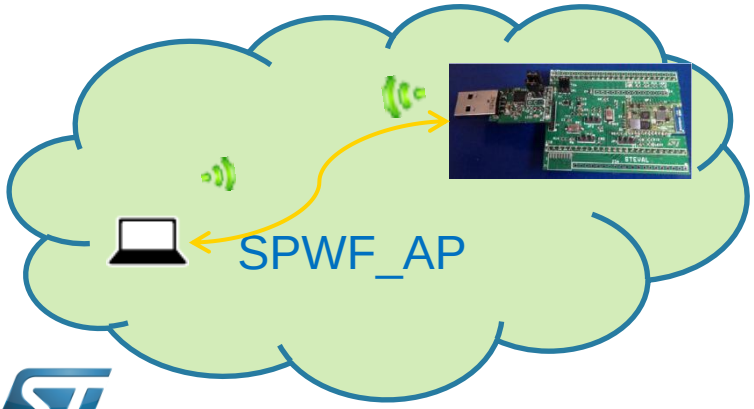
```
+WIND:0:Console active
+WIND:32:WiFi Hardware Started
+WIND:26:WiFi Started AP
+WIND:24:WiFi Up: 172.18.151.1
```

# Lab 10: Mini AP mode



- Connect your PC to the SPWF_AP network
  - Find the SPWF_AP network and connect the end device to the module




*Tera Term output*

**All**

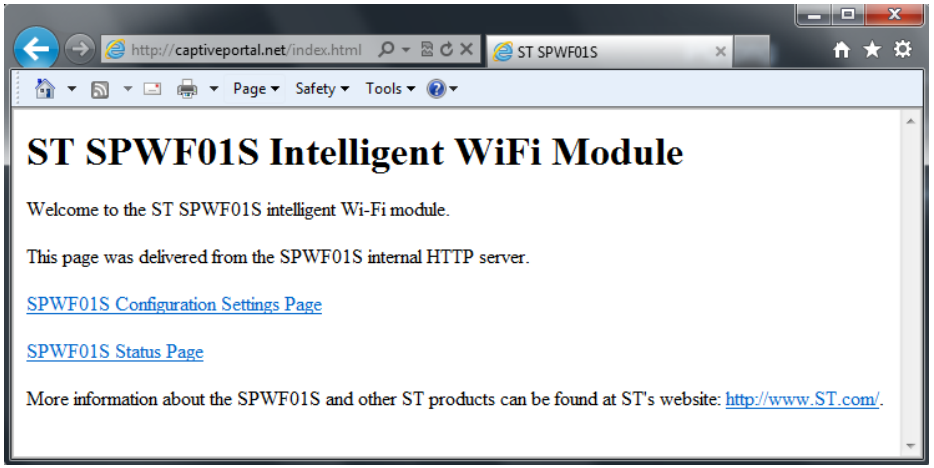
```
+WIND:28:90:18:7C:96:0D:0B Associated
+WIND:29:DHCP reply for 172.18.151.2/90:18:7C:96:0D:0B to 172.18.151.2
+WIND:29:DHCP reply for 172.18.151.2/90:18:7C:96:0D:0B to 172.18.151.2
```

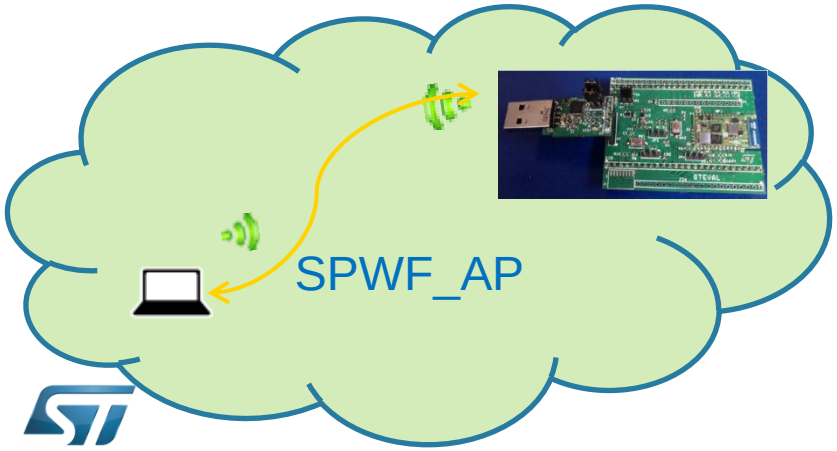
life.augmented

# Lab 10: Access to Web server - Mini AP mode



- Open your web browser
- In the address bar, type captiveportal.net/index.html
- The SPWF01S homepage will be displayed



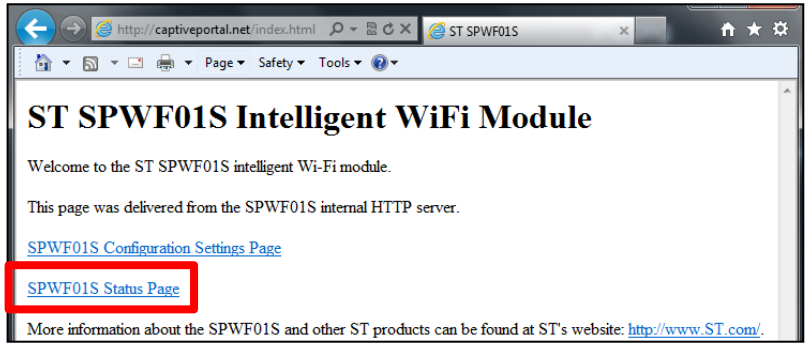




***Tip: The Mini AP domain can be set using the variable ip_apdomainname.***
***The default value is "captiveportal.net".***

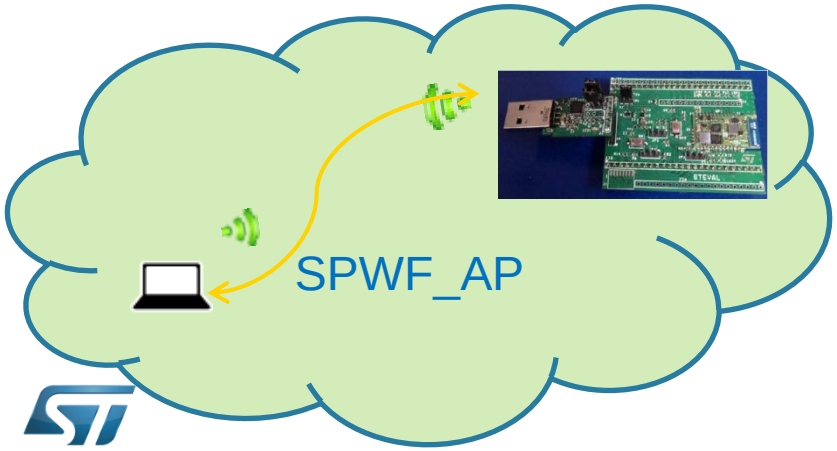
# Lab 10: Access to Web server - Mini AP mode



- Open the SPWF01S Device Status
  - click on the highlighted link

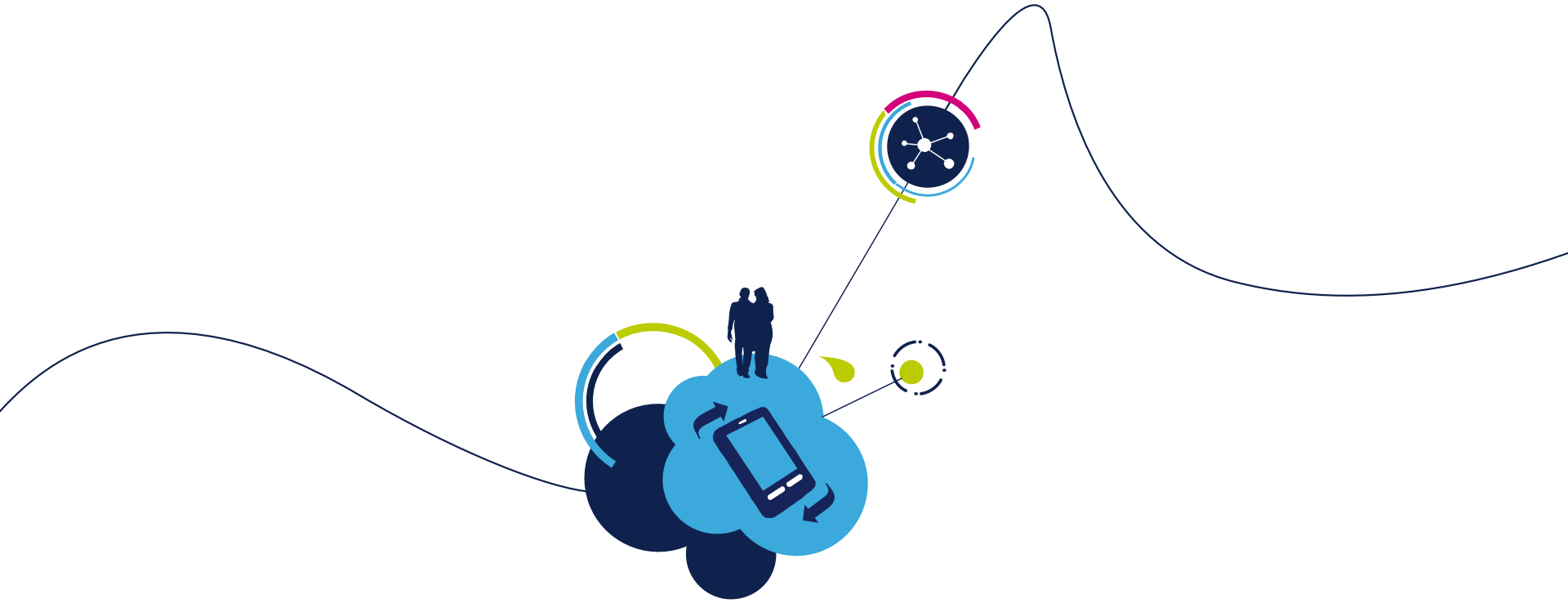



- The SPWF01S Device Status will be displayed


***Tip: The Mini AP domain can be set using the variable ip_apdomainname.***
***The default value is "captiveportal.net".***

life.augmented



# Proceed to the next LAB!

# Lab 11: Remotely control - GPIOs



- Objective
  - write remotely a GPIO
  - configure remotely a GPIO
  - read remotely a GPIO
- Prerequisites
  - module connected to the AP or
    module configured in MiniAP mode

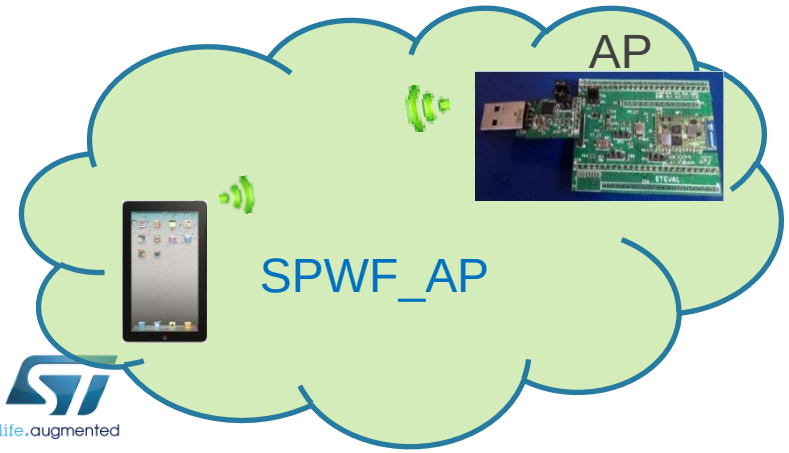
# Lab 11: Remotely control - GPIO

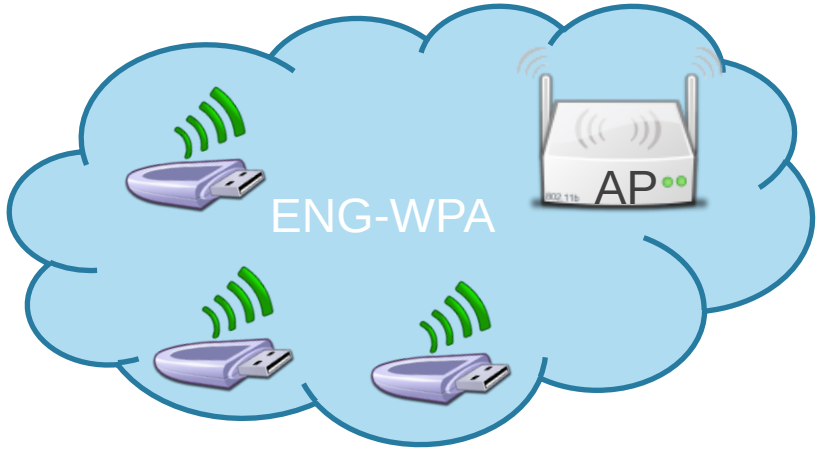


This feature allows to remotely write, configure and read a GPIO.

The module should be connected to the AP (as shown in Lab 3) or should be configured in Mini AP mode (as shown in Lab 8).

The external client have to open the **remote.html** page stored in the module.



# Lab 11: Joining a network



In order to be connected to an available Wifi network, the AP parameters setting is needed.

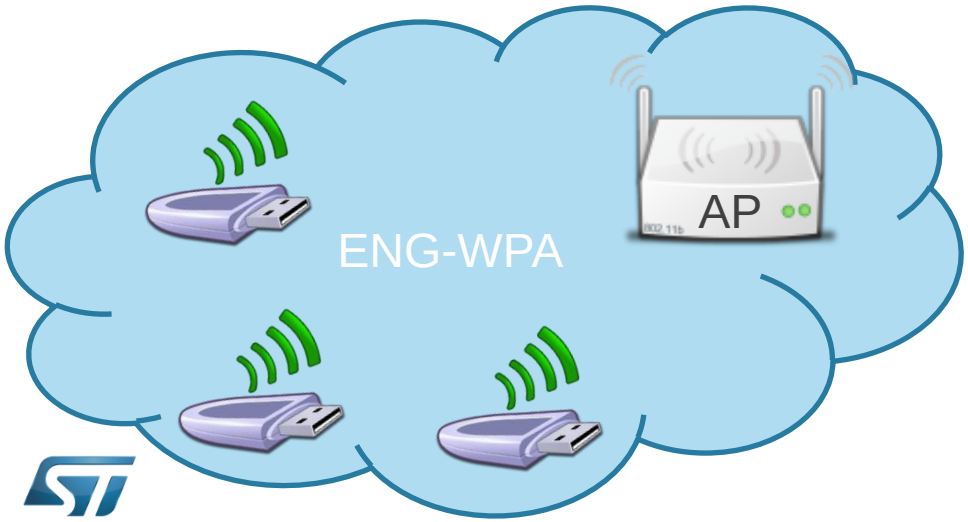
- Set the SSID
  - Type **AT+S.SSIDTXT=ENG-WPA**
- Set the password
  - Type **AT+S.SCFG=wifi_wpa_psk_text,helloworld**
- Set the network privacy mode (0=none, 1=WEP, 2=WPA-Personal (TKIP/AES) or WPA2-Personal (TKIP/AES))
  - Type **AT+S.SCFG=wifi_priv_mode,2**
- Set the network mode (1 = STA, 2 = IBSS, 3 = MiniAP)
  - Type **AT+S.SCFG=wifi_mode,1**

*Tera Term output*

# Lab 11: Joining a network



- Save the settings on the flash memory and reset the module
  - Type **AT&W**
  - Type **AT+CFUN=1**




*Tera Term output*

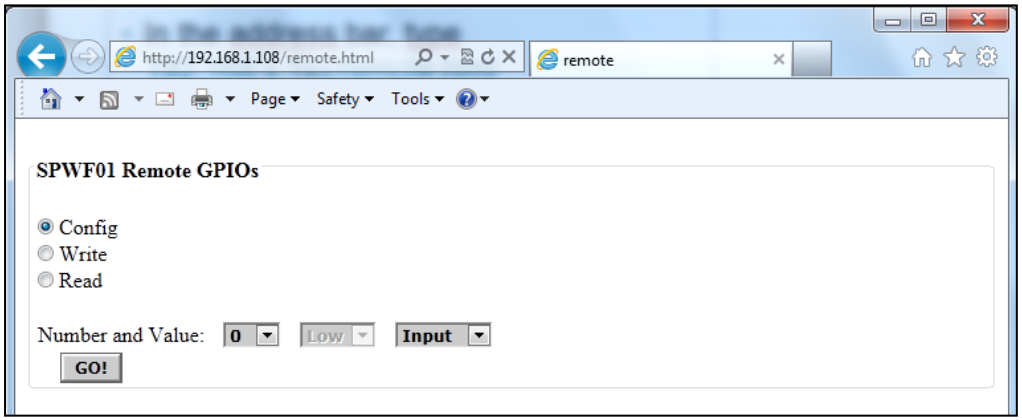
All

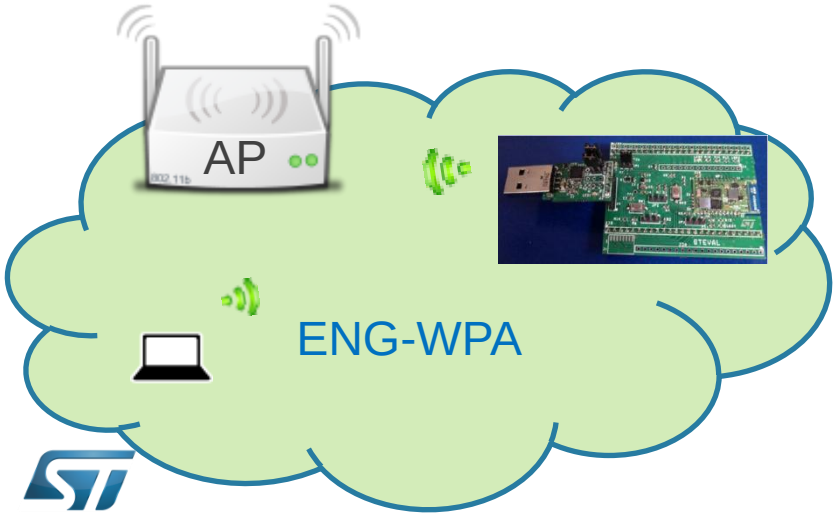
```
+WIND:0:Console active
+WIND:46:WPA: Crunching PSK...
+WIND:32:WiFi Hardware Started
+WIND:21:WiFi Scanning
+WIND:35:WiFi Scan Complete (0x0)
+WIND:19:WiFi Join: 14:D6:4D:24:36:00
+WIND:25:WiFi Association with 'ENG-WPA' successful
+WIND:51:WPA Handshake Complete
+WIND:24:WiFi Up: 192.168.0.1xx
```

# Lab 11: Remotely control in MiniAP mode - GPIO



- Open your web browser
- In the address bar, type 192.168.x.1xx/remote.html



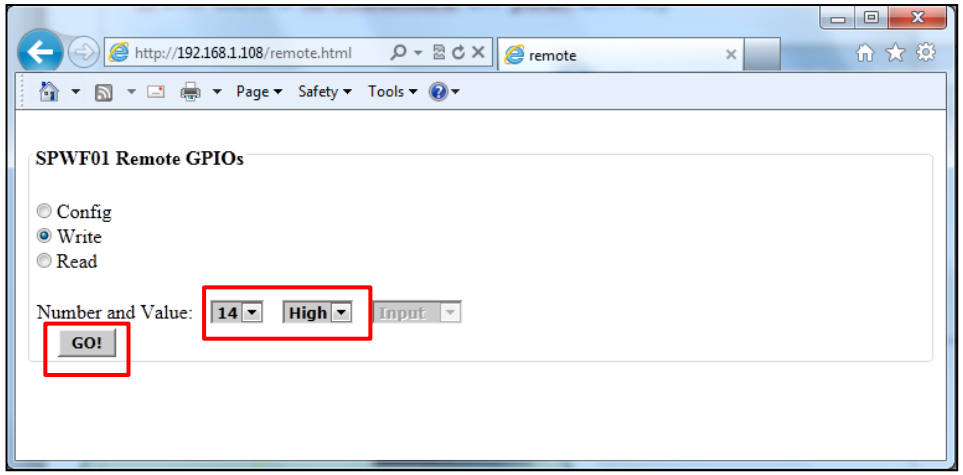
# Lab 11: Remotely control in MiniAP mode - GPIO

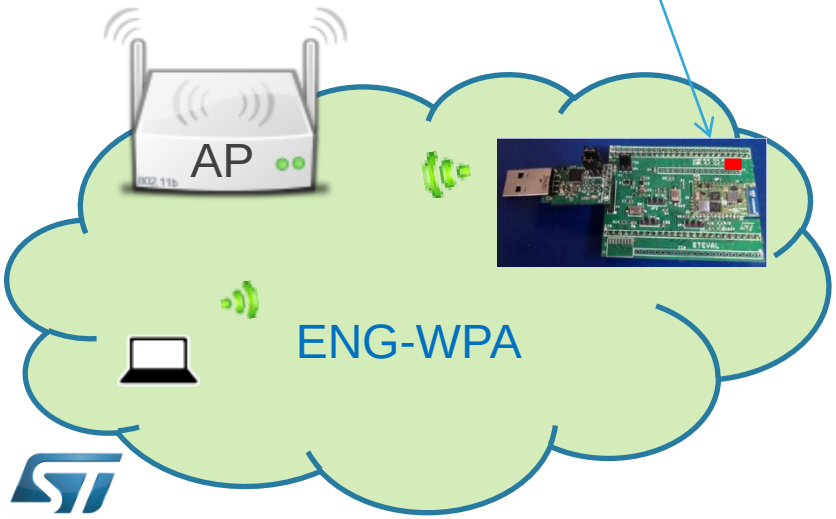


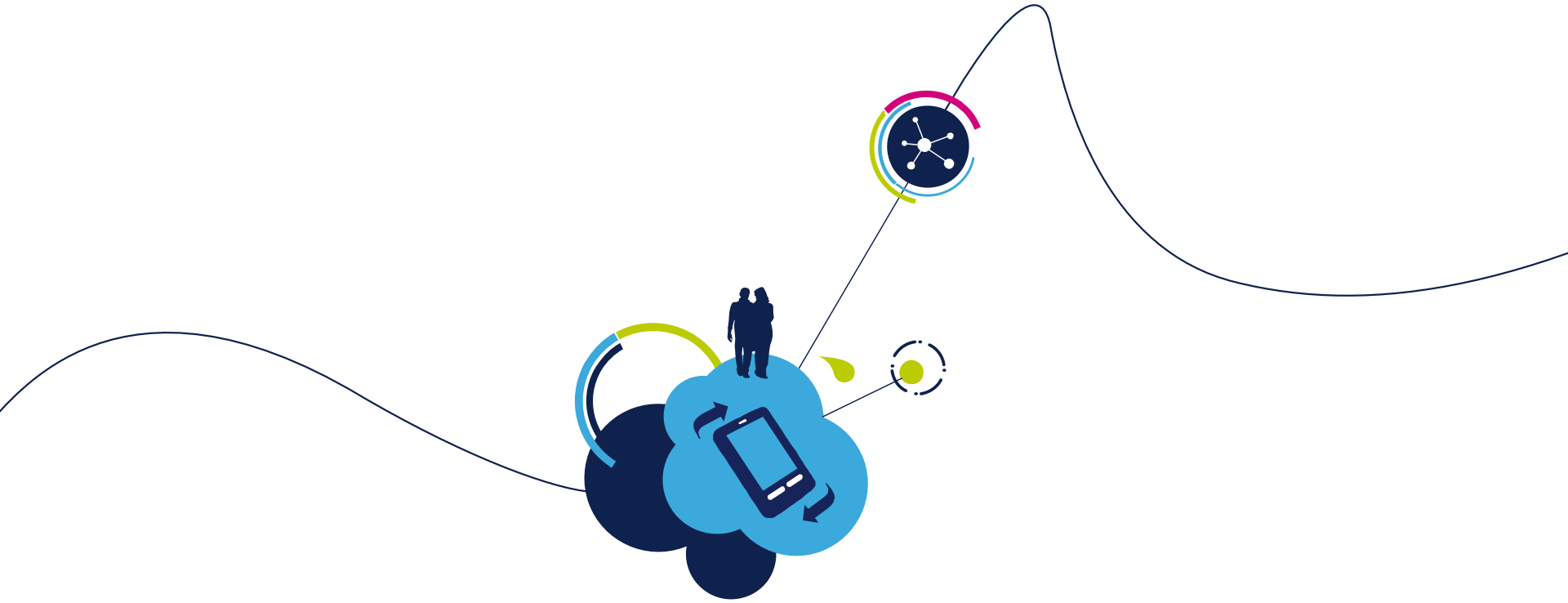
- Try to write the GPIO14 connected to the LED3 (check if the LED3 is mounted on your EVAL BOARD, otherwise can be used the GPIO13)

- Click on the «GO!» button

and LED3 will switch on





# Proceed to the next LAB!

# Lab 12: Input demo



- Objective
  - Send a message from the server (wifi module) to an external client connected to the same network
- Prerequisites
  - module connected to the AP or module configured in MiniAP mode

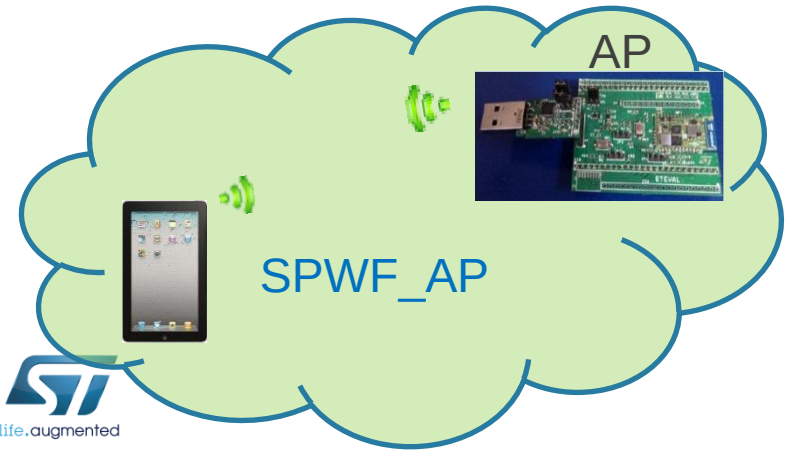
# Lab 12: Input demo

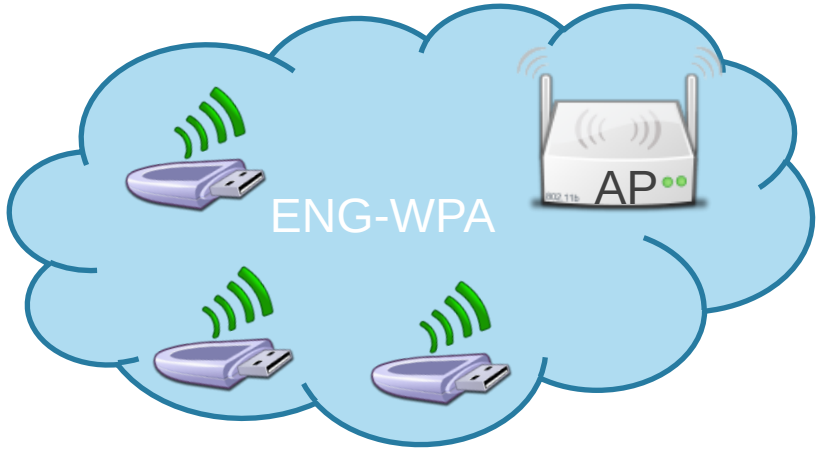


The module provides some DEMOs to show the interaction between the module and an external client connected to the same network.

The module should be connected to the AP (as shown in Lab 3) or should be configured in Mini AP mode (as shown in Lab 8).

In order to run this demo, the client have to open the **input_demo.shtml** page stored in the module.



# Lab 12: Joining a network



In order to be connected to an available Wifi network, the AP parameters setting is needed.

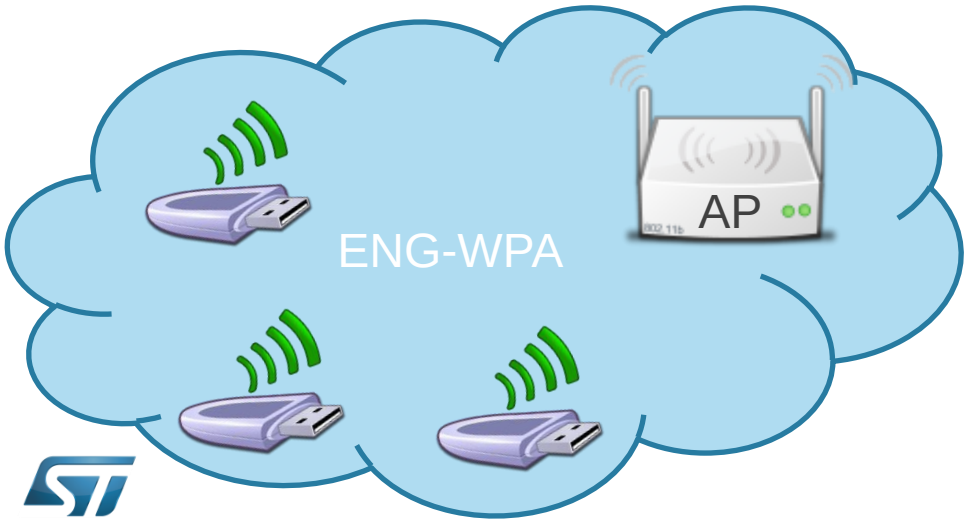
- Set the SSID
  - Type **AT+S.SSIDTXT=ENG-WPA**
- Set the password
  - Type **AT+S.SCFG=wifi_wpa_psk_text,helloworld**
- Set the network privacy mode (0=none, 1=WEP, 2=WPA-Personal (TKIP/AES) or WPA2-Personal (TKIP/AES))
  - Type **AT+S.SCFG=wifi_priv_mode,2**
- Set the network mode (1 = STA, 2 = IBSS, 3 = MiniAP)
  - Type **AT+S.SCFG=wifi_mode,1**

*Tera Term output*

# Lab 12: Joining a network



- Save the settings on the flash memory and reset the module
  - Type **AT&W**
  - Type **AT+CFUN=1**




*Tera Term output*

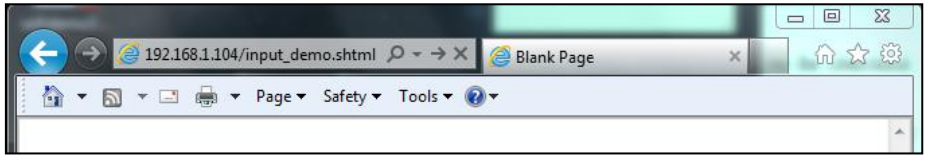
All

```
+WIND:0:Console active
+WIND:46:WPA: Crunching PSK...
+WIND:32:WiFi Hardware Started
+WIND:21:WiFi Scanning
+WIND:35:WiFi Scan Complete (0x0)
+WIND:19:WiFi Join: 14:D6:4D:24:36:00
+WIND:25:WiFi Association with 'ENG-WPA' successful
+WIND:51:WPA Handshake Complete
+WIND:24:WiFi Up: 192.168.0.1xx
```

# Lab 12: Input demo

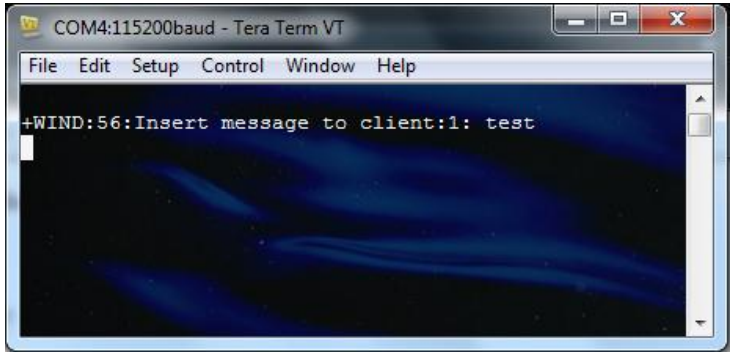


- Open your web browser
- In the address bar, type 192.168.x.1xx/input_demo.shtml




! VERY IMPORTANT:
Both client and server are pending (2 sec) after the page request. If no message is sent from the module, a timeout mechanism will be triggered

- Insert the message to the client i.e. type «test» (**suggested – copy the string and paste it in the terminal**) and then send a carriage return



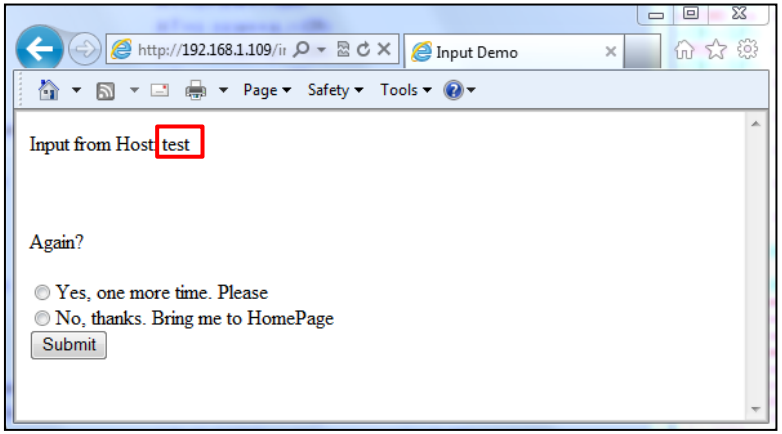

! The server shows the “+WIND:56 message”, and it will wait for a string to be inserted, and terminated by "[Cr]".

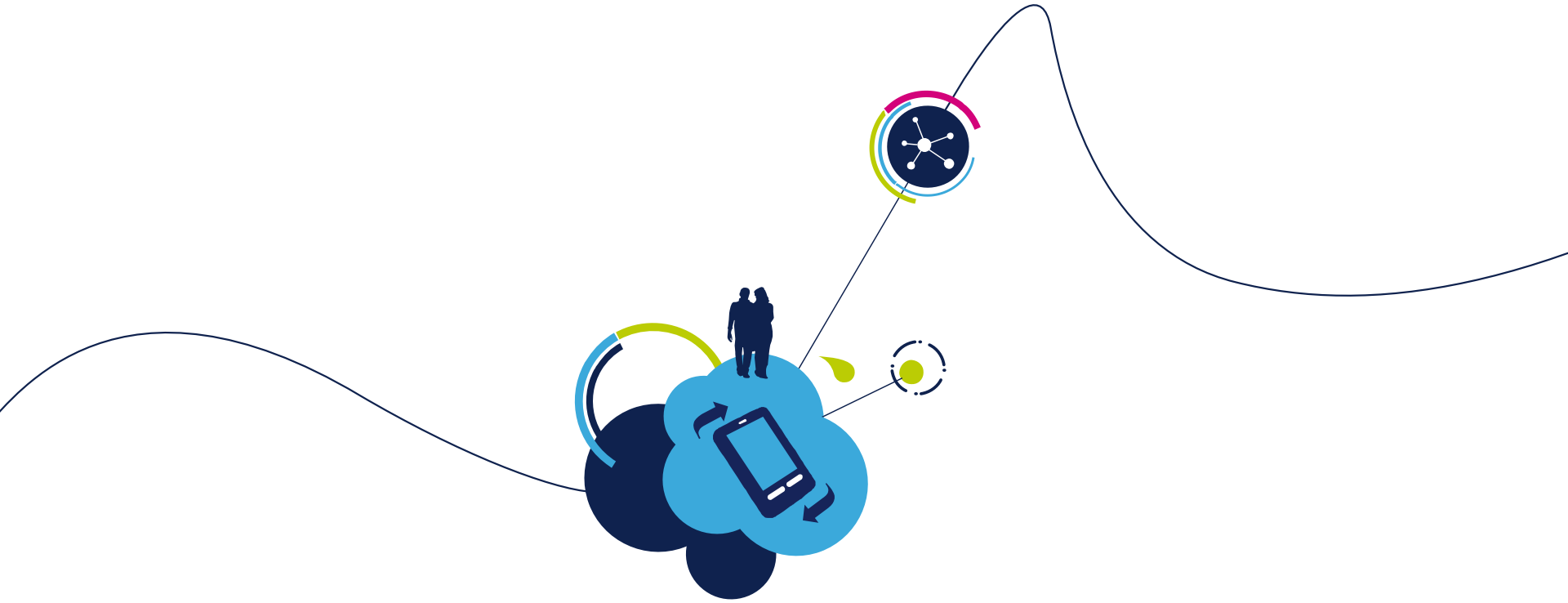
***“+WIND:56:Insert message to client:%d” → %d is the Nth input SSI into html page (please refer to CGI&SSI Application Notes for details)***

# Lab 12: Input demo



- The module receives the HTML page containing the string inserted server-side.
- Below this string, there are two buttons to try again the demo or to return to the homepage



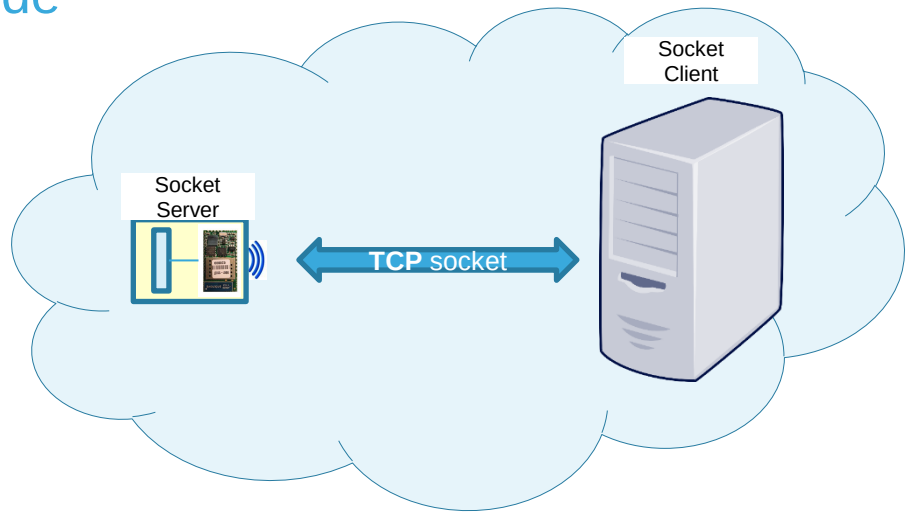


# Proceed to the next LAB!

# Lab 13: Socket Server functionality



- Objective
  - Configure a Server Listening Port
  - Open a TCP connection from a socket client to the module
  - Data mode/Command Mode usage
- Prerequisites
  - module connected to the AP or module configured in MiniAP mode
  - PC to be used as socket client



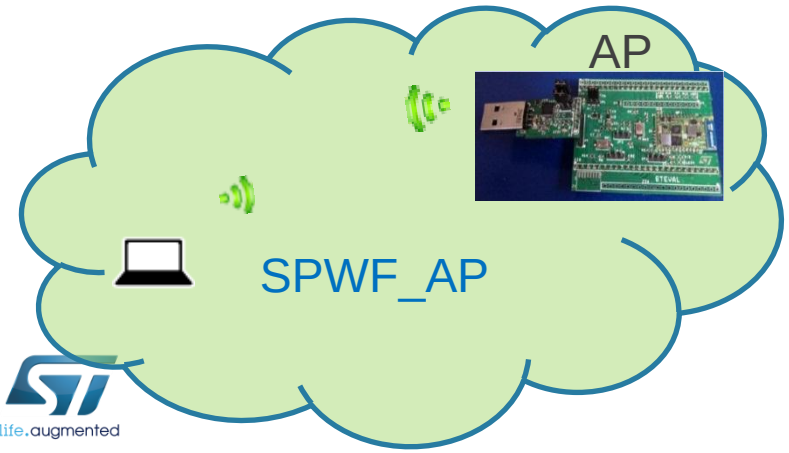
# Lab 13: Socket Server

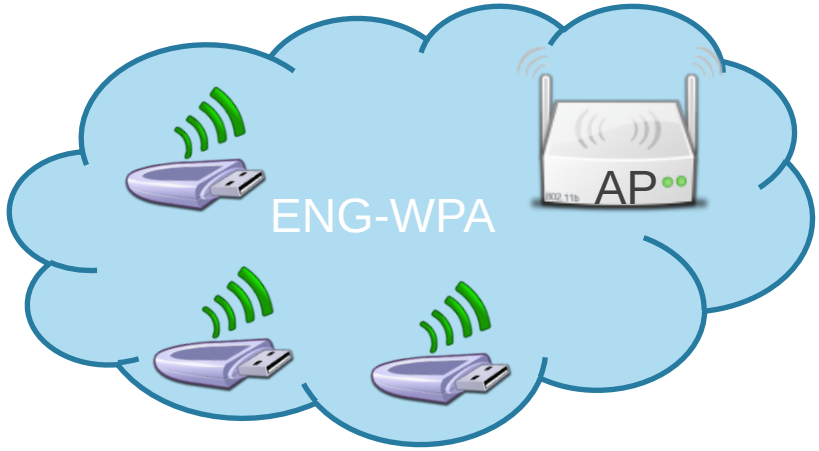


This feature allows to enable the socket server mode. The module can be able to listen for an incoming connection on the specified port.

The module should be connected to the AP (as shown in Lab 3) or should be configured in Mini AP mode (as shown in Lab 8).

Please refer to “Socket Server Application Notes” for more details.



# Lab 13: Joining a network



In order to be connected to an available Wifi network, the AP parameters setting is needed.

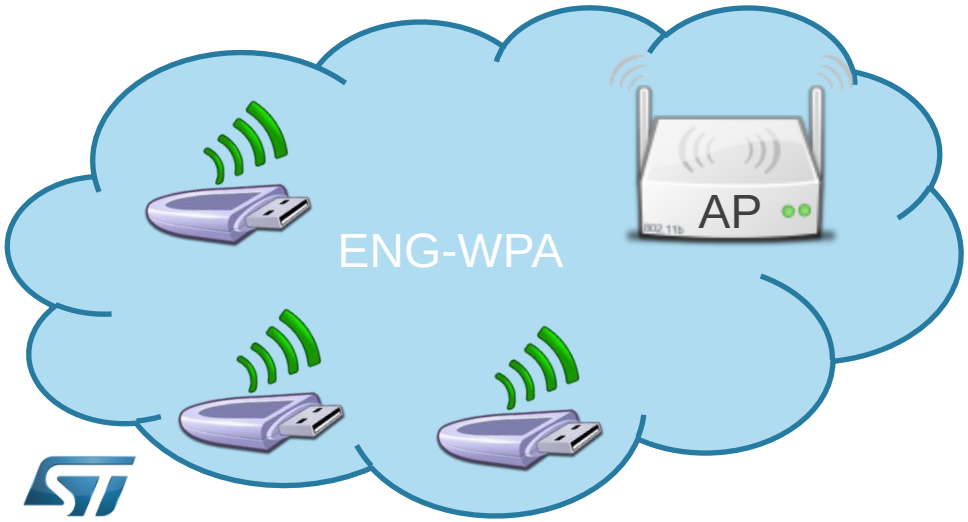
- Set the SSID
  - Type **AT+S.SSIDTXT=ENG-WPA**
- Set the password
  - Type **AT+S.SCFG=wifi_wpa_psk_text,helloworld**
- Set the network privacy mode (0=none, 1=WEP, 2=WPA-Personal (TKIP/AES) or WPA2-Personal (TKIP/AES))
  - Type **AT+S.SCFG=wifi_priv_mode,2**
- Set the network mode (1 = STA, 2 = IBSS, 3 = MiniAP)
  - Type **AT+S.SCFG=wifi_mode,1**

*Tera Term output*

# Lab 13: Joining a network



- Save the settings on the flash memory and reset the module
  - Type **AT&W**
  - Type **AT+CFUN=1**




*Tera Term output*

All

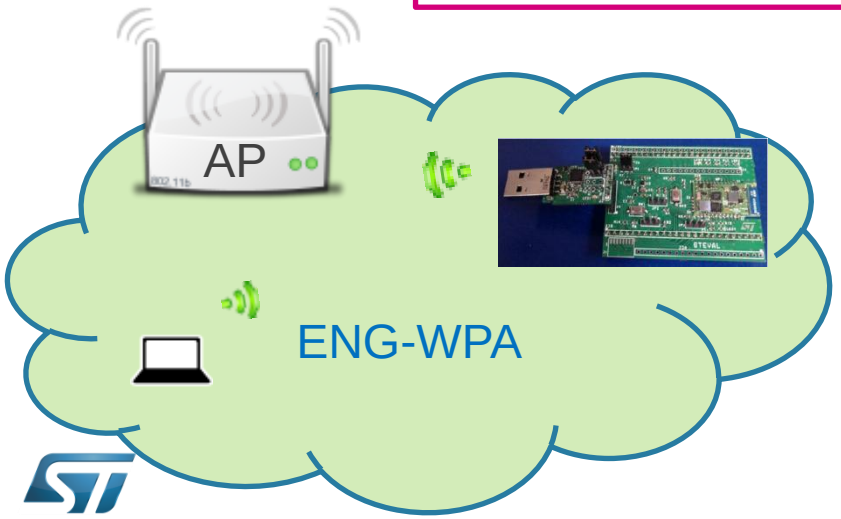
```
+WIND:0:Console active
+WIND:46:WPA: Crunching PSK...
+WIND:32:WiFi Hardware Started
+WIND:21:WiFi Scanning
+WIND:35:WiFi Scan Complete (0x0)
+WIND:19:WiFi Join: 14:D6:4D:24:36:00
+WIND:25:WiFi Association with 'ENG-WPA' successful
+WIND:51:WPA Handshake Complete
+WIND:24:WiFi Up: 192.168.0.1xx
```

# Lab 13: Socket Server



- Socket server: turn ON the TCP Socket Server (user must specify the server listening port)
  - Type **AT+S.SOCKD=32000** (**AT+SOCKD=32000,u** for UDP socket server)
- Check the status of the socket server
  - Type **AT+S.STS=ip_sockd_port**

***Tip: you can Turn OFF the Socket Server using a server listening port = 0 (type AT+S.SOCKD=0)***




*Tera Term output*

**All**

**OK**

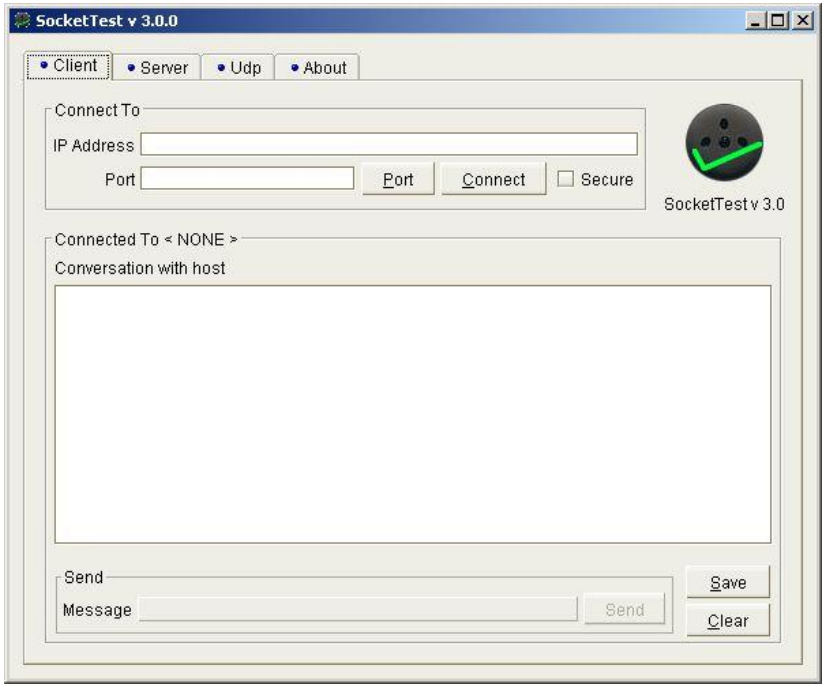
**# ip_sockd_port = 32000**

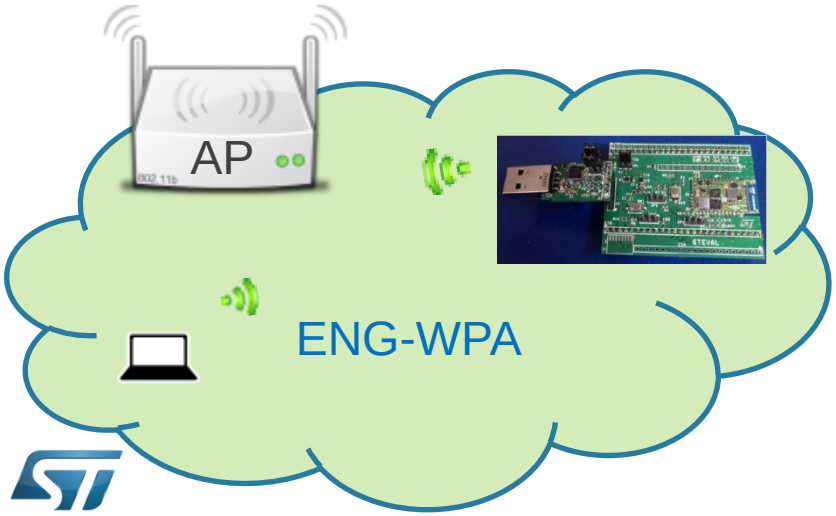
**OK**

# Lab 13: Socket Server



- Socket client: can be used a simple socket client in order to test the communication (SocketTest – available on http://sourceforge.net/projects/sockettest)

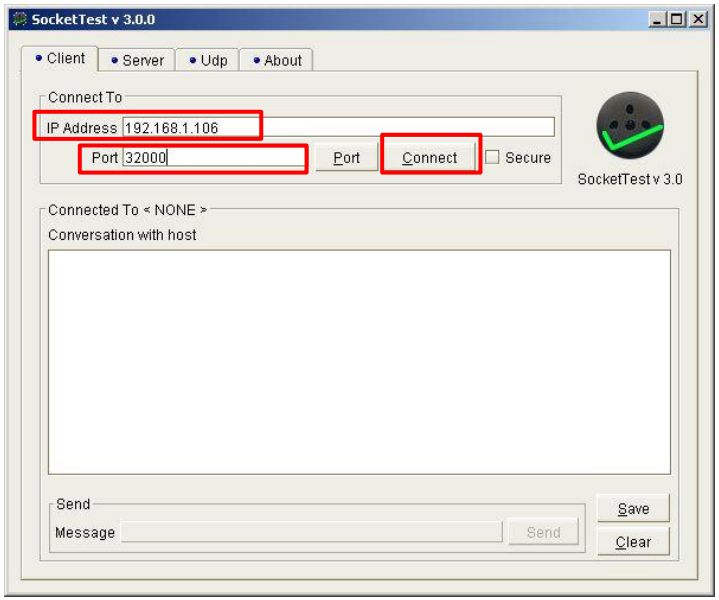


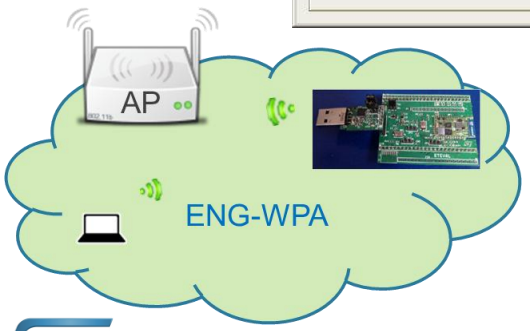



# Lab 13: Socket Server



- Socket client:
  - Insert the **module's IP Address** and the **port**
  - Click on **Connect** button




*Tera Term output*

**All**

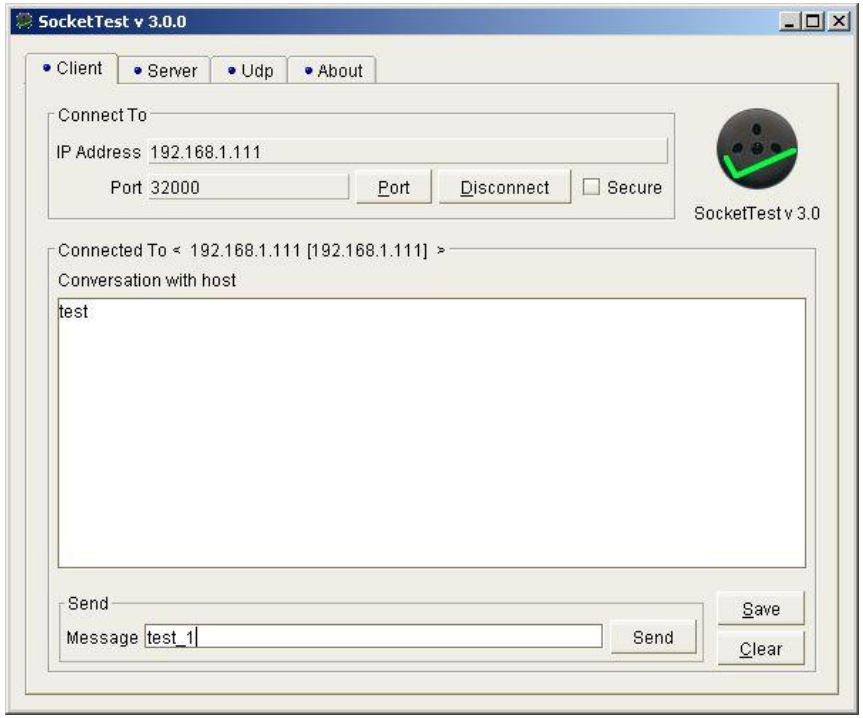
**+WIND:61:Incoming Socket Client:192.168.1.102**

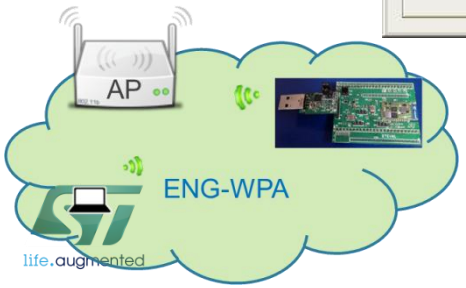
**+WIND:60:Now in Data Mode**

# Lab 13: Socket Server



- The module (socket server) enters in data mode and a bidirectional channel is created to allow exchanging data with the socket client
- Try to send and receive data from the module



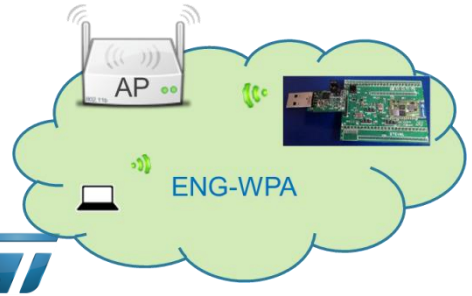

*Tera Term output*

# Lab 13: Socket Server – escape sequence



- From Data Mode to Command Mode
  - Configuration variable used: escape_seq (default is “at+s.”)
- Type **at+s.**
- Now it’s possible to send AT commands while the socket connection is still active

***Tip: During the Command Mode, you cannot send data to client, but can receive data from it. Every data chunk generates an URC (+WIND:64:Sockd Pending Data), with limit to 4. This is the max number of pending messages. From the 5th chunk on, messages are lost.***




*Tera Term output*

All

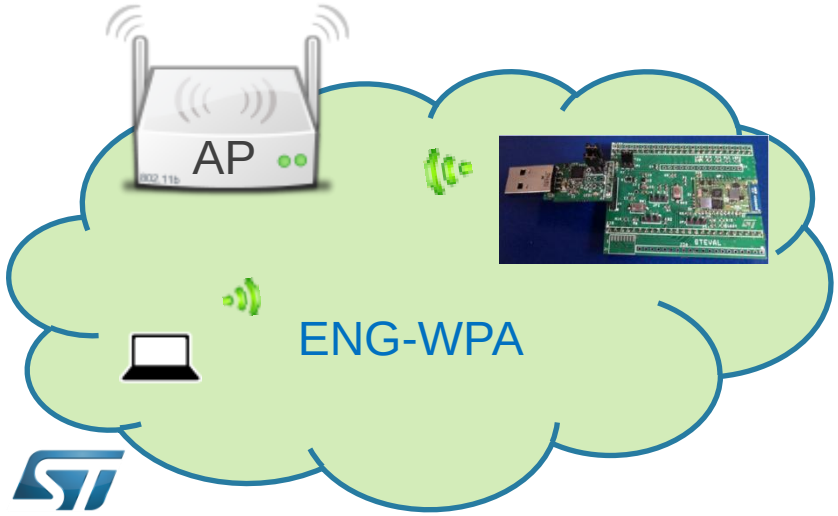
**+WIND:59:Back to Command Mode**

# Lab 13: Socket Server – return to data mode



- From Command Mode to Data Mode
  - While in Command Mode
  - Type AT+S.

- The module returns in data mode




*Tera Term output*

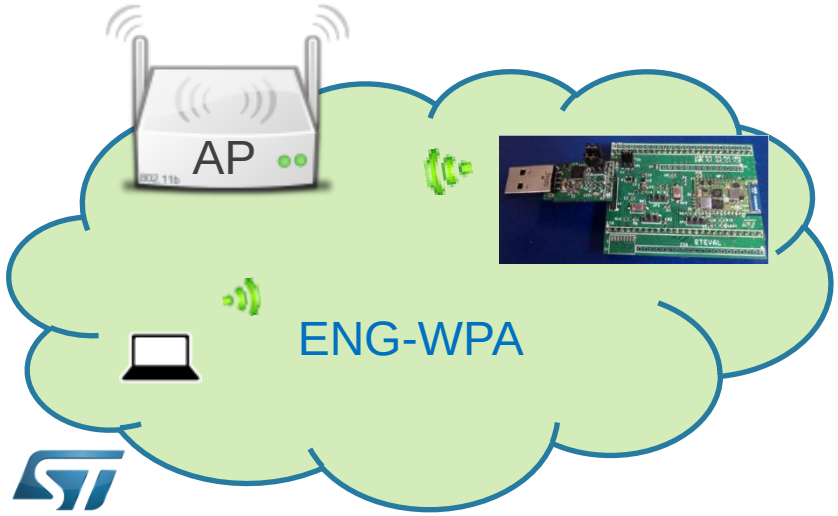
All

**+WIND:60:Now in Data Mode**

# Lab 13: Socket Server – turn off



- Turn OFF the Socket Server
  - Type **at+s. (to enter in command mode)**
  - Type **AT+S.SOCKD=0**

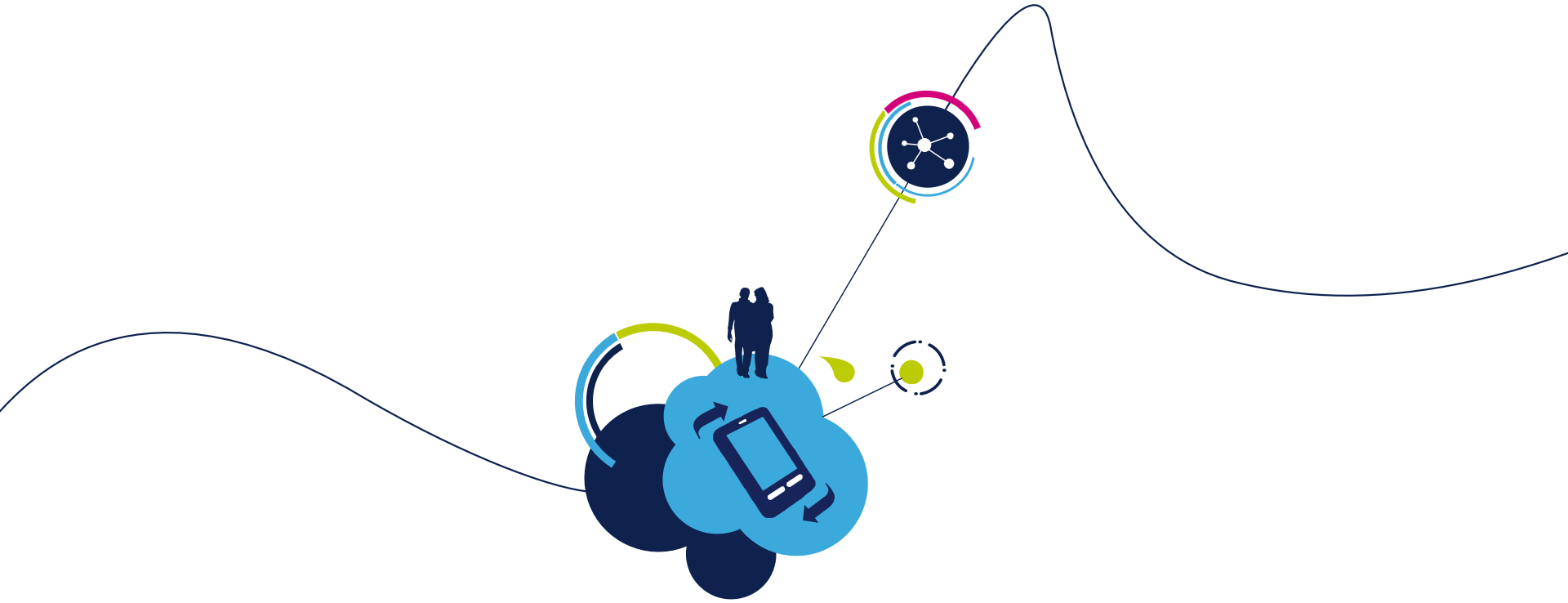



*Tera Term output*

All

**OK**

**+WIND:62:Socket Client Gone:192.168.1.102**



# Proceed to the next LAB!

# Lab 14: Low power modes



- Objective
  - Practice with Radio Power Save
  - Sleep Mode
  - Standby Mode
- Prerequisites
  - module connected to the AP

# Lab 14: Low power modes

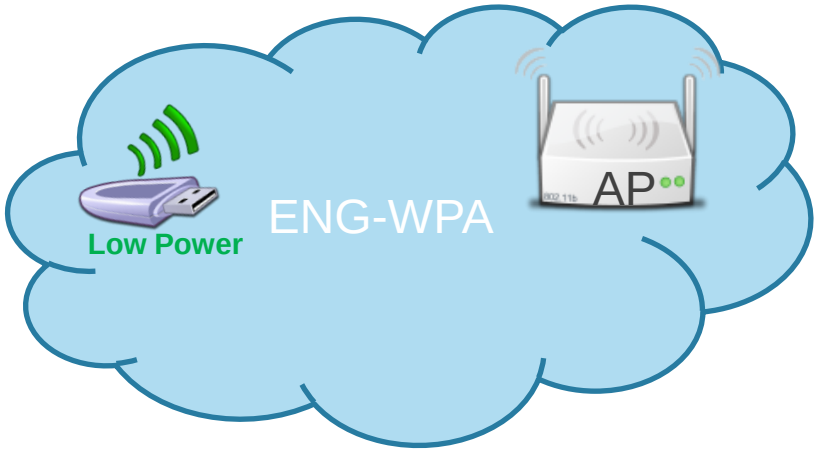


This feature allows to enable the low power states.

The module supports the “Radio Power Save” mode, the “Sleep” mode and the “Standby” mode.

The module should be connected to the AP (as shown in Lab 3) in order to use the “Radio Power Save” mode.

Please refer to “SPWF01S Power Management Application Notes” for more details.



# Lab 14: Joining a network



In order to be connected to an available Wifi network, the AP parameters setting is needed.

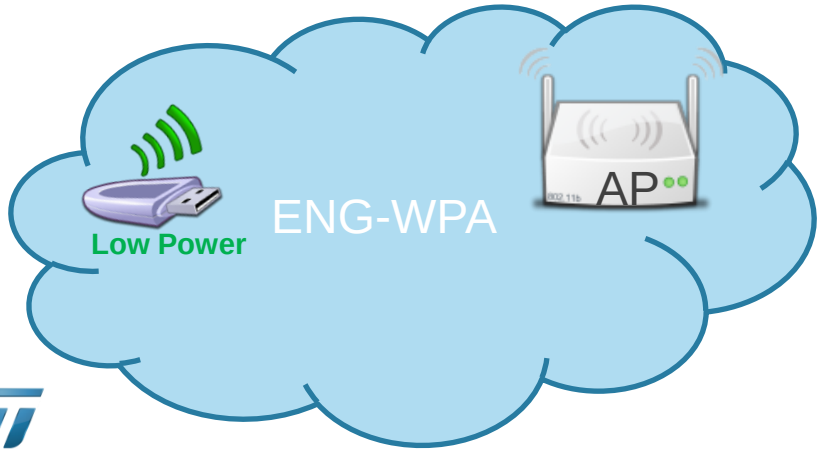
- Set the SSID
  - Type **AT+S.SSIDTXT=ENG-WPA**
- Set the password
  - Type **AT+S.SCFG=wifi_wpa_psk_text,helloworld**
- Set the network privacy mode (0=none, 1=WEP, 2=WPA-Personal (TKIP/AES) or WPA2-Personal (TKIP/AES))
  - Type **AT+S.SCFG=wifi_priv_mode,2**
- Set the network mode (1 = STA, 2 = IBSS, 3 = MiniAP)
  - Type **AT+S.SCFG=wifi_mode,1**

*Tera Term output*

# Lab 14: Joining a network



- Save the settings on the flash memory and reset the module
  - Type **AT&W**
  - Type **AT+CFUN=1**




*Tera Term output*

All

```
+WIND:0:Console active
+WIND:46:WPA: Crunching PSK...
+WIND:32:WiFi Hardware Started
+WIND:21:WiFi Scanning
+WIND:35:WiFi Scan Complete (0x0)
+WIND:19:WiFi Join: 14:D6:4D:24:36:00
+WIND:25:WiFi Association with 'ENG-WPA' successful
+WIND:51:WPA Handshake Complete
+WIND:24:WiFi Up: 192.168.0.1xx
```

# Lab 14: Low power modes – Power Save



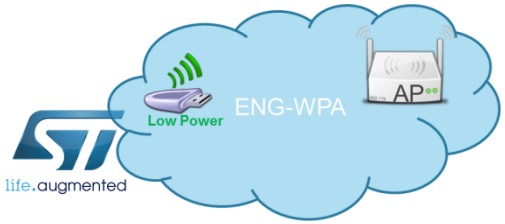
| Module Power State | STM32 | WLAN |
|---|---|---|
| **Standby** | Standby | Standby |
| **Sleep** | Stop | PS |
| **Power Save** | Run | PS or Fast-PS |
| **Active Rx** | Run | Rx Idle / Rx Active |
| **Active Tx** | Run | Tx Active |

# Lab 14: Low power modes – Power Save



- By default, the module starts in ACTIVE mode.
- Enable the Power Save Mode:
  - Type **at+s.scfg=wifi_powersave,1** (“wifi_powersave,2” enables the Fast-PS mode)
- Enable the doze operational mode:
  - Type **at+s.scfg=wifi_operational_mode,11** (“wifi_operational_mode,12” enables the quiescent mode)
- Choose the wake up mode:
  1. Wake up every n. beacon (specified in the wifi_beacon_wakeup variable)
     - Type **at+s.scfg=wifi_listen_interval,0**
     - Type **at+s.scfg=wifi_beacon_wakeup,1**

     **OR**
  2. Wake up every n. beacon adaptively (specified in the wifi_beacon_wakeup variable)
     - Type **at+s.scfg=wifi_listen_interval,1**
     - Type **at+s.scfg=wifi_beacon_wakeup,1**



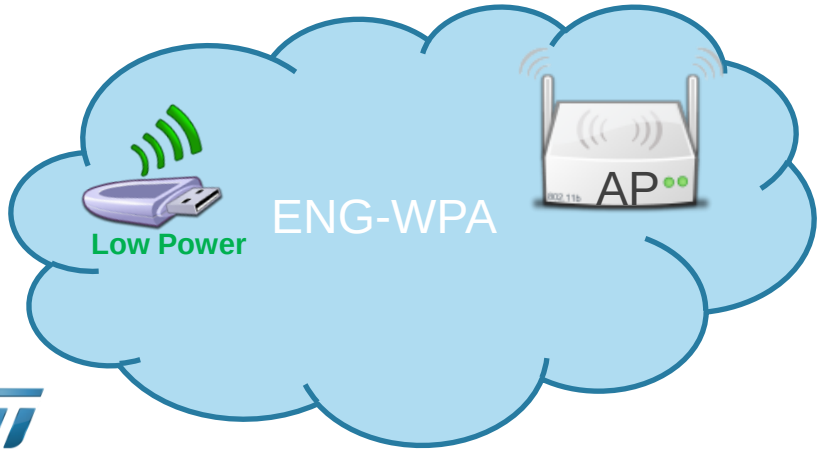

***Tip: The Wifi Power Save (wifi_powersave,1) can also be quickly enabled using the command: AT+CFUN=2***

# Lab 14: Low power modes – Power Save



- Save the settings on the flash memory and reset the module
  - Type **AT&W**
  - Type **AT+CFUN=1**

- The WIND:66 message related to Low Power Mode will be displayed




*Tera Term output*

**All**

```
+WIND:1:Poweron (******-*******-SPWF01S)
+WIND:13:ST SPWF01S IWM: Copyright (c) 2012-2014
STMicroelectronics, Inc.  All rights Reserved.
+WIND:3:Watchdog Running
+WIND:0:Console active
+WIND:32:WiFi Hardware Started
+WIND:21:WiFi Scanning
+WIND:35:WiFi Scan Complete (0x0)
+WIND:19:WiFi Join: 02:62:1F:51:8F:0B
+WIND:25:WiFi Association with 'IoT' successful
+WIND:51:WPA Handshake Complete
+WIND:24:WiFi Up: 192.168.1.104
+WIND:66:Low Power mode:1
```

# Lab 14: Low power modes – Sleep Mode



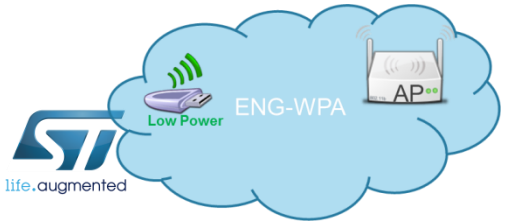
| Module Power State | STM32 | WLAN |
|---|---|---|
| **Standby** | Standby | Standby |
| **Sleep** | Stop | PS or Fast-PS |
| **Power Save** | Run | PS or Fast-PS |
| **Active Rx** | Run | Rx Idle / Rx Active |
| **Active Tx** | Run | Tx Active |

# Lab 14: Low power modes – Sleep Mode



- In the Sleep Mode, the core STM32 is stopped
- Enable the Sleep Mode:
  - Type **at+s.scfg=sleep_enabled,1**
- Enable the Power Save Mode:
  - Type **at+s.scfg=wifi_powersave,1** (“wifi_powersave,2” enables the Fast-PS mode)
- Enable the doze operational mode:
  - Type **at+s.scfg=wifi_operational_mode,11** (“wifi_operational_mode,12” enables the quiescent mode)
- Choose the wake up mode:
  1. Wake up every n. beacon (specified in the wifi_beacon_wakeup variable)
     - Type **at+s.scfg=wifi_listen_interval,0**
     - Type **at+s.scfg=wifi_beacon_wakeup,1**

     **OR**
  2. Wake up every n. beacon adaptively (specified in the wifi_beacon_wakeup variable)
     - Type **at+s.scfg=wifi_listen_interval,1**
     - Type **at+s.scfg=wifi_beacon_wakeup,1**



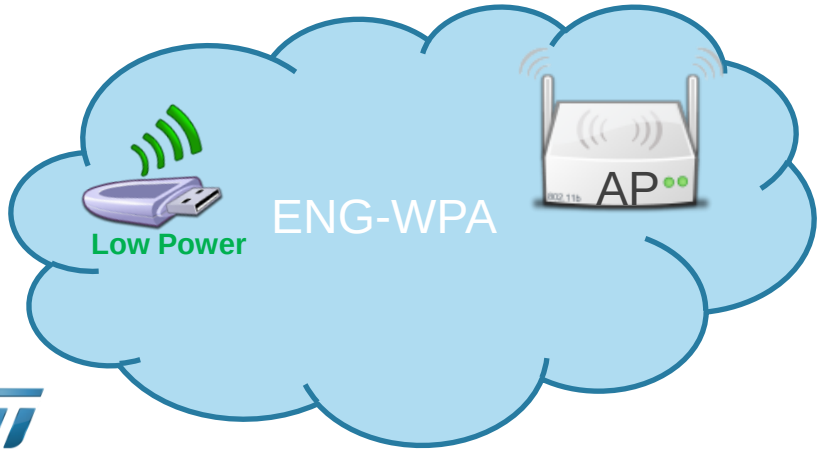

***Tip: The Wifi Sleep Mode (and wifi_powersave,1) can also be quickly enabled using the command: AT+CFUN=3***

# Lab 14: Low power modes – Sleep Mode



- Save the settings on the flash memory and reset the module
  - Type **AT&W**
  - Type **AT+CFUN=1**

- The WIND:69 message related to Sleep Mode will be displayed




*Tera Term output*

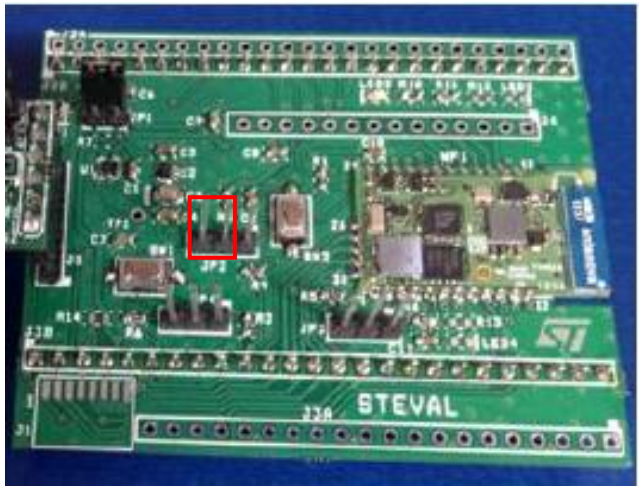
**All**

```
+WIND:1:Poweron (******-*******-SPWF01S)
+WIND:13:ST SPWF01S IWM: Copyright (c) 2012-2014
STMicroelectronics, Inc.  All rights Reserved.
+WIND:3:Watchdog Running
+WIND:0:Console active
+WIND:32:WiFi Hardware Started
+WIND:21:WiFi Scanning
+WIND:35:WiFi Scan Complete (0x0)
+WIND:19:WiFi Join: 02:62:1F:51:8F:0B
+WIND:25:WiFi Association with 'IoT' successful
+WIND:51:WPA Handshake Complete
+WIND:24:WiFi Up: 192.168.1.104
+WIND:66:Low Power mode:1
+WIND:69:Going into DeepSleep
```

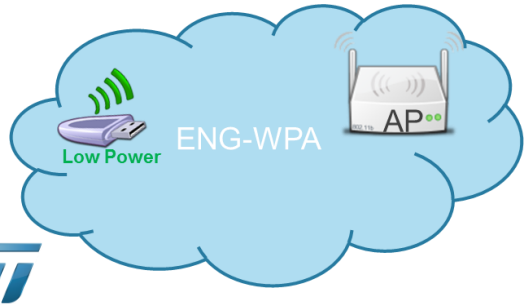
# Lab 14: Low power modes – Sleep Mode



- Wake STM32 up using the GPIO6
  - Put the GPIO6 to 2.5V (jumper on JP2 as in the picture)




- The WIND messages will be displayed




*Tera Term output*

**All**

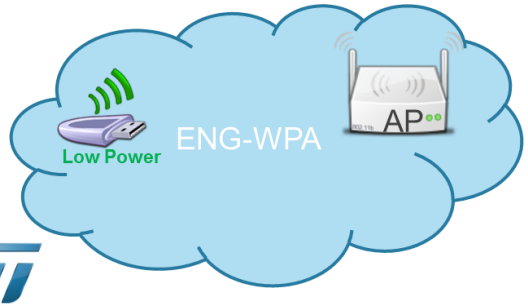
**+WIND:70:Resuming from DeepSleep**

**+WIND:53:Wakeup (GPIO6 High)**

# Lab 14: Low power modes – Sleep Mode



- Put STM32 in sleep mode using the GPIO6
  - Go back the GPIO6 floating (remove jumper - as default)
- The WIND:69 message will be displayed




*Tera Term output*

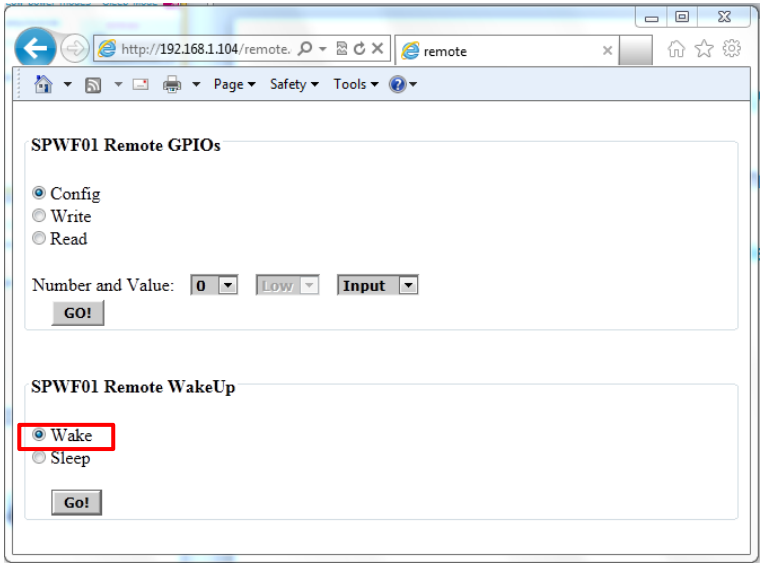
All

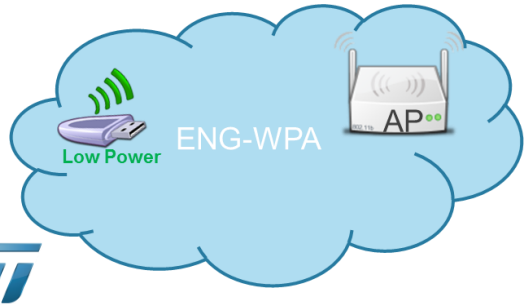
```
+WIND:69:Going into DeepSleep
```

# Lab 14: Low power modes – Sleep Mode



- Wake STM32 up using the remote page
  - **Connect a device to the same module's network**
  - **Open the following link: http://[module_IP_address]/remote.html**
  - **Select "Wake" and click on "Go!" button to wake up the module**
- The WIND message will be displayed




*Tera Term output*

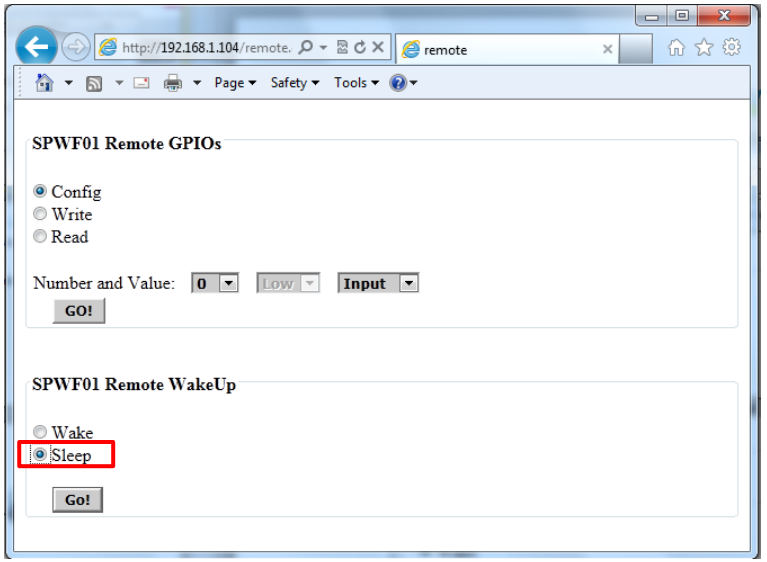
**All**

**+WIND:70:Resuming from DeepSleep**

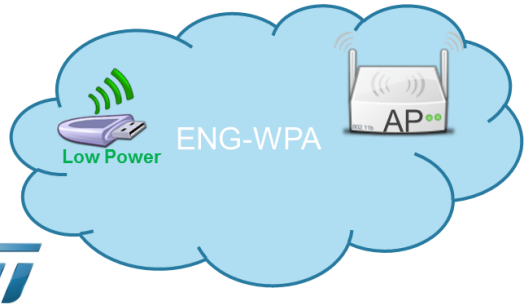
# Lab 14: Low power modes – Sleep Mode



- Put STM32 in sleep mode using the remote page
  - **Connect a device to the same module's network**
  - **Open the following link: http://[module_IP_address]/remote.html**
  - **Select "Sleep" and click on "Go!" button to put the module in sleep mode**




- The WIND message will be displayed




*Tera Term output*

All

**+WIND:69:Going into DeepSleep**

# Lab 14: Low power modes – Standby Mode



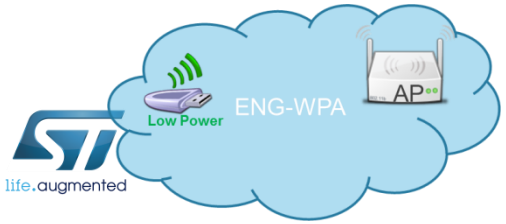
| Module Power State | STM32 | WLAN |
|---|---|---|
| **Standby** | Standby | Standby |
| **Sleep** | Stop | PS or Fast-PS |
| **Power Save** | Run | PS or Fast-PS |
| **Active Rx** | Run | Rx Idle / Rx Active |
| **Active Tx** | Run | Tx Active |

# Lab 14: Low power modes – Standby Mode



- During the Standby Mode, both the STM32 and the Radio will be put in standby mode. The standby mode allows to achieve the lowest power consumption.

- Enable the Standby Mode:
  - Type **at+s.scfg=sleep_enabled,0 (Sleep mode must be disabled)**
  - Type **at+s.scfg=standby_enabled,1**
  - Please be sure that GPIO6 isn't forced high
- Set the standby time to wake up via RTC alarm:
  - Type **at+s.scfg=standby_time,15**



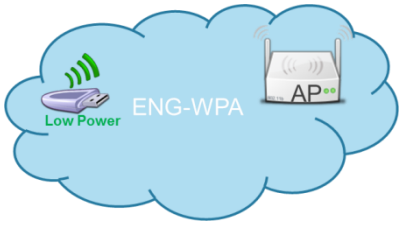

***Tip: The Wifi Standby Mode can also be quickly enabled using the command: AT+CFUN=4***

# Lab 14: Low power modes – Standby Mode



- Save the settings on the flash memory and reset the module
  - Type **AT&W**
  - Type **AT+CFUN=1**

- After 15 seconds, the module will be rebooted and will return in the ACTIVE state.




*Tera Term output*

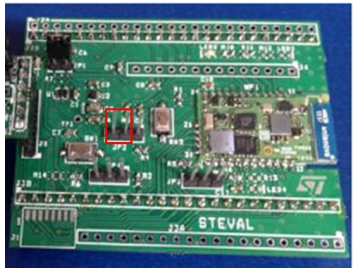
All

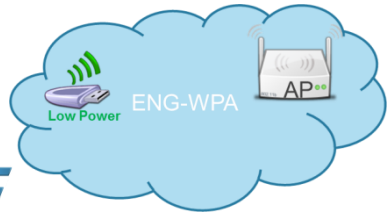
```
+WIND:0:Console active
+WIND:32:WiFi Hardware Started
+WIND:38:WiFi: Powered Down
+WIND:67:Going into Standby:15

+WIND:68:Resuming from Standby
+WIND:1:Poweron (******-*******-SPWF01S)
+WIND:13:ST SPWF01S IWM: Copyright (c) 2012-2014
STMicroelectronics, Inc.  All rights Reserved.
+WIND:3:Watchdog Running
+WIND:0:Console active
+WIND:32:WiFi Hardware Started
+WIND:21:WiFi Scanning
+WIND:35:WiFi Scan Complete (0x0)
+WIND:19:WiFi Join:02:62:1F:51:8F:0B
+WIND:25:WiFi Association with 'IoT' successful
+WIND:51:WPA Handshake Complete
+WIND:24:WiFi Up:192.168.1.113
```

# Lab 14: Low power modes – Standby Mode



- Wake up the module using the GPIO6
- Enable the Standby Mode:
  - Type **at+s.scfg=sleep_enabled,0 (make sure that Sleep mode is disabled)**
  - Type **at+s.scfg=standby_enabled,1**
  - Type **AT&W**
  - Type **AT+CFUN=1**
- Put the GPIO6 to 2.5V to wake up the module
- The WIND message will be displayed and the module will return in ACTIVE state

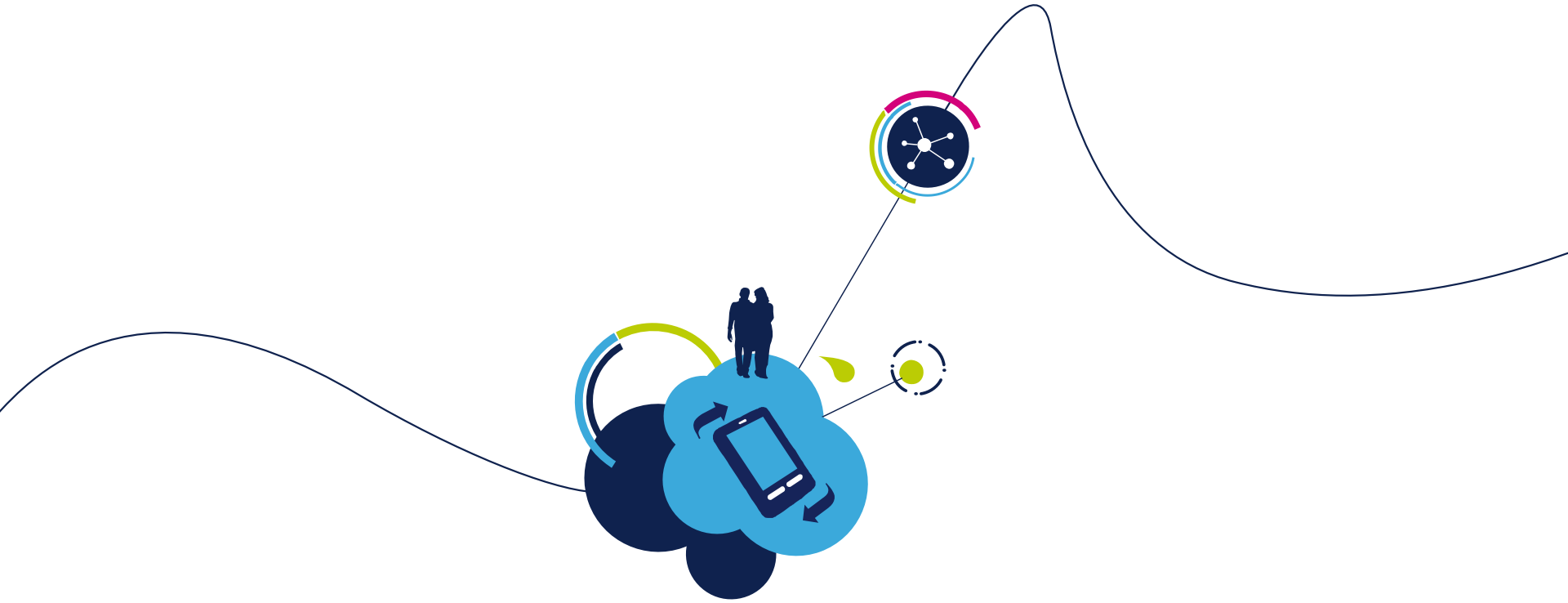



*Tera Term output*

**All**

**+WIND:68:Resuming from Standby**
**+WIND:1:Poweron (******-*******-SPWF01S)**
**……..**

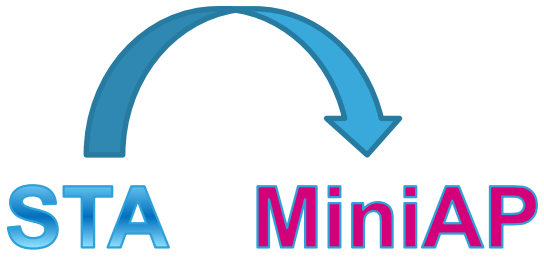


# Proceed to the next LAB!

# Lab 15: STA/MINI AP switcher



- Objective
  - HW “STA to MiniAP” switcher
- Prerequisites
  - Work alone



# Lab 15: STA/MINI AP switcher



This feature allows to force the module in Mini AP mode starting from a preexistent state.

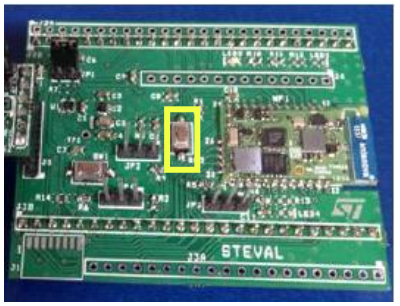
“Recovery Mode”: this functionality could be useful to lead the module in a known state and to reconfigure it (i.e. using the firstset page).

***The GPIO7 will be used to drive this feature.***

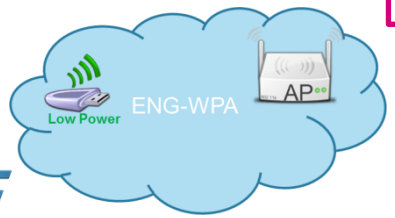
# Lab 15: STA/MINI AP switcher



- Press and hold the SW2 button on the EVAL and then perform a reset (HW reset via SW1 button or SW reset via AT command)
- Release the SW2 button
- The MiniAP mode will be started and the module is discoverable with the following SSID:
  - i**wm-XX-YY-ZZ** where XX-YY-ZZ are the last six digits of module's MAC ADDRESS



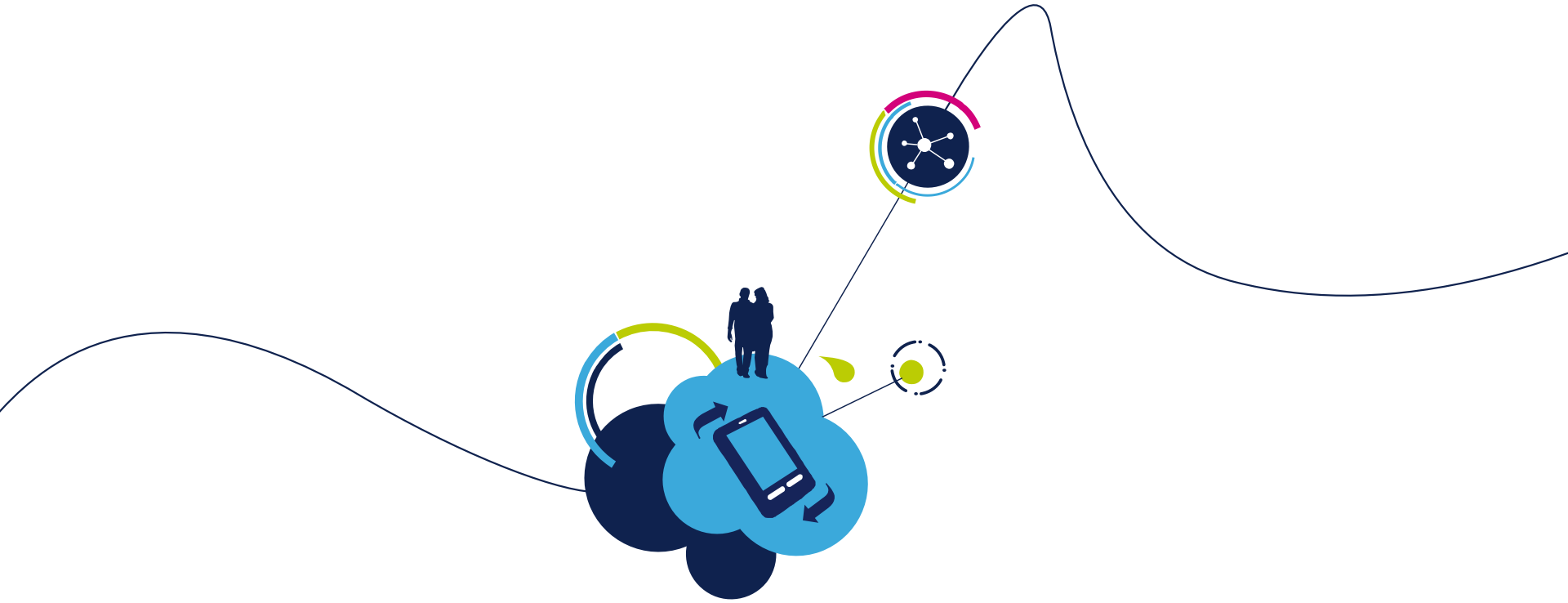

***Tip: To exit from "miniAP mode via GPIO7", perform a module's reset***




*Tera Term output*

**All**

**+WIND:2:Reset**
**+WIND:1:Poweron (******-*******-SPWF01S)**
**+WIND:13:ST SPWF01S IWM: Copyright (c) 2012-2014 STMicroelectronics, Inc. All rights Reserved.**
**+WIND:39:HW in miniAP mode (GPIO7 Low)**
**+WIND:0:Console active**
**+WIND:3:Watchdog Running**
**+WIND:32:WiFi Hardware Started**
**+WIND:26:WiFi Started AP**
**+WIND:24:WiFi Up:172.31.255.1**

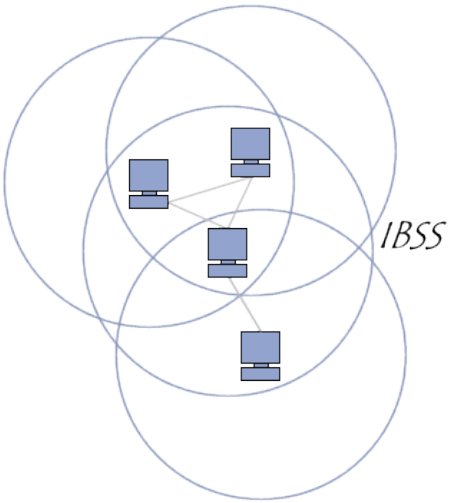


# Proceed to the next LAB!

# Lab 16: IBSS connection



- Objective
  - Create an IBSS network
- Prerequisites
  - Work alone



# Lab 16: Create an IBSS network



In an IBSS network, the SSID is chosen by the client device that starts the network.

**Module settings to create an IBSS network:**

- Set the IBSS SSID
  - Type **AT+S.SSIDTXT=ADHOC**
- Set the network privacy mode (0=OPEN or 1=WEP are supported)
  - Type **AT+S.SCFG=wifi_priv_mode,0**
- Set the network mode (2 = IBSS)
  - Type **AT+S.SCFG=wifi_mode,2**

*Tera Term output*

# Lab 16: Create an IBSS network



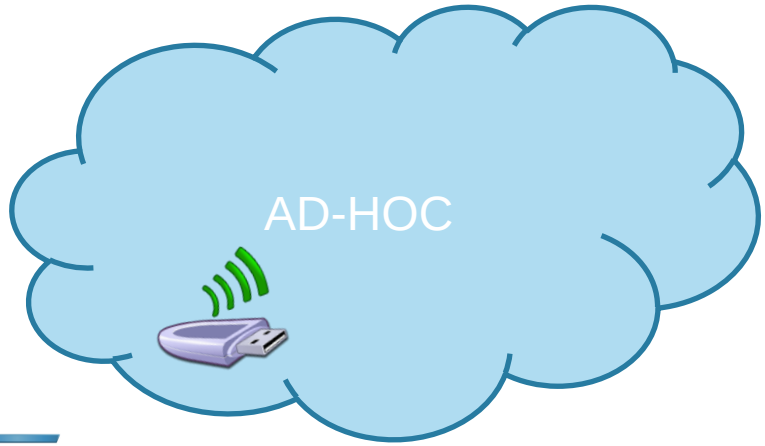
- Set IP address, IP default gateway, IP DNS and IP netmask
  - Type **AT+S.SCFG=ip_ipaddr,192.168.y.1xx**
  - Type **AT+S.SCFG=ip_gw,192.168.y.1**
  - Type **AT+S.SCFG=ip_dns,192.168.y.1**
  - Type **AT+S.SCFG=ip_netmask,255.255.255.0**
- Turn off the DHCP
  - Type **AT+S.SCFG=ip_use_dhcp,0**

*Tera Term output*

# Lab 16: Create an IBSS network



- Save the settings on the flash memory and reset the module
  - Type **AT&W**
  - Type **AT+CFUN=1**




*Tera Term output*

**All**

```
+WIND:0:Console active
+WIND:32:WiFi Hardware Started
+WIND:21:WiFi Scanning
+WIND:35:WiFi Scan Complete (0x0)
+WIND:19:WiFi Join: EE:33:CE:00:24:18
+WIND:25:WiFi Association with 'ADHOC' successful
+WIND:51:WiFi Handshake Complete
+WIND:24:WiFi Up: 192.168.1.3
```

# Lab 16: Create an IBSS network



- Associate the iOS device with the ADHOC network

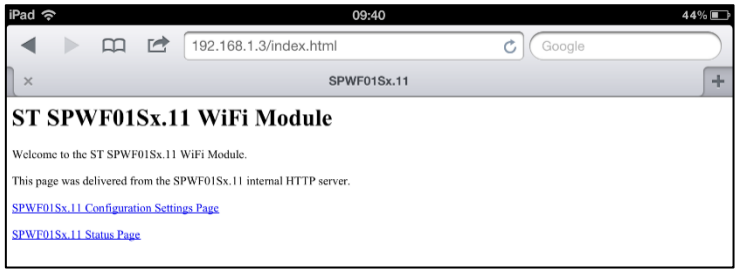

# Lab 16: Create an IBSS network



- Find your IP address
  - Type **AT+S.STS**

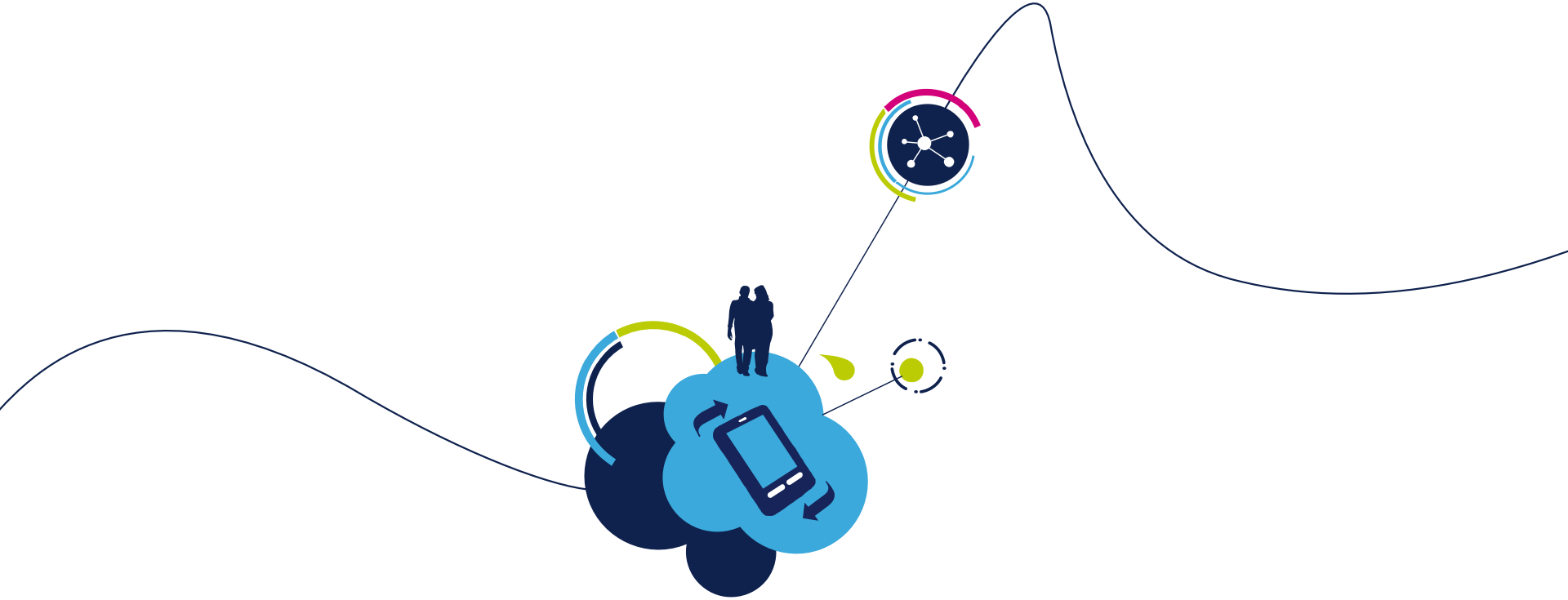
```
#  ip_ipaddr = 192.168.1.3
#  ip_netmask = 255.255.255.0
#  ip_gw = 192.168.1.1
```

- Open Safari web browser
- In the address bar, type <SPWF IP address>/index.html
  - Type 192.168.y.1xx/index.html





life.augmented



# THANK YOU!