SPB204 + STM32F1,F4 Quick Start Guide

1 Introduction

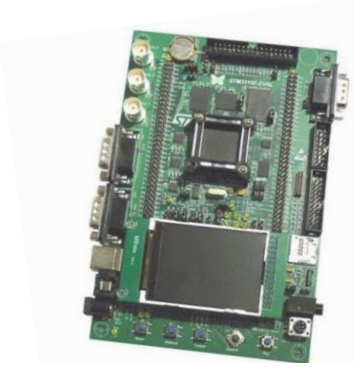
This guide will help you to compile the program and download it to either the STM32F10E-Eval or STM32F324xG-Eval board. The STM32F10E board does not have a touchdisplay and the demo application has to be run via a consol interface.

This getting started guide as well as all development tools and documentation is updated frequently. Please visit www.hd-wireless.se/software&tools **and look for an up to date version of this guide and associated software and development tools before you begin installing this kit.**

2 Requirements

- H&D wireless SPB204 802.11b+g module
- STM3210E or STM324xG Evaluation board
- SD to Micro-SD adapter
- USB to USART adapter (In STM3210E eval case) or USB to serial TTL-level adapter (in STM324xG eval case).
- GCC toolchain, Openocd, Linux and a terminal program e.g. Minicom.
- JTAG debugger (Jlink, Rlink or similar)

3 Getting started with H&D SPB204 + STM3210E(F1) or STM324xG(F4)

STM32F4	STM32F1	SPB204 with uSD adapter	SPB204 Wi-Fi SD EVK
			

- Power down the eval board
- Insert the SPB204 SDIO Wi-Fi card into the SD-Adapter SD end and the Micro-SD end to the eval board Micro SD port.
- Connect the USB-USART adapter between the PC and the USART1 port on STM3210E board. The USART-ports on the STM324xG cannot be used (because of shared resources). If the STM324xG board is to be used, a USB to serial TTL-level adapter with 2.5” connectors has to be used. Connect the USB end to PC and the TTL-connectors to the STM324xG pinheaders as follows:
  RX to CN2 Pin 32, TX to CN2 Pin 33, GND to CN3 Pin 50.
- Connect a debugger (Jlink or Rlink or similar) to the JTAG connector on the board.
- Start a serial port terminal (e.g.Linux Minicom or similar) Serial port settings: 115200-8-N-1.

4 Compiling the HTTP demo application

The Wi-Fi library is compiled with GCC, the toolchain can be downloaded here:
http://www.mentor.com/embedded-software/sourcery-tools/sourcery-codebench/editions/lite-edition/

Download the EABI Release
Submit registration
Select "Sourcery G++ Lite 2011.03-42" in the link obtained throught the "confirm email".
Download "IA32 GNU/Linux Installer" or "IA32 GNU/Linux TAR"

Make sure that the arm-none-eabi- tools are in your path.

Drivers from the ST Standard Peripheral Libraries are used and need to be downloaded.
These can be downloaded here:

For STM3210E:
http://www.st.com/st-web-ui/static/active/en/st_prod_software_internet/resource/technical/software/firmware/stsw-stm32054.zip

For STM324xG:
http://www.st.com/st-web-ui/static/active/en/st_prod_software_internet/resource/technical/software/firmware/stm32f4_dsp_stdperiph_lib.zip

To compile using the provided makefile in the gcc/ directory, open a terminal in Linux and type in these commands:

For STM3210E-eval type:
$ cd path/to/owl_stm32-r7634/platform/stm32/apps/http_server/gcc
$ export STM32F10x_STD_PERIPH_BASE=/path/to/STM32F10x_StdPeriph_Lib_V3.5.0
$ make BOARD=STM3210E_EVAL CONFIG=(sta,ap) (sta for station mode and ap for access point mode)

For STM324xG-eval type:
$ cd path/to/owl_stm32-r7634
$ export STM32F4xx_STD_PERIPH_BASE=/path/to/STM32F4xx_DSP_StdPeriph_Lib_V1.0.1
$ make BOARD=STM324xG_EVAL CONFIG=(sta,ap) (sta for station mode and ap for access point mode)

When this is done the screen should look like this:

```
arm-none-eabi-gcc -std=gnu99 -W -Wwrite-strings -Winline -Wfloat-equal -Wundef -Wredundant-decls -Winit-self -Wno-psabi -Wlogical-op -O0 -ffunct
ion-sections -fdata-sections -I .. -I.. -I../wl_api -I../owl/core -I../owl/ports/stm32 -I../lwip/ports/wl/include -I../lwip/lwip-1.3.2/src/inclu
de -I../lwip/lwip-1.3.2/src/include/ipv4 -I/home/sebbe/ARM/SDK_STM32F4/SDK/STM32F4xx_DSP_StdPeriph_Lib_V1.0.1/Libraries/STM32F4xx_StdPeriph_Driv
er/inc -I/home/sebbe/ARM/SDK_STM32F4/SDK/STM32F4xx_DSP_StdPeriph_Lib_V1.0.1/Libraries/CMSIS/Include -I/home/sebbe/ARM/SDK_STM32F4/SDK/STM32F4xx_
DSP_StdPeriph_Lib_V1.0.1/Libraries/CMSIS/Device/ST/STM32F4xx/Include -I/home/sebbe/ARM/SDK_STM32F4/SDK/STM32F4xx_DSP_StdPeriph_Lib_V1.0.1/Utilit
ies/STM32_EVAL/STM3240_41_G_EVAL -I/home/sebbe/ARM/SDK_STM32F4/SDK/STM32F4xx_DSP_StdPeriph_Lib_V1.0.1/Utilities/STM32_EVAL/Common -I/home/sebbe/
ARM/SDK_STM32F4/SDK/STM32F4xx_DSP_StdPeriph_Lib_V1.0.1/Project/STM32F4xx_StdPeriph_Templates -DUSE_STDPERIPH_DRIVER -DSTM32F4XX -DSYSCLK_FREQ_16
8MHz=168000000 -DHSE_VALUE=25000000  -DUSE_STM324xG_EVAL -DDEBUG_DEBUG -DMODE_LEGACY -DOWL_CONFIG_STA=1 -DOWL_CONFIG_LOOP=0 -DOWL_CONFIG_AP=0 -D
CONFIG_OWL -DBYTE_ORDER=LITTLE_ENDIAN -DHDG104 -g -c -mcpu=cortex-m4 -mthumb ../lcd_ap.c -o ../bin/lcd_ap.o
arm-none-eabi-gcc -std=gnu99 -W -Wwrite-strings -Winline -Wfloat-equal -Wundef -Wredundant-decls -Winit-self -Wno-psabi -Wlogical-op -O0 -ffunct
ion-sections -fdata-sections -I .. -I.. -I../wl_api -I../owl/core -I../owl/ports/stm32 -I../lwip/ports/wl/include -I../lwip/lwip-1.3.2/src/inclu
de -I../lwip/lwip-1.3.2/src/include/ipv4 -I/home/sebbe/ARM/SDK_STM32F4/SDK/STM32F4xx_DSP_StdPeriph_Lib_V1.0.1/Libraries/STM32F4xx_StdPeriph_Driv
er/inc -I/home/sebbe/ARM/SDK_STM32F4/SDK/STM32F4xx_DSP_StdPeriph_Lib_V1.0.1/Libraries/CMSIS/Include -I/home/sebbe/ARM/SDK_STM32F4/SDK/STM32F4xx_
DSP_StdPeriph_Lib_V1.0.1/Libraries/CMSIS/Device/ST/STM32F4xx/Include -I/home/sebbe/ARM/SDK_STM32F4/SDK/STM32F4xx_DSP_StdPeriph_Lib_V1.0.1/Utilit
ies/STM32_EVAL/STM3240_41_G_EVAL -I/home/sebbe/ARM/SDK_STM32F4/SDK/STM32F4xx_DSP_StdPeriph_Lib_V1.0.1/Utilities/STM32_EVAL/Common -I/home/sebbe/
ARM/SDK_STM32F4/SDK/STM32F4xx_DSP_StdPeriph_Lib_V1.0.1/Project/STM32F4xx_StdPeriph_Templates -DUSE_STDPERIPH_DRIVER -DSTM32F4XX -DSYSCLK_FREQ_16
8MHz=168000000 -DHSE_VALUE=25000000  -DUSE_STM324xG_EVAL -DDEBUG_DEBUG -DMODE_LEGACY -DOWL_CONFIG_STA=1 -DOWL_CONFIG_LOOP=0 -DOWL_CONFIG_AP=0 -D
CONFIG_OWL -DBYTE_ORDER=LITTLE_ENDIAN -DHDG104 -g -c -mcpu=cortex-m4 -mthumb ../main.c -o ../bin/main.o
arm-none-eabi-gcc -std=gnu99 -W -Wwrite-strings -Winline -Wfloat-equal -Wundef -Wredundant-decls -Winit-self -Wno-psabi -Wlogical-op -O0 -ffunct
ion-sections -fdata-sections -I .. -I.. -I../wl_api -I../owl/core -I../owl/ports/stm32 -I../lwip/ports/wl/include -I../lwip/lwip-1.3.2/src/inclu
de -I../lwip/lwip-1.3.2/src/include/ipv4 -I/home/sebbe/ARM/SDK_STM32F4/SDK/STM32F4xx_DSP_StdPeriph_Lib_V1.0.1/Libraries/STM32F4xx_StdPeriph_Driv
er/inc -I/home/sebbe/ARM/SDK_STM32F4/SDK/STM32F4xx_DSP_StdPeriph_Lib_V1.0.1/Libraries/CMSIS/Include -I/home/sebbe/ARM/SDK_STM32F4/SDK/STM32F4xx_
DSP_StdPeriph_Lib_V1.0.1/Libraries/CMSIS/Device/ST/STM32F4xx/Include -I/home/sebbe/ARM/SDK_STM32F4/SDK/STM32F4xx_DSP_StdPeriph_Lib_V1.0.1/Utilit
ies/STM32_EVAL/STM3240_41_G_EVAL -I/home/sebbe/ARM/SDK_STM32F4/SDK/STM32F4xx_DSP_StdPeriph_Lib_V1.0.1/Utilities/STM32_EVAL/Common -I/home/sebbe/
ARM/SDK_STM32F4/SDK/STM32F4xx_DSP_StdPeriph_Lib_V1.0.1/Project/STM32F4xx_StdPeriph_Templates -DUSE_STDPERIPH_DRIVER -DSTM32F4XX -DSYSCLK_FREQ_16
8MHz=168000000 -DHSE_VALUE=25000000  -DUSE_STM324xG_EVAL -DDEBUG_DEBUG -DMODE_LEGACY -DOWL_CONFIG_STA=1 -DOWL_CONFIG_LOOP=0 -DOWL_CONFIG_AP=0 -D
CONFIG_OWL -DBYTE_ORDER=LITTLE_ENDIAN -DHDG104 -g -c -mcpu=cortex-m4 -mthumb ../owl/core/owl_wifi.c -o ../bin/owl_wifi.o
arm-none-eabi-gcc -Wl,--gc-sections,-Map=../bin/http_server_gui.map,-cref -T stm32F4.ld -L../gcc/wl_api -mcpu=cortex-m4 -mthumb ../bin/misc.o ..
/bin/stm32f4xx_adc.o ../bin/stm32f4xx_dcmi.o ../bin/stm32f4xx_hash.o ../bin/stm32f4xx_hash_md5.o ../bin/stm32f4xx_hash_sha1.o ../bin/stm32f4xx_r
ng.o ../bin/stm32f4xx_syscfg.o ../bin/stm32f4xx_cryp.o ../bin/stm32f4xx_cryp_aes.o ../bin/stm32f4xx_cryp_des.o ../bin/stm32f4xx_cryp_tdes.o ../b
in/stm32f4xx_can.o ../bin/stm32f4xx_crc.o ../bin/stm32f4xx_dac.o ../bin/stm32f4xx_dbgmcu.o ../bin/stm32f4xx_dma.o ../bin/stm32f4xx_exti.o ../bin
/stm32f4xx_flash.o ../bin/stm32f4xx_fsmc.o ../bin/stm32f4xx_gpio.o ../bin/stm32f4xx_i2c.o ../bin/stm32f4xx_iwdg.o ../bin/stm32f4xx_pwr.o ../bin/
stm32f4xx_rcc.o ../bin/stm32f4xx_rtc.o ../bin/stm32f4xx_sdio.o ../bin/stm32f4xx_spi.o ../bin/stm32f4xx_tim.o ../bin/stm32f4xx_usart.o ../bin/stm
32f4xx_wwdg.o ../bin/stm324xg_eval.o ../bin/stm324xg_eval_lcd.o ../bin/stm324xg_eval_ioe.o ../bin/system_stm32f4xx.o ../bin/startup_stm32f4xx.o
../bin/tcp.o ../bin/tcp_out.o ../bin/init.o ../bin/mem.o ../bin/memp.o ../bin/pbuf.o ../bin/netif.o ../bin/stats.o ../bin/raw.o ../bin/dhcp.o ..
/bin/udp.o ../bin/tcp_in.o ../bin/dns.o ../bin/ip.o ../bin/ip_frag.o ../bin/ip_addr.o ../bin/igmp.o ../bin/icmp.o ../bin/inet_chksum.o ../bin/in
et.o ../bin/autoip.o ../bin/etharp.o ../bin/wlif.o ../bin/owl_shell.o ../bin/owl_loop.o ../bin/owl_debug.o ../bin/owl_net.o ../bin/owl_timer.o .
./bin/owl_err.o ../bin/owl_os.o ../bin/owl_sdio.o ../bin/httpd.o ../bin/httpd_404.o ../bin/ttcp.o ../bin/lcd_demo.o ../bin/lcd_ap.o ../bin/main.
o ../bin/owl_wifi.o -lwl_sdio-cortex-m3 -lwl_api-cortex-m3-hdg104-sta-intwpa -o ../bin/http_server_gui.elf
```

4.1 Downloading the HTTP demo application

The second step is to flash (program) the actual demo application into the evaluation kit using openocd.

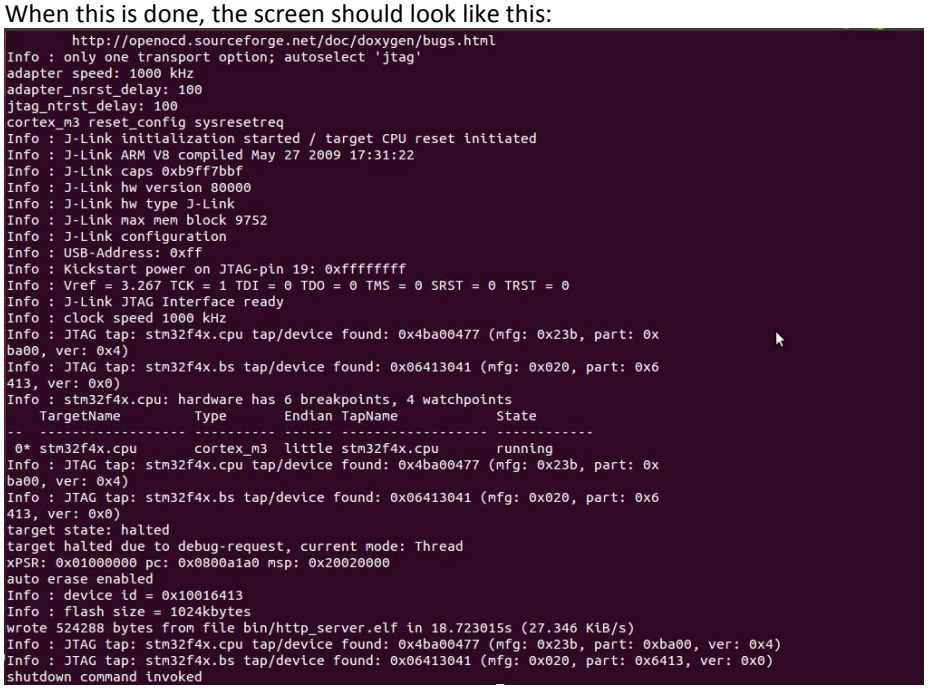
(Note: The openocd .cfg files that are included is written for older openocd versions. It is possible that the .cfg file has to be modified in order to flash the device properly. If this is the case, open the openocd_upload.cfg file located in the path/to/owl_stm32-r7634/platform/stm32/apps/http_server/openocd folder, in a text editor and change the line “source [find target/stm32.cfg]” to “source[find target/stm32f1x.cfg]”. Depending on the type of debugger used, the line “source [find interface/rlink.cfg]” should be changed to “source [find interface/jlink.cfg]” eg if a jlink debugger is used instead.)

Download openocd from the terminal using the command:
$ apt-get install openocd

In the linux terminal now write:
(For STM3210E-eval)
$ openocd -f openocd/openocd_upload.cfg

(For STM324xG-eval)
$ openocd -f openocd/openocd_upload_f4.cfg

When this is done, the screen should look like this:



5 Running the HTTP demo application via console

When the installation and compilation/programming is complete the HTTP demo application will start as soon as the evaluation kit is powered on.

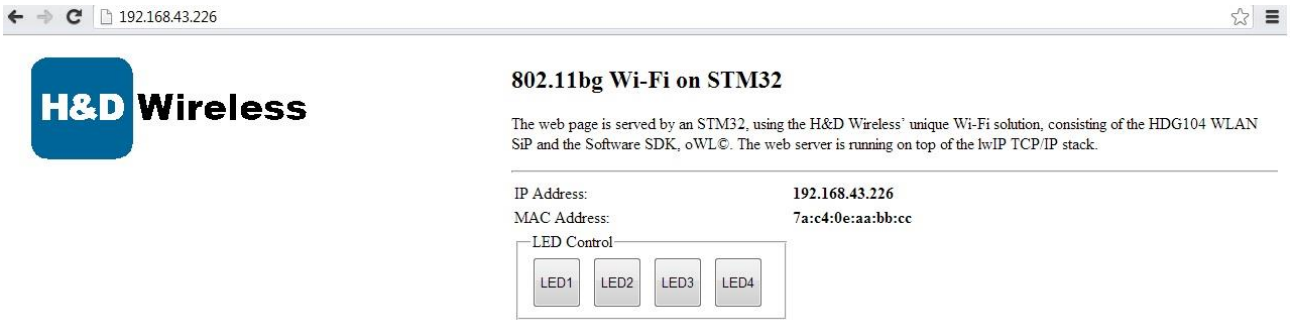
A command line interface to the application is provided through the STM3210E serial port, or the pinouts on the STM324xG described in section 3.

Open up a terminal program (e.g. Minicom) and set the serial port settings to: 115200-8-N-1.

- Use the command **“scan”** to find available access points in the area. The list of available network is limited to 16 networks. If more networks are found, the 16 with the strongest signal will be shown.
- If WEP encryption is used in the access point, use the command **“setkey”** to set the appropriate key.
- If WPA/WPA2/RSN Pre-shared key security is used in the access point, use the command **“wpass”** to set the pass phrase.
- Use the command **“connect [access point]”** to make a connection to an access point (see figure below).
- If the key is wrong the application will continue to try to connect until the right key is set.

```
$ help
scan                scan for networks
connect             connect to network
wpass               set wpa passphrase
powersave           set powersave mode
psconf              config powersave mode
setkey              set wep key(s)
dhcp                enable dhcp
ifconfig            config ip address
ttcp                ttcp throughput test
help                print this information
$
```

The assigned IP address will be printed during the connection procedure. It should now be possible to connect to the web server using a browser on a PC that is connected to the same network.



It is possible to control and supervise the STM32 EVAL board through the web interface:

- Four buttons, LED0-3, are shown on the web page. When clicking a button, the corresponding LED on the STM3210E / STM324xG will toggle from disabled-state to enabled-state or vice versa.

See the HTTP demo application user guide [1543-DRF100 HTTP-demo Users Manual] for more information on how to use the HTTP demo application.

6 Running the HTTP demo application via the GUI

On the STM324xG, the HTTP demo can be run stand alone using the on board display and GUI.
The Web interface works the same way as when the application is controlled by consol see 5. The network list after scan is limited to 16 networks.

The GUI-demo is precompiled and located in the owl_stm32-r3434 folder named http_server_gui.elf.
In the terminal, browse to the owl_stm32-r3434 folder.
Write: $ openocd -f openocd/openocd_upload_f4_gui.cfg.
This will program the STM324xG with the GUI-demo.

When finished, the GUI will be present on the screen.

The start menu will show four buttons:
(SCAN), scanning for nets.
(DHCP),used for setting DHCP on or off. In on mode the webserver will request an ip from the net connected to.

(WPASS), used for setting the WPA passphrase when connecting to nets using WPA/WPA2 protocol.
(WEPKEY),used for setting the WEPkey when connecting to nets with WEP protocol. The WEPkey has by default index 1. (Key 1).

Additional information is also shown:
(Bnd to:), shows the IP-address of the webserver.
(Co to:), shows the SSID of the net connected to.
(DHCP:), shows DHCP on or off.
(Wpass:), shows the WPA passphrase set by user.
(Wepkey:), shows the WEPkey set by user.

Start by pressing SCAN on the screen. This will scan the surrounding nets and show them in a list. After the SSID the encryption-type and rssi strength will be shown. If there are more than 8 nets available, a “NEXT” button will appear in the downright corner which will, if pressed, show the rest (up to 8 more) of the nets.

Press the BACK button to go back and set WPA passphrase or WEP or press SCAN again to update list of nets. Press the WPASS-button to set the WPA Passphrase. By pressing the “<” or “>” button different sets of symbols can be used to set passphrase. The button “<<” deletes one character in the passphrase (backspace). Press OK when done. In the startmenu you will see the passphrase set.

Now Press the SCAN button and select the net you want to connect to by pressing on the name of the net. When the connection is established the SSID of the net will be shown in the start menu.

Now press the DHCP button to set DHCP to on. If everything is ok the IP-adress of the webserver will be shown on the screen. (This can take a few seconds).

7 Further development and information

For information about how to develop custom WiFi applications using the H&D wireless interface, see the API documentation provided in the STM32 Software Framework.