NRC7292 Evaluation Kit
User Guide

(Standalone)
Ultra-low power & Long-range Wi-Fi

Ver 1.4
July 13, 2020

NEWRACOM, Inc.

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

© 2020 NEWRACOM, Inc.

Office

Newracom, Inc.
25361 Commercentre Drive, Lake Forest, CA 92630 USA
http://www.newracom.com

Contents

List of Tables

List of Figures

1 Overview

This document introduces the NRC7292 Software Development Kit (SDK). The SDK allows users to build custom standalone applications that run on NRC7292 Evaluation Boards (EVB) without external hosts.
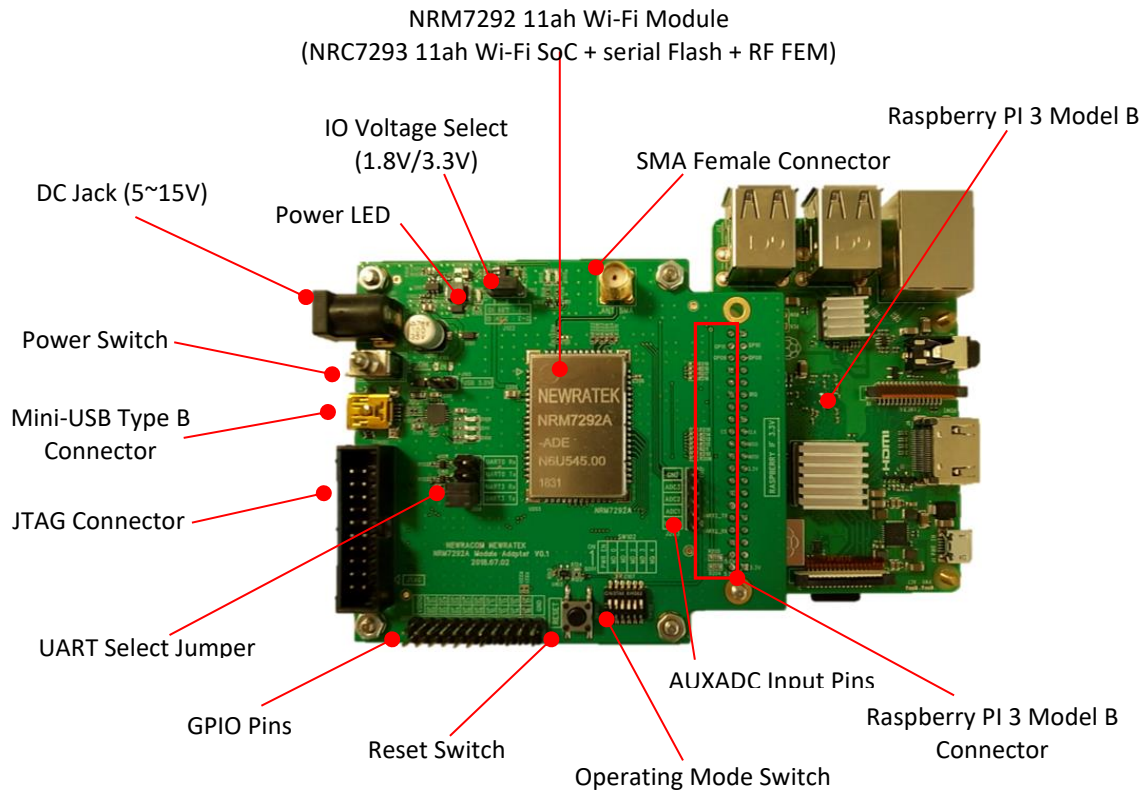
Figure 1.1   NRC7292 evaluation board (v.1.0)

NRM7292 11ah Wi-Fi Module
(NRC7293 11ah Wi-Fi SoC + serial Flash + RF FEM)

SMA Female
Antenna Connector

AUXADC Input Pins

JTAG Connector

UART Select Jumper

Power Switch

Power LED

Raspberry PI 3 Model B
Connector/Cable

DC Jack (4.5~5.5V)

Reset Switch

GPIO Pins

IO Voltage Select
(1.8V/3.3V)

FEM Power
Voltage Select
(3.3V/4.0V)

Operating Mode Switch
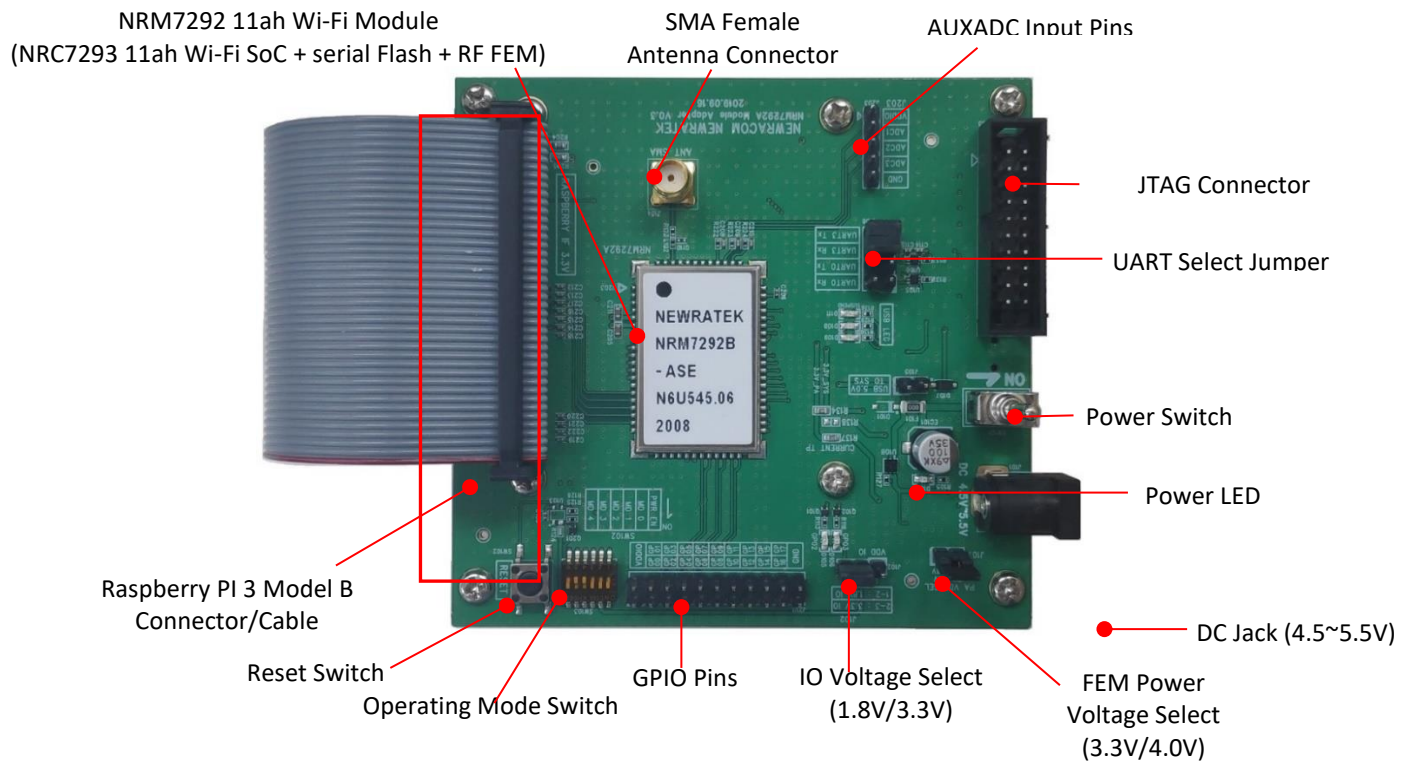
Figure 1.2   NRC7292 evaluation board (v.2.0 – top view)

Raspberry PI 3 Model B

Mini-USB Type B
Connector
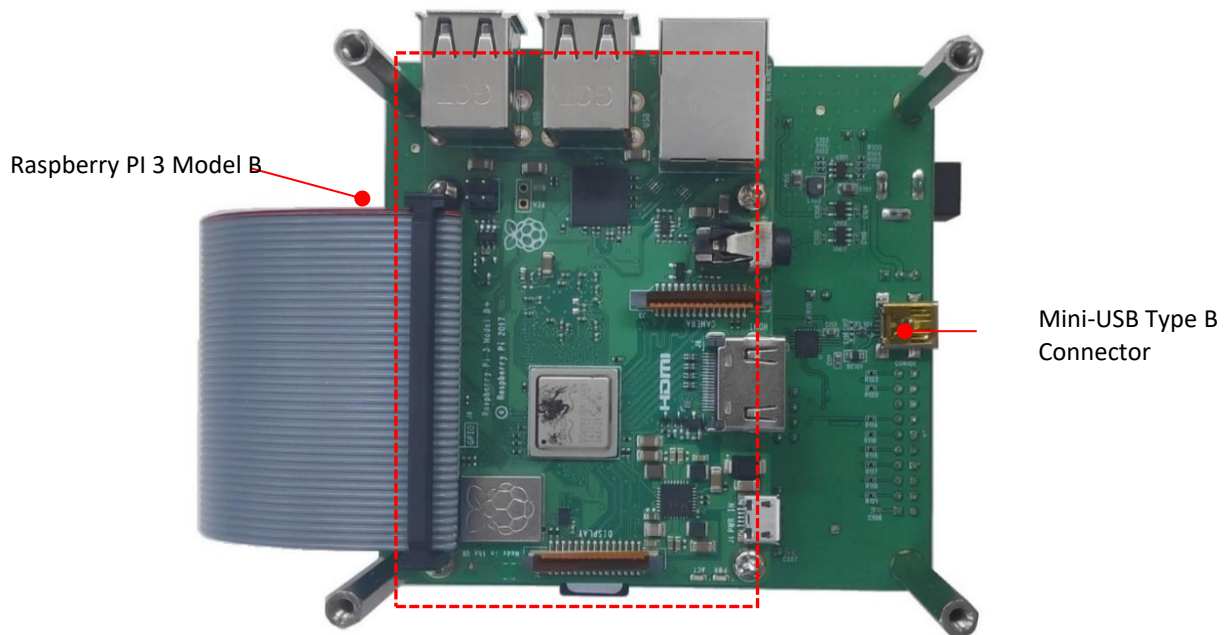
Figure 1.3   NRC7292 evaluation board (v.2.0 – bottom view)

1.1 H/W list

The NRC7292 EVK consists of three components: a module, a 5V power adapter, 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 updates, and the 32KB cache can be used for the execution-in-place (XIP) functionality and the user configuration data storage.

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 Pi 3 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 RPi 3 host can be used for both test and normal operation.
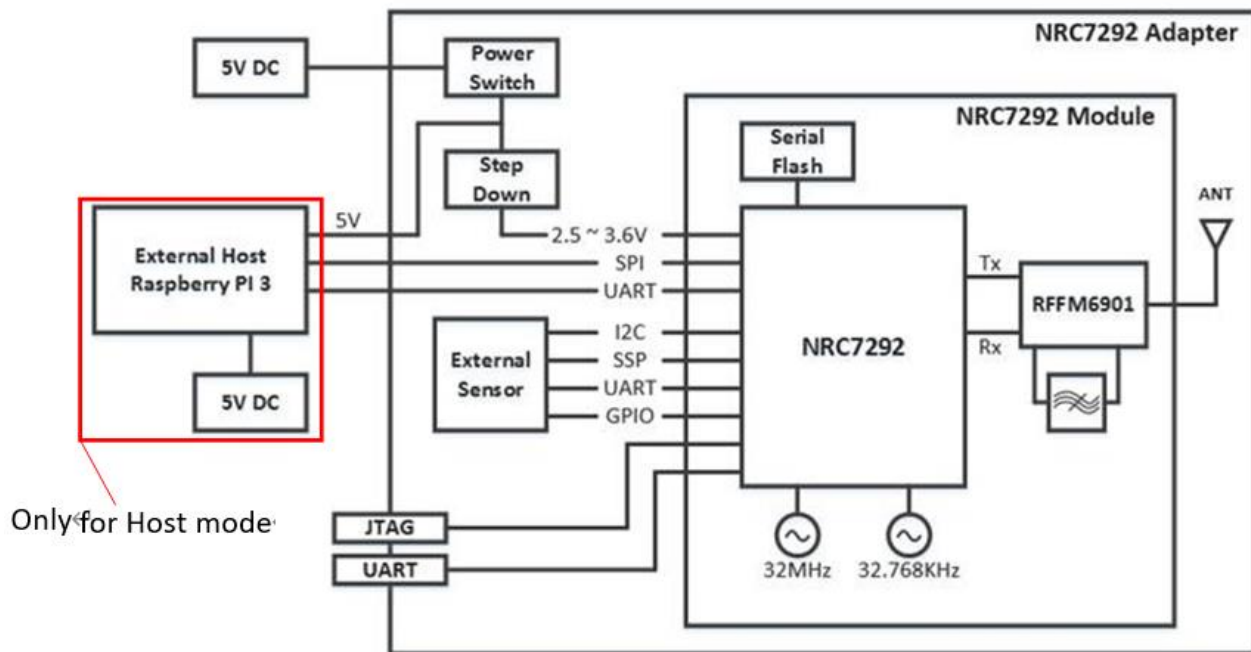
Figure 1.4   NRC7292 evaluation board block diagram

1.2 S/W list

1.2.1 NRC7292 SDK

The NRC7292 Software Development Kit (SDK) can be used to develop custom applications on evaluation boards. The SDK includes various types of APIs for controlling Wi-Fi connectivity, TCP/IP communication, peripherals, timer, memory, etc. In addition, users can attach various sensors on the evaluation boards and communicate with them with the application via UART, SPI, or I2C.

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)

1.2.2 NRC7292 binary download tool

The Binary Download Tool (BDT) for Windows included in the release package can be used to download a unified binary onto the flash memory on the EVK.

2 Setup S/W build environment

The NRC7292 Software Development Kit (SDK) supports linux environment. This document includes development environment setting, build application and download on the EVK board.

2.1 Toolchain Setup

There are a few prerequisites which must be installed on your machine before you will be able to build NRC7292 Software Development Kit (SDK).

Required:
- Ubuntu 18.04 LTS (Bionic Beaver)
- GCC Toolchain for ARM Embedded Processors
  - Download the GNU Arm Embedded Toolchain v7-2018Q2
  - gcc-arm-none-eabi-7-2018-q2-update-linux.tar.bz2 (The version must exactly match)
  - https://developer.arm.com/tools-and-software/open-source-software/developer-tools/gnu-toolchain/gnu-rm/downloads

The prerequisites 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.

The following commands will install all required and optional dependencies on Ubuntu 18.04 or later:

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

Download the GCC Toolchain (gcc-arm-none-eabi-7-2018-q2-update-linux.tar.bz2) from the arm developer website and copy the file to $HOME location and extract it.

```
tar -xvf gcc-arm-none-eabi-7-2018-q2-update-linux.tar.bz2
```

The GCC toolchain will be extracted into ~/ gcc-arm-none-eabi-7-2018-q2-update-linux/ directory.
To use it, you will need to update your PATH environment variable in ~/.bashrc file. You can change the path

```
export PATH="$PATH:$HOME/gcc-arm-none-eabi-7-2018-q2-update/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 SDK directory contained in the repository consists of several subdirectories: doc, lib, make, sdk, apps, and tool. The doc directory contains all the user guide documents. The lib directory contains various third-party library codes including FreeRTOS, lwip, mbedtls, etc along with the SDK modem library. The make directory contains makefiles. The apps directory contains sample applications and the AT-command application. The tool directory contains the binary download tool (BDT).

```
.
├── doc
├── lib
│    ├── FreeRTOS
│    ├── aws_iot
│    ├── cJSON
│    ├── hostap
│    ├── libcoap
│    ├── lwip
│    ├── mbedtls
│    ├── modem
│    ├── mxml
│    ├── paho.mqtt
│    ├── tests
│    ├── tinycbor
│    └── tr6260
├── make
├── sdk
│    └── inc
│    └── apps
└── tool
     └── downloader
```

2.3 SDK applications

2.3.1 Applications

```
├────── wifi_common
├────── atcmd
├────── hello_world
├────── sample_tcp_client
├────── sample_tcp_server
├────── .......................................
└────── sample_xml
```

The 'sdk/apps' directory contains various sample standalone applications and the AT-command application 'atcmd'. The 'sdk/inc' directory contains the API header files (GPIO, I2C, UART, etc).

The wifi_common directory contains header/source files to be included for configuring and enabling the WiFi connectivity procedure, but it is not a sample application. These header files must be included in any other sample/custom application source files that require WiFi-related functions.

2.3.2 Application Project Structure

Except for the AT-command application 'atcmd', the application directory name and the main source file name must match. The main source file must contain the 'void user_init(void)' function, which serves as the entry point of the application. For example, the application directory 'sample_tcp_client' contains the main source file 'sample_tcp_client.c', which defines the function 'user_init()'. Furthermore, because the sample application requires WiFi functions, the header files contained in the 'wifi_common' directory are included in the source file.

```
├────── 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
```

Each project directory contains the third-party library build configuration file '.config' and the makefile 'Makefile'.

The '.config' is used to specify third-party libraries (CJSON, MQTT, MXML, AWS, TINYCBOR, COAP, etc) required for the application. For example, if the application required the CJSON third-party library, the user must add the line 'CONFIG_CJSON = y' in '.config'.

```
CONFIG_CJSON = y
CONFIG_MQTT = n
CONFIG_MXML = n
CONFIG_AWS   = n
CONFIG_TINYCBOR = n
CONFIG_COAP = n
```

Note that the FREERTOS, LWIP and MBEDTLS libraries are always included in the common SDK makefile 'Makefile.nrc7292.sdk.release'.

The application source files must be listed in the application makefile 'Makefile' and appended to its CSRCS variable. It should contain {project_name.c} file which is the same name of project. If the application requires WiFi functions, the makefile must include 'wifi_common.make', as shown below.

```
CSRCS += \
        sample_tcp_client.c

include $(SDK_WIFI_COMMON)/wifi_common.make
```

2.3.3 Build Application

Except for the AT-command 'atcmd' application, the command 'make select target=nrc7292.sdk.release APP_NAME=($APP NAME)' is commonly used to configure the build option for all other sample/custom applications. For example, the following command is used to configure the build option for the application 'sample_tcp_client':

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

The AT-command application 'atcmd' build option can be configured using one of the three commands:

1. **HSPI mode:**
    *   make select target=nrc7292.sdk.release APP_NAME=ATCMD_HSPI
2. **UART mode (without hardware flow control):**
    *   make select target=nrc7292.sdk.release APP_NAME=ATCMD_UART
3. **UART mode (with hardware flow control):**
    *   make select target=nrc7292.sdk.release APP_NAME=ATCMD_UART_HFC

Running the command will create the file '.build-target'. The file specifies the target makefile name and appends the application name to an internal build parameter.

```
MAKEFILE = nrc7292.sdk.release

PARAM := -- APP_NAME=ATCMD_HSPI
```

For start the build procedure, simply type in the two commands 'make clean' and 'make'.

```
make clean
make
```

The standalone binary, map and elf files will then be generated in the directory:
'out/nrc7292/standalone_xip/{project_name}'.

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

3 How to download compiled binaries

The Windows tool NRC7292 BDT can be used to download the unified binary onto the EVK. The BDT is included in the 'tool' directory. The steps outlined below explain how to download the binary.

3.1 Download mode

Configure the DIP switch to download mode:

U U U D D U   **U(Up), D(Down)**

Figure 3.1   Standalone mode DIP switch configuration

3.2 UART connection between PC and EVK

Connect the PC to the EVK 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.
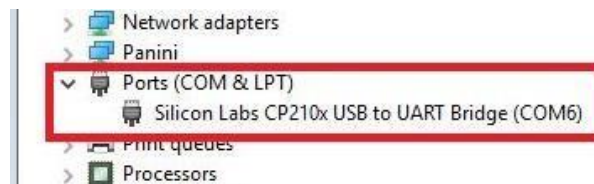

Figure 3.2   COM port in Device Manager

3.3 Download the unified binary

Launch the BDT by running the file "newrafcGUI_v2.X.X.exe". Provide the paths to the firmware and bootloader binaries, and the COM port number. If the bootloader path field is left as blank, the default path (./bootloader/boot.bin) will be used. A MAC address can also be optionally written to the flash. The checkbox underneath the MAC address field must be checked to write the provided MAC address to the flash. If the checkbox is unchecked, any existing MAC address stored in the flash will be preserved.
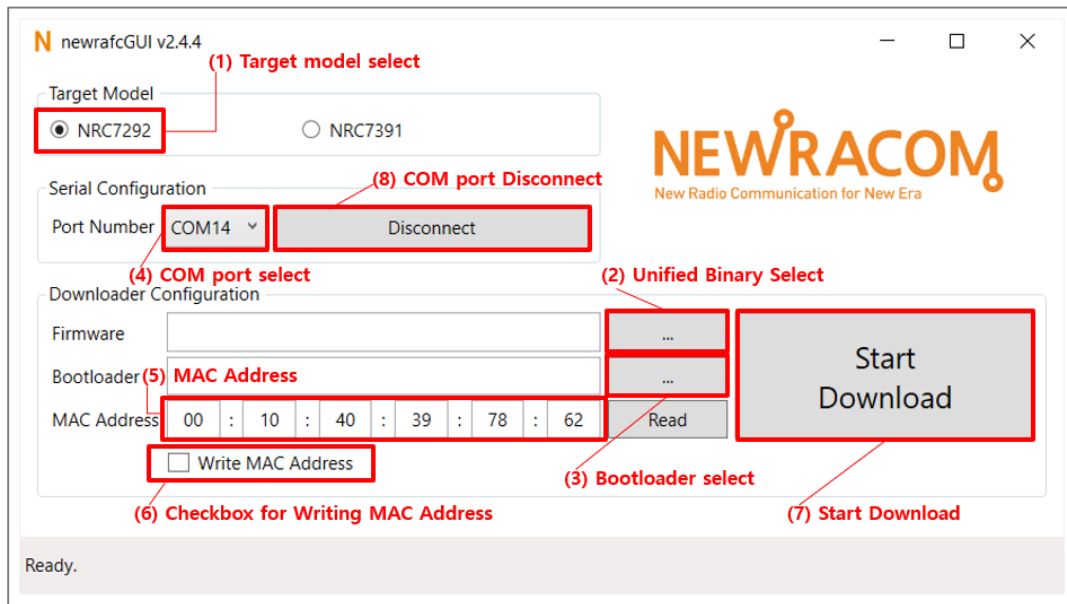
Figure 3.3   NRC7292 binary download tool

To start downloading the selected binaries, click the 'Start Download' button. If the requested download procedure cannot be completed, an error message pop-up box will appear. In case an error is encountered, make sure that:

1. The board is powered on.

2. No other process [(e.g.) a serial terminal] is occupying the selected serial COM port.

3. The DIP switch is configured to download mode.

Figure 3.4   Error message pop-up box

3.4 Standalone operation mode

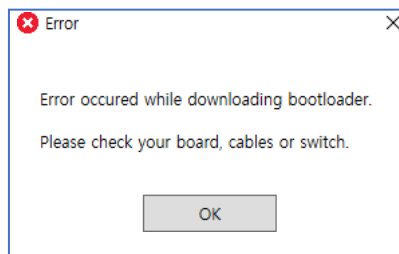Once the download procedure is complete, the DIP switch must be configured to standalone mode as shown in Figure 4.5. After changing the configuration, press the reset button on the module to start the standalone operation.
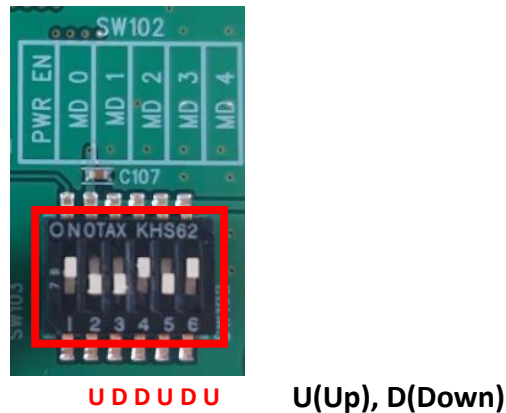

U D D U D U   U(Up), D(Down)

Figure 3.5   Standalone mode DIP switch configuration

4 How to use standalone applications with SDK APIs

The NRC7292 SDK provides sample applications for users who wish to write their own standalone applications. This section gives brief introduction about the structures of sample applications.

The NRC7292 SDK supports Wi-Fi and network functions through Wi-Fi and Connection APIs. With these APIs, developers can create applications that connect to their UDP/TCP server via Wi-Fi connection. Developers may also want to incorporate external sensors like LED, temperature sensors, gyroscopes, etc. into their IoT applications. Typically, these sensors communicate through I/O interfaces such as GPIO, UART, SPI, I2C, ADC, etc. which are all supported by the peripheral APIs.

⚠ *To print out logs via UART console, the debug UART console must first be enabled by calling the API function "nrc_uart_console_enable( )". Once the console is enabled, users can use the API function "nrc_usr_print()" to print to the console.*

Table 4.1   NRC7292 SDK APIs

API	Description
Wi-Fi	Wi-Fi connection
Protocol	Layer 3 (IP) and upper layer connection (UDP/TCP)
Memory	FreeRTOS memory resource utilization
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

4.1 Sample applications

Table 4.2   NRC7292 sample applications

Category	Subcategory	Name	Description
Helloworld	Helloworld	hello_world	Repeatedly print hello message
AT-CMD	AT-CMD	atcmd	AT command
Wi-Fi	Connection	sample_wifi_state	Repeat Wi-Fi Connection and Disconnection every 3 seconds.
	SoftAP	sample_softap_udp_server	Run SoftAP and receive udp data.
		sample_softap_tcp_server	Run SoftAP and receive tcp data.
Protocol	UDP	sample_udp_client	Send a UDP packet every 1 second
		sample_udp_server	Receive UDP packets.
	TCP	sample_tcp_client	Connect to a TCP server and send 1000 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.
	HTTP	sample_http	Send a HTTP request and receive the corresponding HTTP response.
	FOTA	sample_fota	Run FOTA operation.
Peripheral	Timer	sample_timer	Start 6 timers with different periods.
	Memory	sample_memory	Repeatedly allocate and free memory.
	GPIO	sample_gpio	Set GPIO_01(LED) as a pin and toggle on and off every 1 second. (LED blinks 1 sec.)
	UART	sample_uart	Bytes fed into UART CH2 are sent to a remote UDP server.
	SPI	sample_spi	Communicate with a sensor via SPI.
	I2C	sample_i2c	Communicate with a sensor via I2C.
	ADC	sample_adc	Communicate with a sensor via ADC.
	PWM	sample_pwm	Enable PWM and configure the PWM duty cycle for an analog sensor.
	Serial flash	sample_sf_userconfig	read/write user config area
	Power Save	sample_ps_standalone	deep sleep operation
		sample_ps_tcp_client	Repeatedly send tcp data and enter the deepsleep mode

Middleware	MQTT	Sample_mqtt	Send data to MQTT Server using MQTT protocol
	XML	sample_xml	Test XML creation and conversion behavior
	CJSON	sample_json	Test JSON creation and conversion behavior
	AWS	sample_aws_client	connetc to aws and publish message
	OneM2M	sample_onem2m_client	connetc to oneM2M and receive data via mqtt
	CoAP	sample_coap_server	Execute CoAP Server
		sample_coap_client	Execute CoAP client and get information from Server

5 Abbreviations and acronyms

Abbreviations Acronyms	Definition
ADC	Analog Digital Converter
AP	Access Point
DHCP	Dynamic Host Configuration Protocol
EVB	Evaluation Board
EVK	Evaluation Kit
FEM	Front End Module
GPIO	General Purpose Input Output
HDMI	High Definition Multimedia Interface
HW	Hardware
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
I2C	Inter-Integrated Circuit
LAN	Local Area Network
lwIP	Lightweight Internet Protocol
LED	Light Emitting Diode
NAT	Network Address Translation
PWM	Pulse Width Modulation
RTOS	Real Time Operating System
SDK	Software Development Kit
SoC	System on Chip
SPI	Serial Peripheral Interface
SSID	Service Set Identifier
STA	Station
SW	Software
TCP	Transmission Control Protocol
UART	Universal Asynchronous Receive Transmitter
UDP	User Datagram Protocol
USB	Universal Serial Bus
XIP	eXecution In Place
FOTA	Firmware Over the Air
MQTT	Message Queuing Telemetry Transport
CoAP	Constrained Application Protocol

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 eclispe environment