NEWRACOM

NEW RADIO COMMUNICATION FOR NEW ERA

NRC7292 Evaluation Kit User Guide

(Standalone)

Ultra-low power & Long-range Wi-Fi

Ver 2.3
Feb. 28, 2023

NEWRACOM, Inc.

NRC7292 Evaluation Kit User Guide (Standalone)
Ultra-low power & Long-range Wi-Fi

Office

Newracom, Inc.
505 Technology Drive, Irvine, CA 92618 USA
http://www.newracom.com

Contents

List of Tables

List of Figures

1 Overview

This document introduces the NRC7292 Software Development Kit (SDK) that allows users to develop their application program running on NRC7292 Evaluation Boards (EVB) without external hosts. Figure to Figure 1.3 shows the pictures of NRC7292 EVB.

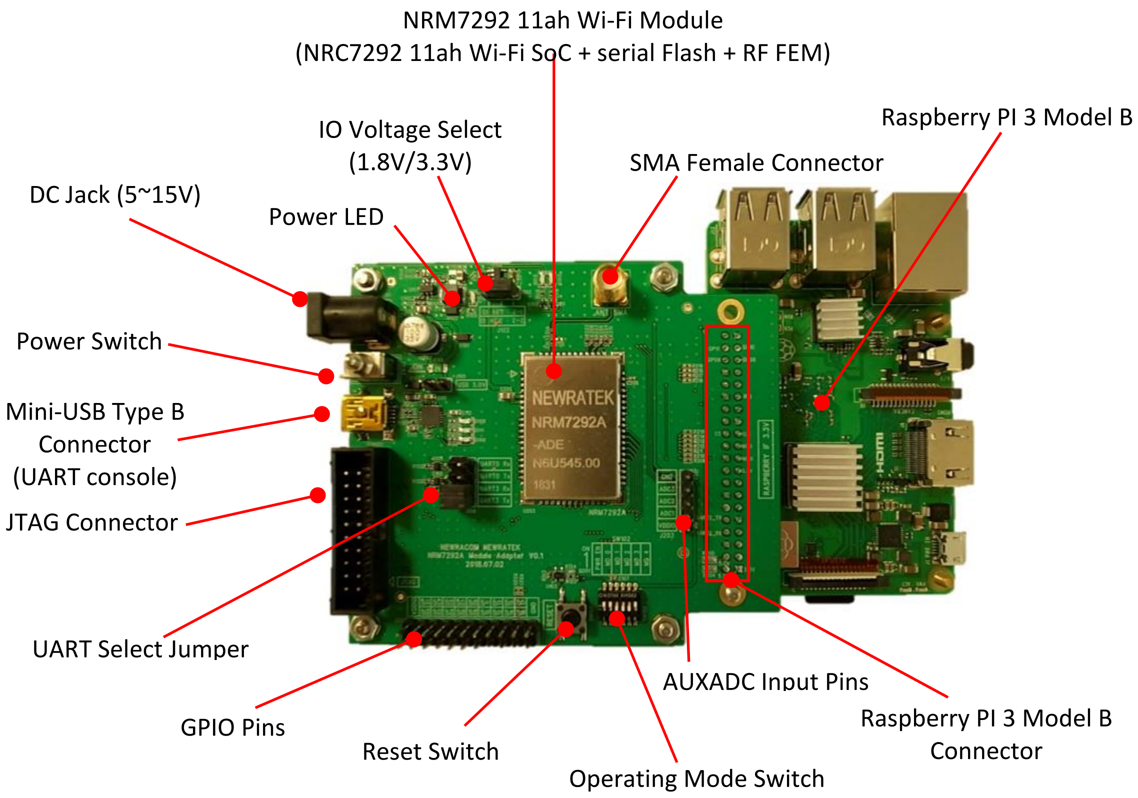

NRM7292 11ah Wi-Fi Module
(NRC7292 11ah Wi-Fi SoC + serial Flash + RF FEM)
IO Voltage Select (1.8V/3.3V)
Raspberry PI 3 Model B
SMA Female Connector
DC Jack (5~15V)
Power LED
Power Switch
Mini-USB Type B Connector (UART console)
JTAG Connector
UART Select Jumper
GPIO Pins
Reset Switch
AUXADC Input Pins
Raspberry PI 3 Model B Connector
Operating Mode Switch

Figure 1.1 NRC7292 evaluation board (v1.0)

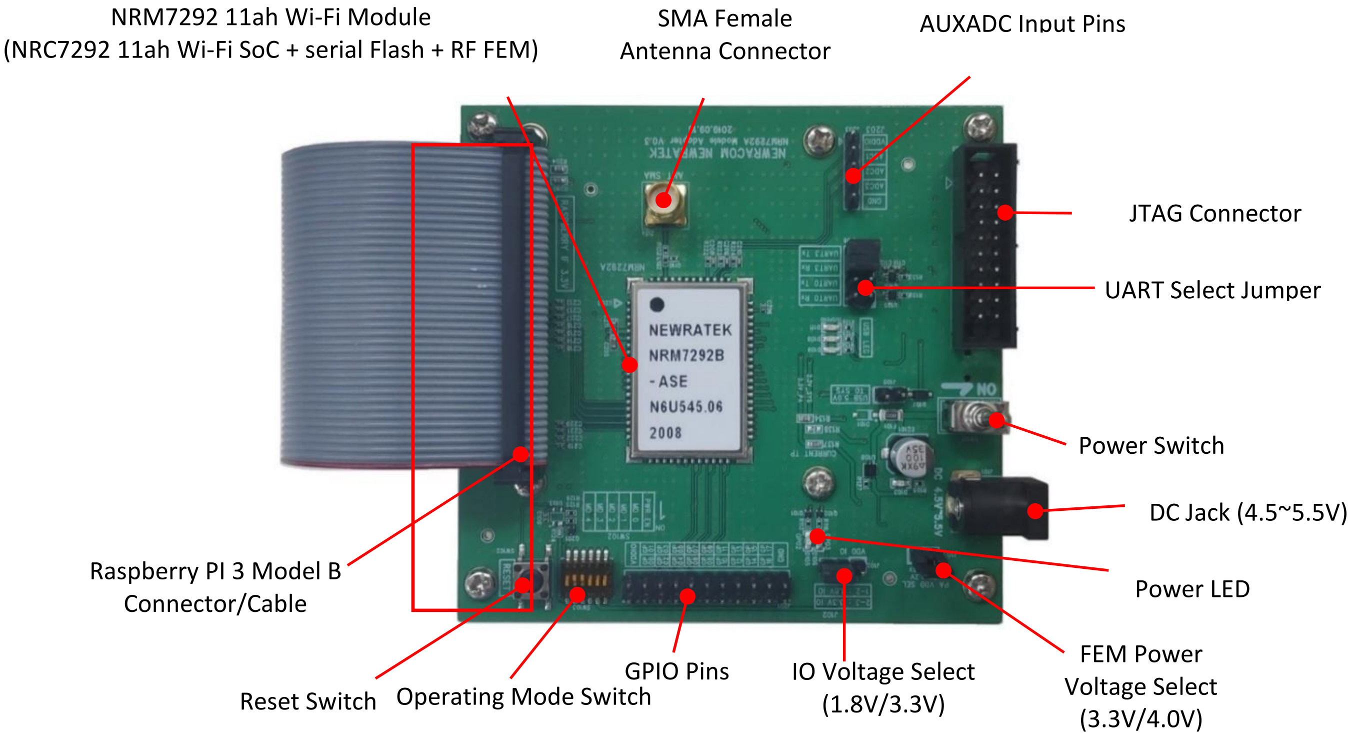


Figure 1.2 NRC7292 evaluation board (v2.0 – top view)

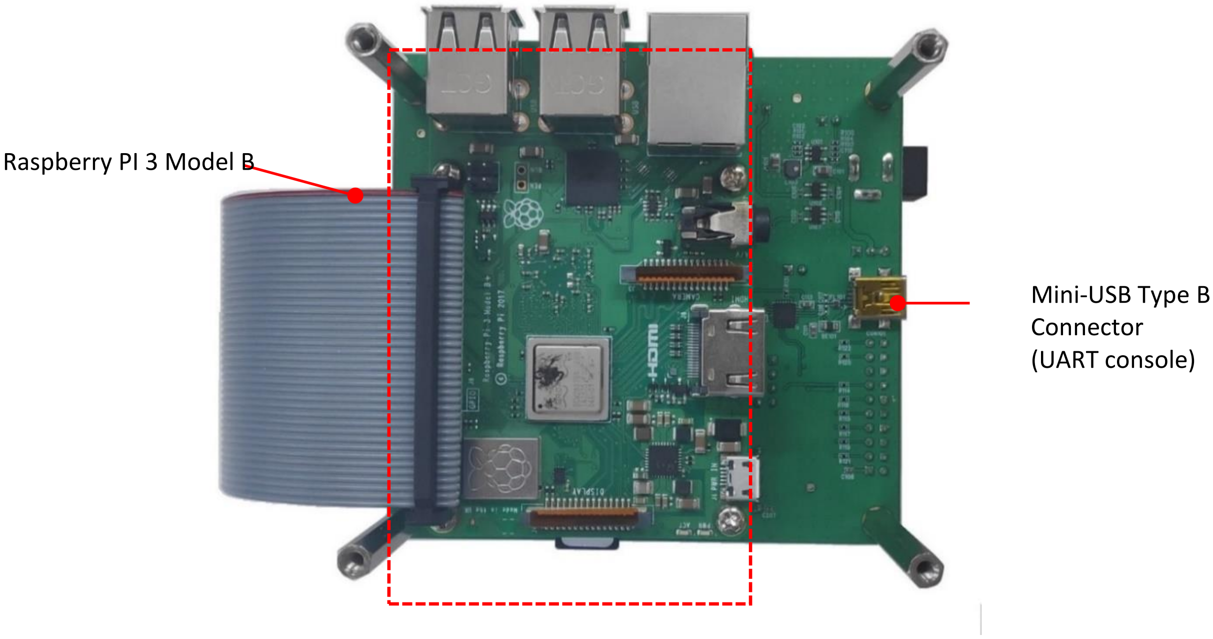


Figure 1.3 NRC7292 evaluation board (v2.0 – bottom view)

1.1 H/W list

The NRC7292 EVB consists of three components: an 11ah Wi-Fi module, an adapter board, and a host board.

1.1.1 NRC7292 module board

The NRC7292 module contains the IEEE 802.11ah Wi-Fi SoC solution. It also includes an RF front end module to amplify the transmission power up to 23 dBm. The on-board serial flash memory can be used for Over-The-Air (OTA) firmware update, user configuration data storage. The module also supports Execution-In-Place (XIP) functionality along with the 32KB cache in the SoC.

1.1.2 NRC7292 adapter board

The NRC7292 adapter board mainly acts as a communication interface to sensors or an external host. It also supplies main power to the NRC7292 Wi-Fi module.

1.1.3 Host board

The use of a Raspberry Pi3 (RPi3) or Raspberry Pi4 (RPi4) host is optional for standalone operation. The NRC7292 module can be used either as a standalone or a slave to a host processor via serial peripheral interface (SPI) or universal asynchronous receive transmitter (UART). The Raspberry Pi host can be used for test or AT-command operation.

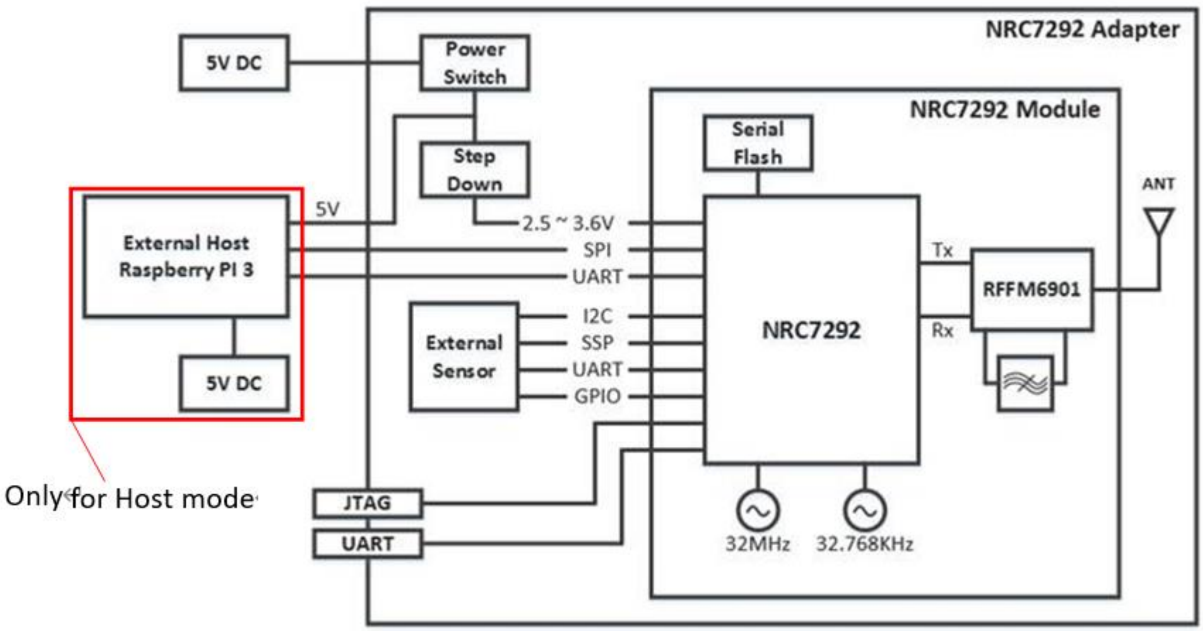


Figure 1.4 NRC7292 evaluation board block diagram

1.2 S/W list

1.2.1 NRC7292 SDK

The NRC7292 SDK can be used to develop user's application program running on NRC7292 EVB. The SDK includes various types of Application Program Interfaces (APIs) for controlling Wi-Fi connectivity, Transport Control Protocol/Internet Protocol (TCP/IP) communication, peripherals, timer, etc. In addition, users can attach various sensors on the evaluation board and communicate with them via UART, SPI, or I2C APIs.

As all standalone applications run on FreeRTOS, users can take advantage of FreeRTOS features including multi-tasking, Inter-Task Communication (ITC), memory management, etc.
(Refer to at https://www.freertos.org for more information)

2 Setup S/W build environment

The NRC7292 SDK supports a Linux environment. This chapter describes how to set up the development environment, build a user's application program, and download the binary on the EVB.

2.1 Toolchain setup

GNU ARM embedded toolchain is required to build the user's application program. Note that users should use 64-bit Linux machine to build successfully.

- Ubuntu 16.04 LTS(64-bit PC(AMD64) desktop image) or later
- GCC toolchain for ARM embedded processors
- Download the GNU Arm embedded toolchain
  - ***gcc-arm-none-eabi-10.3-2021.10-x86_64-linux.tar.bz2*** (version must exactly match)
    https://developer.arm.com/downloads/-/gnu-rm

Before installing the ARM embedded toolchain, users need some additional packages which can be easily installed through the standard package manager (apt-get) for Ubuntu. The following instructions discuss which packages are required, with instructions on how to install them.

```
sudo apt-get update
sudo apt-get install build-essential python2.7 python-pip git lzop
```

Once the required packages are successfully installed, download the GCC toolchain from the ARM developer website, copy the file to $HOME location, and extract it.

```
tar -xvf gcc-arm-none-eabi-10.3-2021.10-x86_64-linux.tar.bz2
```

The GCC toolchain will be extracted into ~/ gcc-arm-none-eabi-10.3-2021.10 directory. If the PATH environmental variable is set as shown below, users can run the GCC toolchai n anywhere without giving the complete path for the toolchain.

```
export PATH="$PATH:$HOME/gcc-arm-none-eabi-10.3-2021.10/bin"
```

2.2 Download SDK

Users can download the NRC7292 SDK from GitHub (https://github.com/newracom/nrc7292_sdk.git).

```
git clone https://github.com/newracom/nrc7292_sdk.git
```

The NRC7292 SDK in the repository consists of several subdirectories: doc, lib, make, sdk, bdf, and tools. The doc directory contains all documents for the users, including the user guides. The lib directory holds various third-party library codes including FreeRTOS, LwIP, MbedTLS, etc. along with the SDK modem library. The make and apps directories carry makefiles and sample application programs, respectively. The tool holds the NRC7292 Standalone Firmware Downloader, AT command test tool and a firmware Flash Tool. The bdf directory contains a board data files about TX power control. The board data files depend on target hardware and country.

2.3 SDK application program

2.3.1 Sample application programs

The package provides various sample application programs in the 'nrc7292_sdk/package/standalone /sdk/apps' directory. However, the 'wifi-common' directory in the 'sdk/ apps' are not a sample program but contains header and source files for AT-command and Wi-Fi connectivity, respectively. If users want to use Wi-Fi functionalities, they must include the header files in their application program.

The 'nrc7292_sdk/package/standalone/sdk/inc' directory contains the API header files for GPIO, I2C, UART, etc. Only the header files for these APIs are provided.

2.3.2 Application program project structure

Except for the AT-command application, every project directory has the '.config', Makefile, and main source file that are the same name as the project. For example, as shown below, the sample application project for the TCP client has the main source file named 'sample_tcp_client', which is the same as its project name.

The main source file should contain the 'void user_init(void)' function that serves as the entry point of the application.

```
├──── sample_tcp_client
│     ├────.config
│     ├──── Makefile
│     └──── sample_tcp_client.c
└──── wifi_common
      ├──── wifi_common.make
      ├──── wifi_config.h
      ├──── wifi_config_setup.c
      ├──── wifi_config_setup.h
      ├──── wifi_connect_common.c
      └──── wifi_connect_common.h
```

As mentioned above, to use the Wi-Fi connectivity, users should include the header files of ‘wifi_common’ in their main source file as well as adding ‘wifi_common.make’ at the end of the Makefile as shown below. Also, the application source files must be listed following the CSRCS variable in the Makefile.

```
CSRCS += \
        sample_tcp_client.c
include $(SDK_WIFI_COMMON)/wifi_common.make
```

Some third-party libraries (CJSON, MQTT, MXML, AWS, etc.) are provided in the package. Users can easily include these libraries by selecting each of them in the '.config' file. For example, if the application requires the CJSON library, the user can simply change ‘n’ to ‘y’ in the '.config' file for use.

```
CONFIG_MQTT    = n
CONFIG_AWS     = n
...........
```

2.3.3 Build application program

Users can use the ‘make’ command at the standalone directory (nrc7292_sdk/package/standalone) to build the application program. Before running the ‘make’ command, however, users must create the build-target file (.build-target) which specifies the makefile used for build and the name of the application project.

(Ex) build-target file:

```
MAKEFILE = nrc7292.sdk.release
PARAM := -- APP_NAME=sample_tcp_client
```

Users can create the build-target file by following the instruction below at the standalone directory.

Usage of make command for build-target file:

```
make select target=nrc7292.sdk.release APP_NAME=($APP NAME)
```

For the general application programs, users need to give the name of the application project as the APP_NAME. For example, to build a sample TCP client application, users can write a command as shown below.

```
make select target=nrc7292.sdk.release APP_NAME=sample_tcp_client
```

Once the build-target file is created at the standalone directory, users can run the 'make' command at the same directory. This command will generate the map, elf, and unified binary file of the application program at the 'out/nrc7292/standalone_xip/{project_name}' directory.

The binary file 'nrc7292_standalone_xip_{project_name}.bin' can then be downloaded onto the module.

2.3.4 SDK APIs

Various SDK APIs in several categories are provided for user application programming as shown in Table 2.1. Please refer to UG-7292-005-Standalone SDK API in the packet for more information.

◈To print out logs via UART console (see Figure 1.1 and Figure 1.3), the debug UART console must first be enabled by calling the API function, "nrc_uart_console_enable()." Once the console is enabled, users can print the logs out to the UART console using the API function, "nrc_usr_print()."

Table 2.1 NRC7292 SDK APIs

Category	Description
Wi-Fi	Wi-Fi connection
System	System configuration and Log level
Timer	Timer-based application
UART	UART peripheral I/O
GPIO	GPIO peripheral I/O

I2C	I2C peripheral I/O
ADC	ADC peripheral I/O
PWM	PWM peripheral I/O
SPI	SPI peripheral I/O
HTTP Client	HTTP Client
FOTA	Firmware Over-The-Air
Power Save	Sleep mode (Modem sleep / Deep sleep)
WPS_PBC	WPS pushbutton
System	System configuration and

2.3.5 Sample applications

Table 2.2 provides the information of the various sample application programs included in the release package.

Table 2.2 Sample applications

Category	Name	Description
Helloworld	hello_world	Repeatedly print hello message
Wi-Fi	sample_wifi_state	Repeat Wi-Fi connection and disconnection every 3 seconds
	sample_wps_pbc	Connect and AP with WPS-PBC
	sample_w5500_eth	Ethernet bridge with W5500 (spi)
	sample_softap_udp_server	Run SoftAP and receive UDP data
	sample_softap_tcp_server	Run SoftAP and receive TCP data
	sample_fota	Run FOTA operation
Protocol	sample_udp_client	Send UDP packets
	sample_udp_server	Receive UDP packets
	sample_tcp_client	Connect to a TCP server and send packets to the TCP server
	sample_tcp_server	Start a TCP server, wait for an incoming TCP client connection and receive data from the connected TCP client
Power Save	sample_ps_standalone	Deep sleep operation(i2c)
	sample_ps_tcp_client	Repeatedly send TCP data and enter the deep sleep mode
Peripheral	sample_timer	Start 2 timers with different periods
	sample_gpio	LED is blinking on board
	sample_uart	Bytes fed into UART CH2
	sample_adc	Communicate with a sensor via ADC

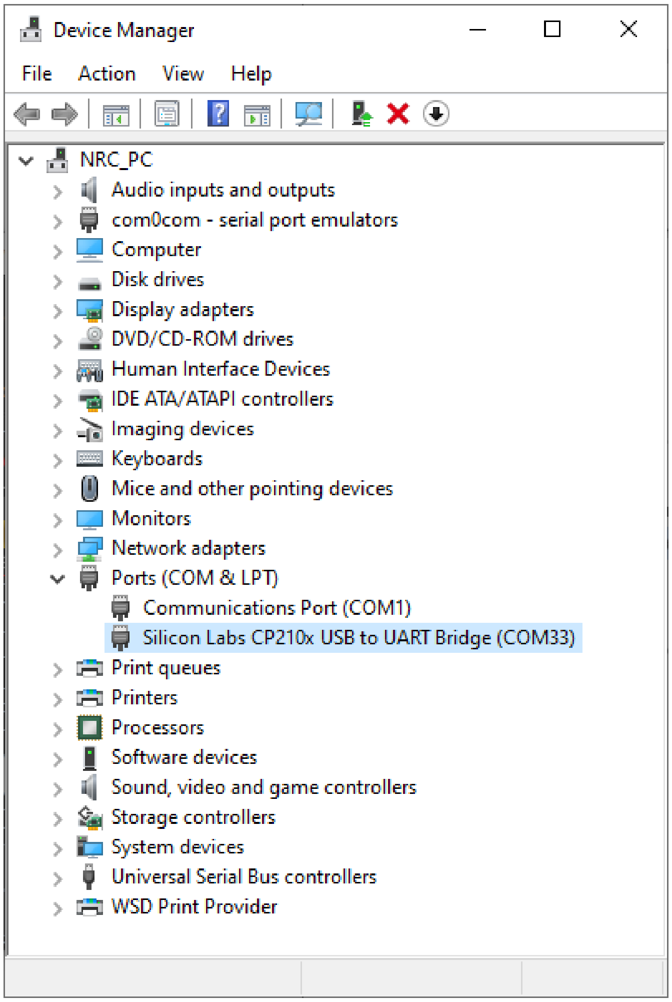
	sample_nvs	Use NVS(Non-volatile Storage) library
	sample_pwm	Enable PWM and configure the PWM duty cycle
	sample_bme680_sensor	Temperature sensor(spi/i2c)
	sample_sgp30_sensor	Air quality sensor(i2c)
	sample_sht30_sensor	Humidity sensor(i2c)
	sample_epd_2in66b	E-paper(i2c)
	sample_hink_e116a07	E-paper(i2c)
	sample_ssd1306	OLED(i2c)
	sample_xa1110_gps	GPS module sample(i2c)
Middleware	sample_xml	Test XML creation and conversion behavior
	sample_json	Test JSON creation and conversion behavior
	sample_aws_iot_sensor	Connect to AWS(Amazon Web Service) and publish message
	Sample_mqtt	Send data to MQTT server using MQTT protocol
	sample_http	Send a HTTP request and receive the corresponding response
	sample_http_server	Run SoftAP with HTTP server and then run as STA after submitting WLAN information
User Scenarios	sample_ps_schedule	It wakes up every set time and transmits sensor data.
	sample_cmd_user	Get the input data via UART and handle the data

3 How to download compiled binaries

The NRC7292 Standalone Firmware Downloader in the 'tool' directory can be used to download the unified binary onto the EVB. The steps outlined below explain how to download the binary.

3.1 UART connection between PC and EVB

Connect the PC to the EVB using a UART-USB cable and check the corresponding COM port number using the Device Manager. The COM port number will be required in the next step.



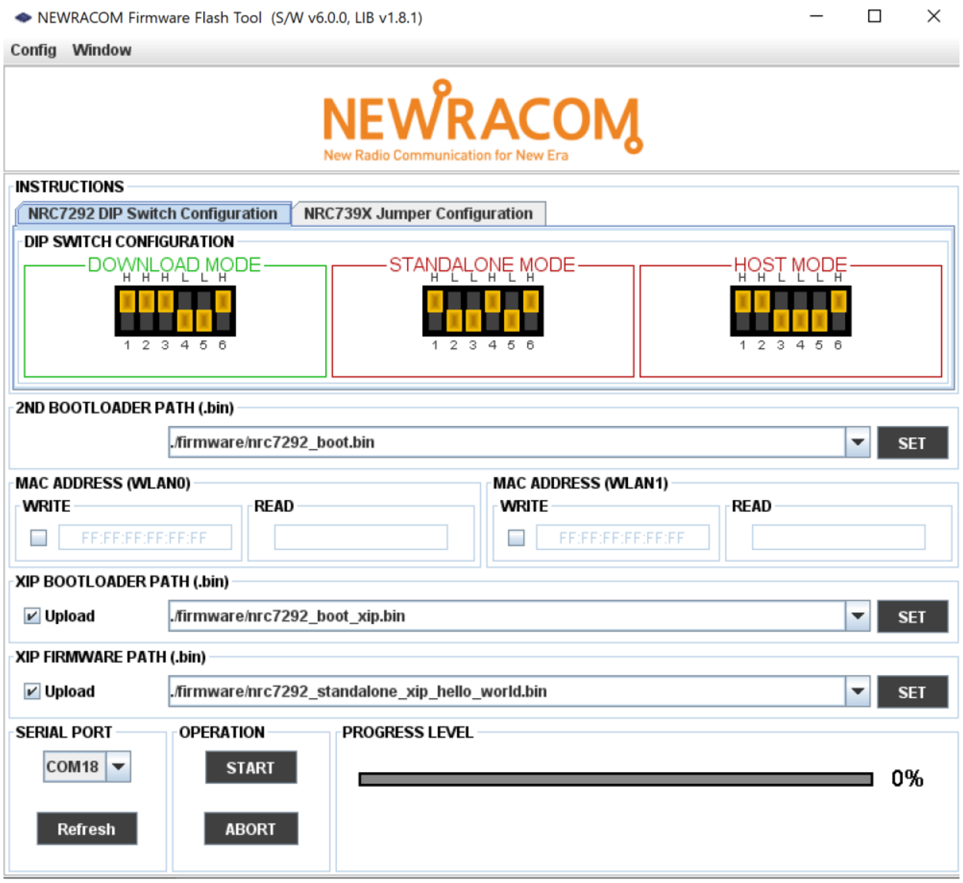
- NRC_PC
  - Audio inputs and outputs
  - com0com - serial port emulators
  - Computer
  - Disk drives
  - Display adapters
  - DVD/CD-ROM drives
  - Human Interface Devices
  - IDE ATA/ATAPI controllers
  - Imaging devices
  - Keyboards
  - Mice and other pointing devices
  - Monitors
  - Network adapters
  - Ports (COM & LPT)
    - Communications Port (COM1)
    - Silicon Labs CP210x USB to UART Bridge (COM33)
  - Print queues
  - Printers
  - Processors
  - Software devices
  - Sound, video and game controllers
  - Storage controllers
  - System devices
  - Universal Serial Bus controllers
  - WSD Print Provider

Figure 3.1 COM port in Device Manager

3.2 Upload the unified binary

Launch the NRC7292 Standalone Firmware flash tool and select the correct serial port. Either directly type in the path to the standalone XIP boot and firmware binary or press the ‘**SET**’ button to launch the file selector. The initial bootloader and XIP Boot is located in ‘./firmware/’ folder and assigned path automatically. So, the developer does not need to change boot path. MAC addresses (for WLAN0 and WLAN1) can be read from the flash.

Before click the download f/w, the developer should be change the DIP Switch mode to DOWNLOAD MODE, ‘HHHLLH’ in advance. The NRC7292 Standalone Firmware Downloader for Windows included in the release package can be used to download a unified binary onto the flash memory on the EVB. The guide document is included in a tool folder, ‘tools/external/docs/index.html’.



NEWRACOM Firmware Flash Tool (S/W v6.0.0, LIB v1.8.1)

2ND BOOTLOADER PATH (.bin): ./firmware/nrc7292_boot.bin

MAC ADDRESS (WLAN0) WRITE □ FF:FF:FF:FF:FF:FF READ

MAC ADDRESS (WLAN1) WRITE □ FF:FF:FF:FF:FF:FF READ

XIP BOOTLOADER PATH (.bin): ☑ Upload ./firmware/nrc7292_boot_xip.bin

XIP FIRMWARE PATH (.bin): ☑ Upload ./firmware/nrc7292_standalone_xip_hello_world.bin

SERIAL PORT: COM18

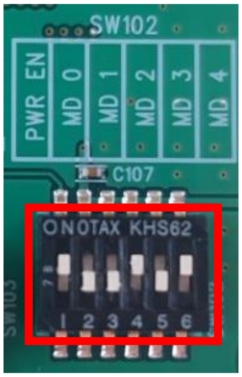
PROGRESS LEVEL: 0%

Figure 3.2 Standalone firmware downloader

To start downloading the selected binary, click the ‘**START**’ button.

3.3 Standalone operation mode

After downloading the firmware, the DIP switch must be configured to the standalone mode as shown in the figure below. Pressing the reset button on the module will start the standalone operation.



U D D UDU U(Up), D(Down)

Figure 3.3 Standalone mode DIP switch configuration

4 Performance Evaluation

Iperf is a tool for network performance measurement and tuning. It has TCP/UDP client and server functionalities and can create data streams to measure the throughput between the two ends. For performance evaluation, the NRC7292 standalone release package provides sample programs for Iperf TCP/UDP client and server, which use SDK APIs and LwIP sockets. The console command is enabled, the developer could test performance using iperf command.

4.1 Preparation of test binary

Test binary for iperf testing could be built in below and download the ‘nrc7394_standalone_xip_.bin’ in the ‘out/nrc7394/standalone_xip/standalone/’ folder.

```
make select target=nrc7292.sdk.release
make clean
make
```

4.2 Console command

The console command could be used for Wi-Fi connection and applications such as DHCP client, iperf, ping.

4.2.1 WPA

The wpa cli command is supported. Instead of ‘wpa_cli’, we use the ‘wpa’. The common comands are supported for wifi connection. The command could run such as ‘wpa [command] [args]’.

Command	args	description
wpa scan		request new BSS scan
wpa scan_results		get latest scan results
wpa add_network		add a network
wpa set_network	<network id> <variable> <value>	set network variables
wpa enable_network	<network id>	enable a network
wpa set country	<country>	set country

(Open Mode)

```
wpa set country US
wpa scan
wpa scan_results
wpa add_network
wpa set_network 0 ssid "AP_SSID"
wpa set_network 0 key_mgmt NONE
wpa enable_network 0
```

(WPA2 Mode)

```
wpa set country US
wpa scan
wpa scan_results
wpa add_network
wpa set_network 0 ssid "AP_SSID"
wpa set_network 0 key_mgmt WPA-PSK
wpa set_network 0 psk "PASSWORD"
wpa enable_network 0
```

(WPA2 Mode)

```
wpa set country US
wpa scan
wpa scan_results
wpa add_network
wpa set_network 0 ssid "AP_SSID"
wpa set_network 0 proto RSN
wpa set_network 0 ieee80211w 2
wpa set_network 0 key_mgmt SAE
wpa set_network 0 sae_password "12345678"
wpa enable_network 0
```

(WPA3-OWE Mode)

```
wpa set country US
wpa scan
wpa scan_results
wpa add_network
wpa set_network 0 ssid "AP_SSID"
wpa set_network 0 proto RSN
wpa set_network 0 ieee80211w 2
wpa set_network 0 key_mgmt OWE
wpa set_network 0 owe_only 0
wpa enable_network 0
```

4.2.2 DHCP

The dhcp command is used for getting IP via DHCP client from DHCP server.

Command	args	description
dhcp		request ip address

4.2.3 IPERF

The iperf command for testing throughput. This application based on only iperf, not iperf3. It supports some madatory options.

Command	args	description
iperf	[-s|-c host] [options] * [options] -b : bandwidth -p : port -t : time ※for stopping iperf based on [-s|-c host] [options]	iperf tcp/udp server&client.

	(ex) For stopping the operation, please us 'stop' in below [Start UDP server] iperf -s <host> -u [Stop UDP server] iperf -s <host> -u stop	

4.2.4 PING

The ping command is used for testing connection.

Command	args	description
ping	-s : symbol size -c: ping number -t: ping time	send ICMP packet for testing connection

5 Abbreviations and acronyms

Abbreviations Acronyms	Definition
ADC	Analog Digital Converter
AP	Access Point
API	Application Program Interface
AWS	Amazon Wed Service
CJSON	C JavaScript Object Notation
EVB	Evaluation Board
EVK	Evaluation Kit
FEM	Front End Module
FOTA	Firmware Over the Air
GPIO	General Purpose Input Output
HTTP	Hypertext Transfer Protocol
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
ITC	Inter-Task Communication
I2C	Inter-Integrated Circuit
LAN	Local Area Network
LwIP	Lightweight Internet Protocol
LED	Light Emitting Diode
MQTT	Message Queuing Telemetry Transport
MXML	Music Extensible Markup Language
OTA	Over-the-Air
PWM	Pulse Width Modulation
RPi3	Raspberry Pi 3
RTOS	Real Time Operating System
SDK	Software Development Kit
SoC	System on Chip
SPI	Serial Peripheral Interface
STA	Station
TCP	Transmission Control Protocol
UART	Universal Asynchronous Receive Transmitter
UDP	User Datagram Protocol
USB	Universal Serial Bus
XIP	eXecution In Place

6 Revision history

Revision No	Date	Comments
Ver 1.0	11/01/2018	Initial version for customer review created
Ver 1.1	03/25/2019	APIs and sample App for SoftAP are added
Ver 1.2	07/02/2019	Description of Sample Applications updated
Ver 1.3	11/06/2019	Update Binary download &NRC7292 SDK directories and files
Ver 1.4	07/13/2020	Added Linux build environment and remove eclipse environment
Ver 1.5	09/15/2020	Added folder location for make select
Ver 1.6	12/04/2020	Update supported ubuntu image (16.04 64bit or later)
Ver 1.7	12/10/2020	Update performance evaluation
Ver 1.8	10/22/2021	Updated APIs and sample applications
Ver 1.9	05/24/2022	Updated sample applications
Ver 2.0	01/13/2023	Updated performance evaluation and sample application table
Ver 2.1	01/25/2023	Remove ATCMD build
Ver 2.2	02/10/2023	Remove coap, tinycbor library and samples
Ver 2.3	02/28/2023	Remove roaming samples