SRM100A EVB User Manual

Rev.06

Aug. 27, 2020

Contents

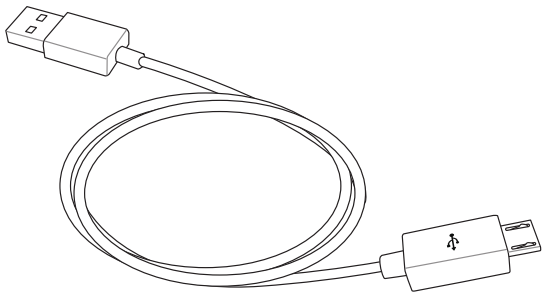
Model	F/W
SRM100A_EVB	-

Hardware

Evaluation Kit Component

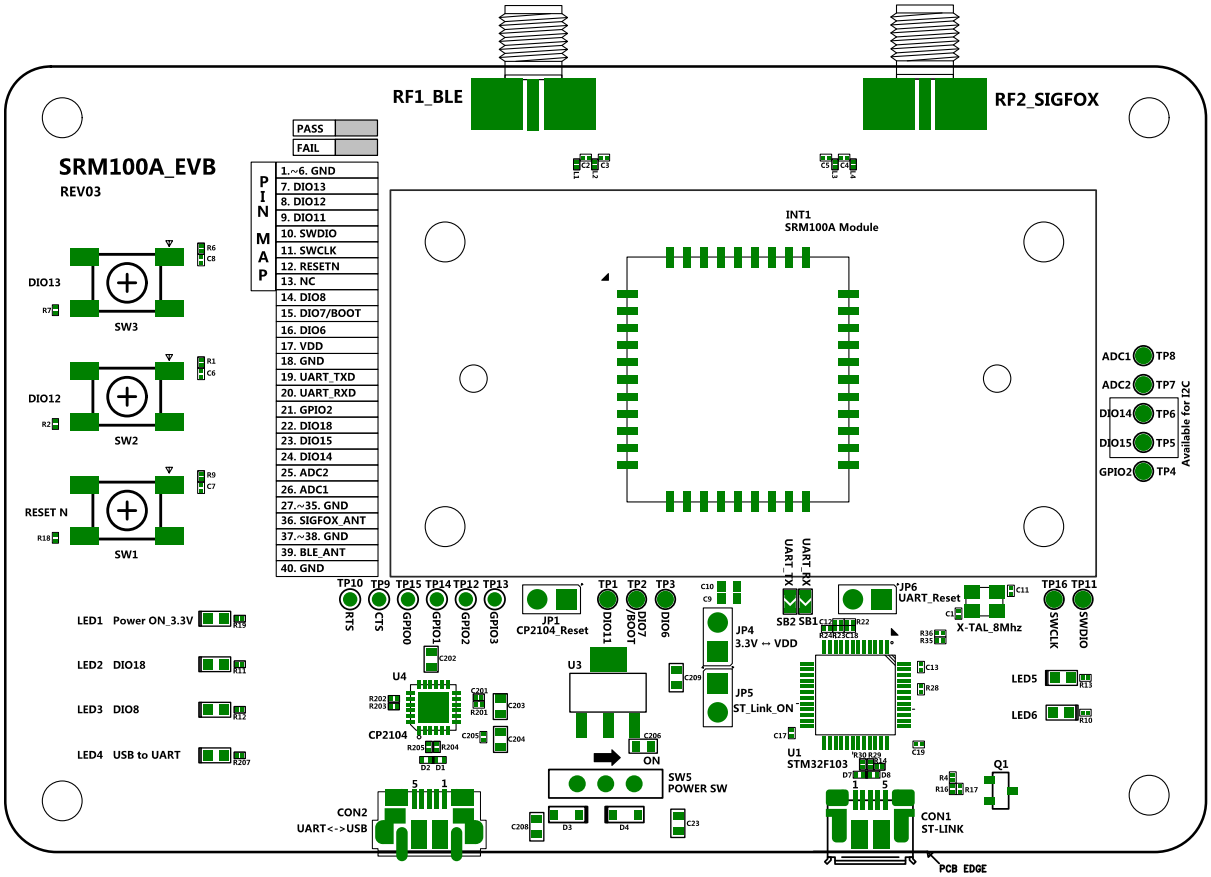
SRM100A_EVB Evaluation Kit Component

1) SRM100A_EVB(Rev.4): 1EA
2) SMA Antenna: 1EA
3) Micro USB cable: 1EA



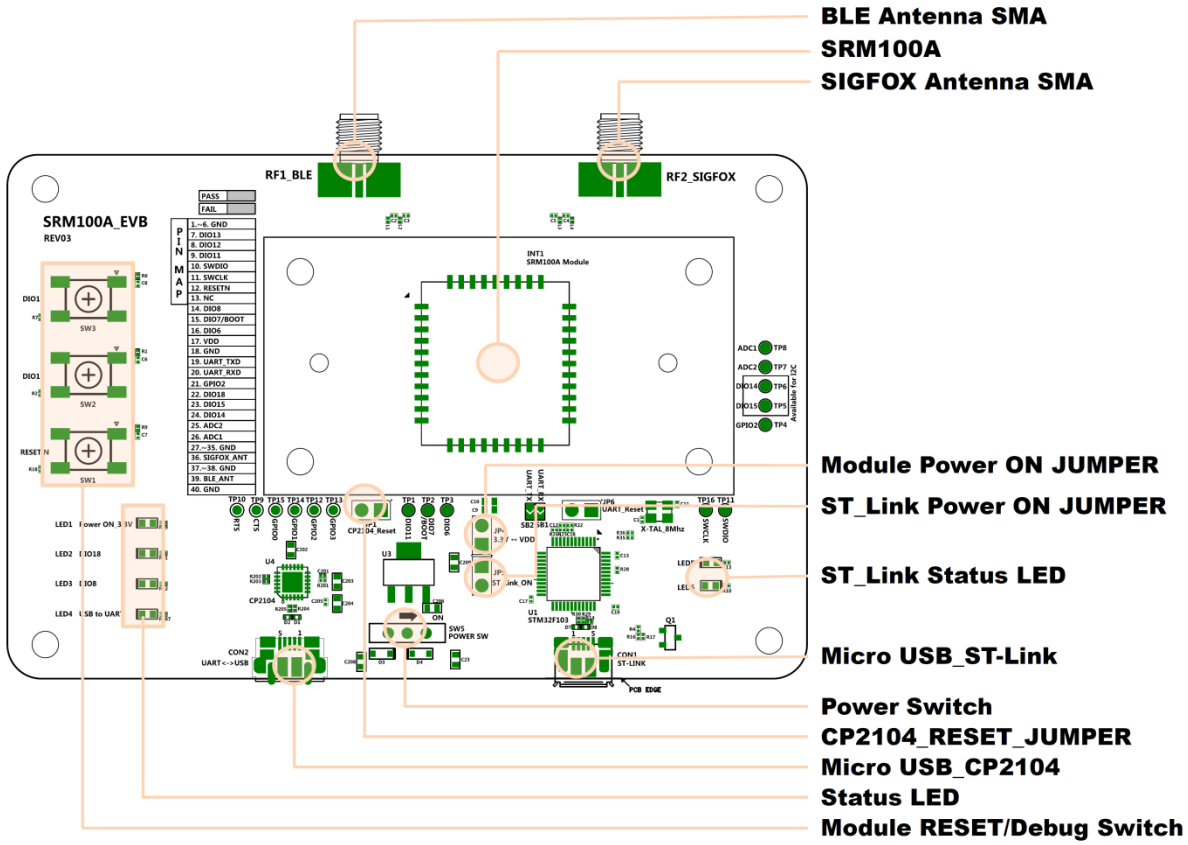
Micro USB cable

Antenna



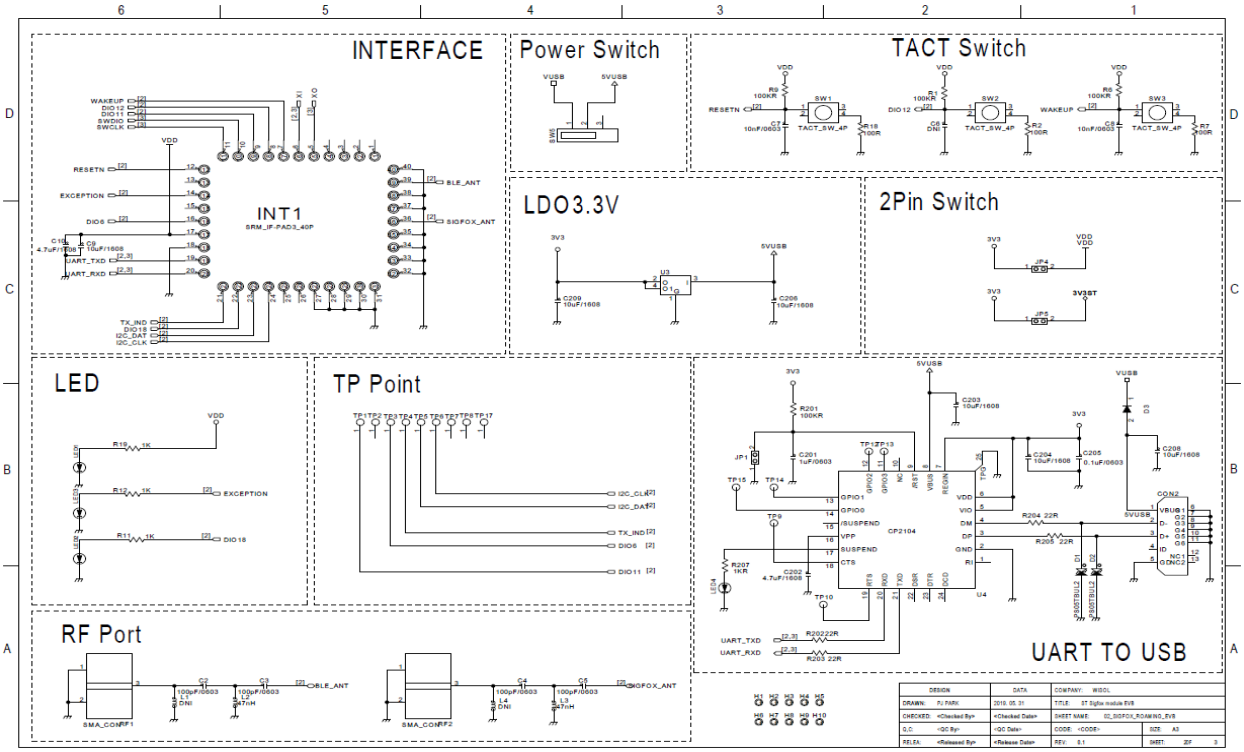
SRM100A_EVB(REV04)

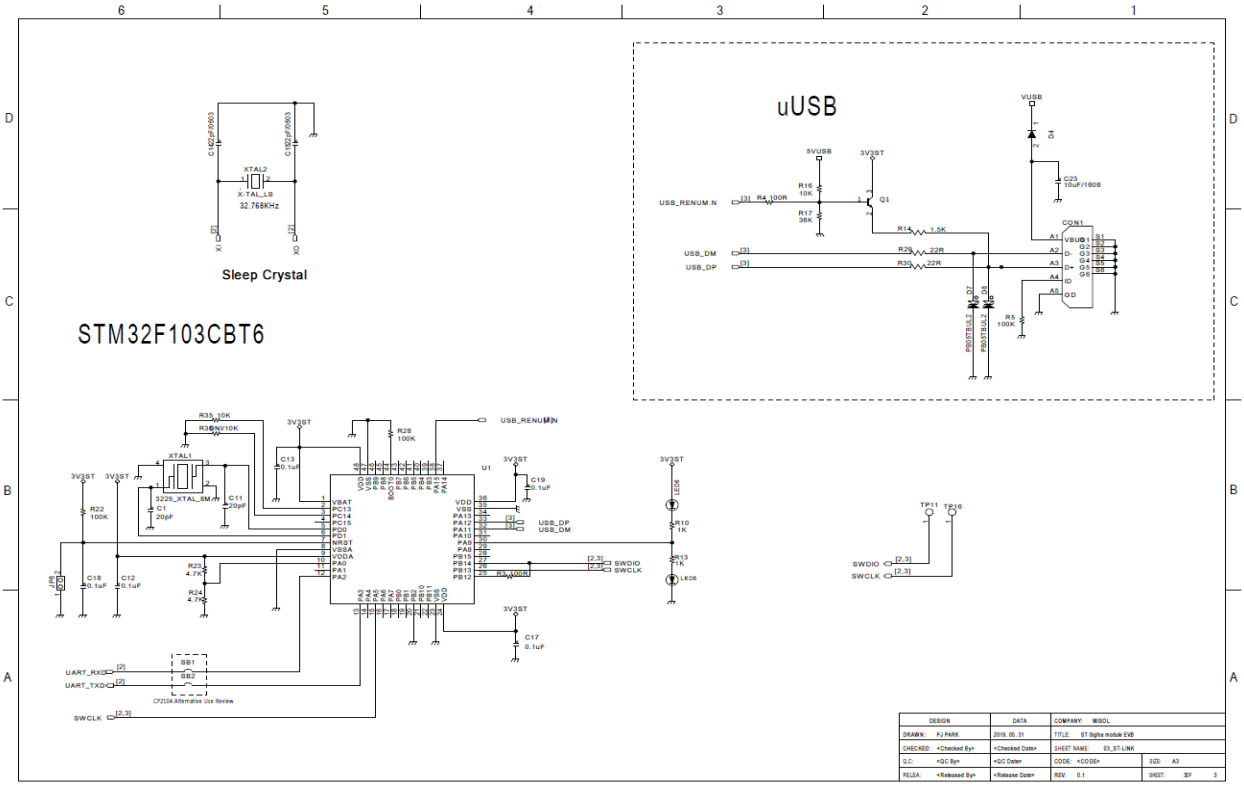
SRM100A_EVB Board



- **BLE Antenna SMA :** BLE connector for Antenna
- **SRM100A :** Sigfox Roaming module
- **WIFI Antenna SMA :** WIFI connector for Antenna
- **Module power Jumper :** SRM100A power supply jumper PIN
- **ST_Link Power on Jumper :** When downloading F/W using ST_Link
- **ST_Link Status LED :** ST_Link operation status LED
- **Micro USB_ST_Link :** Micro USB Connector
- **Power Switch :** EVB Power On/OFF Switch
- **CP2104_RESET_JUMPER** : CP2104 reset
- **Micro USB_CP2104** : Micro USB Connector
- **Staus LED** : Power On, USB to UART, I/O operation status check LED
- **Module RESET/Debug Switch** : RESET Tact Switch

Schematic


INTERFACE
Power Switch
TACT Switch
LDO3.3V
2Pin Switch
LED
TP Point
RF Port
UART TO USB
INT1
SRM_IF-PAD3_40P
SMA_CONRF1
SMA_CONRF2


uUSB
Sleep Crystal
STM32F103CBT6
X-TAL_LS
32.768KHz

Connector PIN Description

NO	PIN NAME	NO	PIN NAME	NO	PIN NAME
1	N.C.	16	DIO6	31	GND
2	N.C.	17	VCC	32	GND
3	N.C.	18	GND	33	GND
4	N.C.	19	UART_TXD	34	GND
5	XO	20	UART_RXD	35	GND
6	XI	21	TX_IND	36	SIGFOX_ANT
7	WAKEUP	22	DIO18	37	GND
8	DIO12	23	I2C_DAT	38	GND
9	DIO11	24	I2C_CLK	39	BLE_ANT
10	SWDIO	25	N.C.	40	GND
11	SWCLK	26	N.C.		
12	RESETN	27	GND		
13	N.C	28	GND		
14	EXCEPTION	29	GND		
15	N.C.	30	GND		

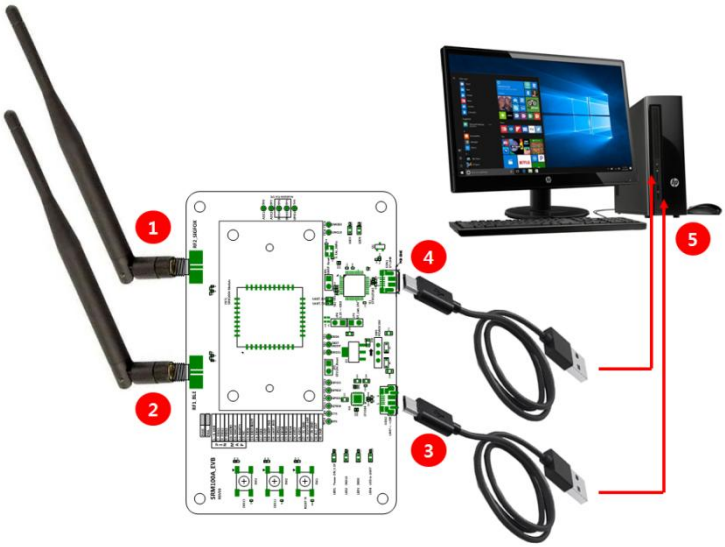
Pin No.	Pin name	Type	Description
1	N.C.	N.C	Not Connected.
2	N.C.	N.C	Not Connected.
3	N.C.	N.C	Not Connected.
4	N.C.	N.C	Not Connected.
5	XO	O	32.768kHz Sleep Crystal Output
6	XI	I	32.768kHz Sleep Crystal Input
7	WAKEUP	I	Wake up SRM100A from sleep. Edge Triggered : H -> L
8	DIO12	I/O	General purpose digital I/O
9	DIO11	I/O	General purpose digital I/O
10	SWDIO	I/O	Serial wire debug data in/output **It must be connected to an external connector or TP for Use in RF regulatory certifications.**
11	SWCLK	I	Serial wire debug clock in **It must be connected to an external connector or TP for**

			Use in RF regulatory certifications.
12	RESETN	I	System reset
13	N.C	N.C	Not Connected.
14	EXCEPTION	O	Exception Notification. Need to reset SRM100A. Exception : HIGH (Normal : LOW)
15	N.C.	N.C	Not Connected.
16	DIO6	I/O	General purpose digital I/O
17	VCC	VCC	Supply voltage input, +3.3Vdc typ.
18	GND	GND	Common ground
19	UART_TXD	O	UART Tx data
20	UART_RXD	I	UART Rx data
21	TX_IND	O	Indicate Sigfox Tx State.(Tx : HIGH, IDLE : LOW)
22	DIO18	I/O	General purpose digital I/O
23	I2C_DAT	I/O	I2C DATA (option)
24	I2C_CLK	I/O	I2C_CLK (option)
25	N.C.	N.C	Not Connected.
26	N.C.	N.C	Not Connected.
27~35	GND	GND	Common ground
36	SIGFOX_ANT	RF I/O	Sigfox RF in/out put
37,38	GND	GND	Common ground
39	BLE_ANT	RF I/O	BLE RF in/out put
40	GND	GND	Common ground

Test Program

Evaluation board Connection

1. SRM100A_EVB connect to Window PC by USB cable.

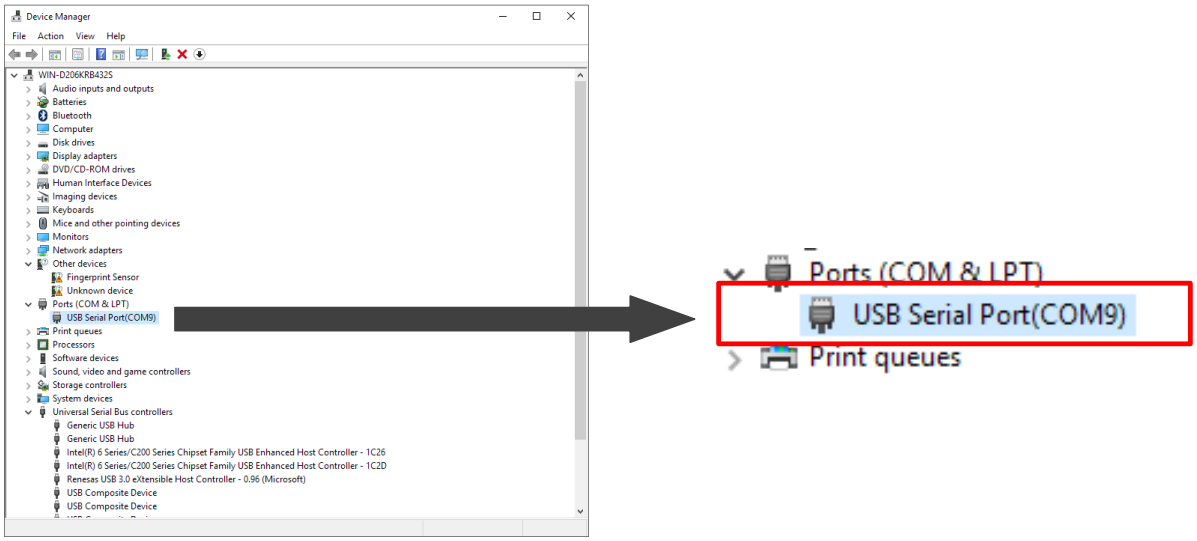


(1) SIGFOX Multi-Band Antenna
(2) BLE Antenna
(3) Micro USB cable_CP2104
(for input command and debug)
(4) Micro USB Cable_ST_Link
(for download program)
(5) PC

Program execution

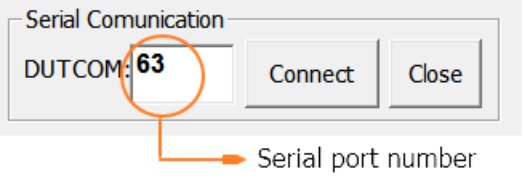
1. SRM100A_EVB connected serial-port in Windows PC, and then check the COM-port number in device manager.

- USB Serial Port(Com□□)



[Fig. SRM100A_EVB serial port]

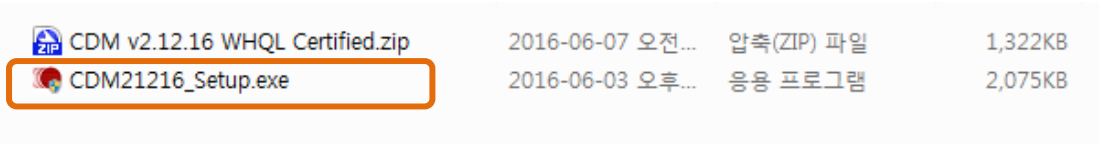
2. Run serial communication program "SRM100A_AT_TEST.exe"
3. Write serial port Number in 'DUTCOM' BOX, and then 'connect' click.



[Fig. SRM100A_EVB serial port number]

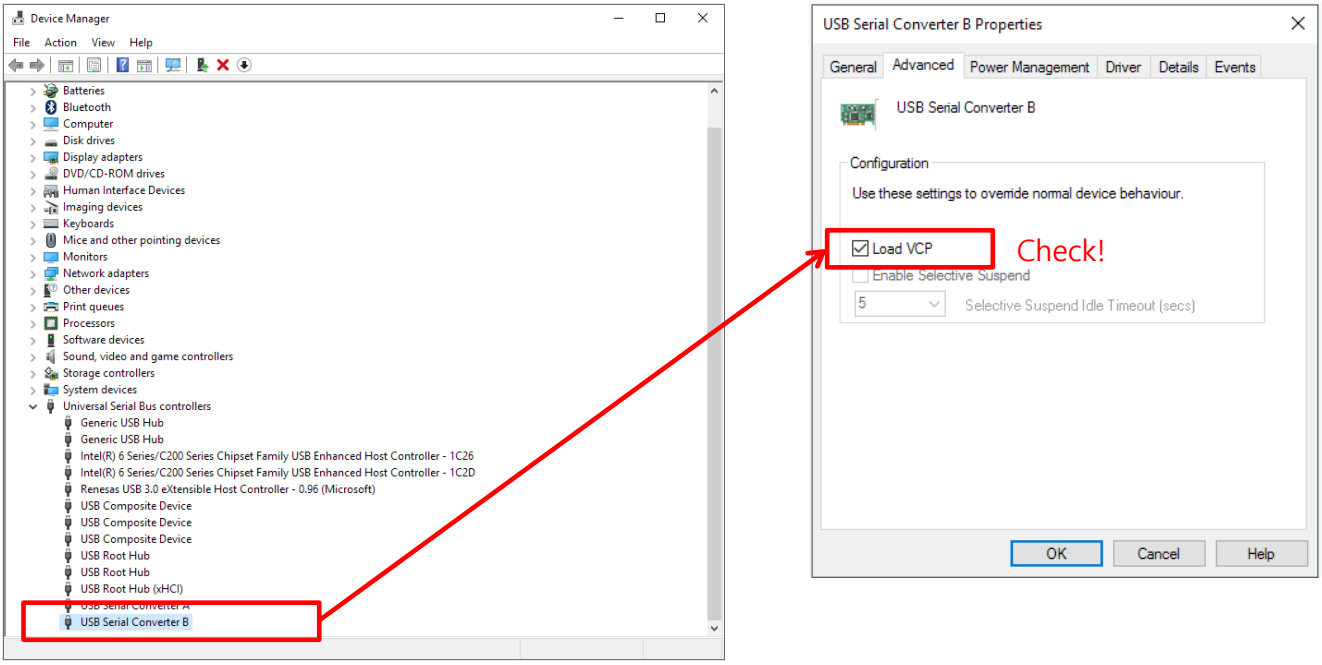
Install USB driver

1. Execute "CDM21216_Setup.exe" file.



[Fig. USB driver set-up file]

2. Setting device manager in Windows.



[Fig. Setting device manager]

Test Program

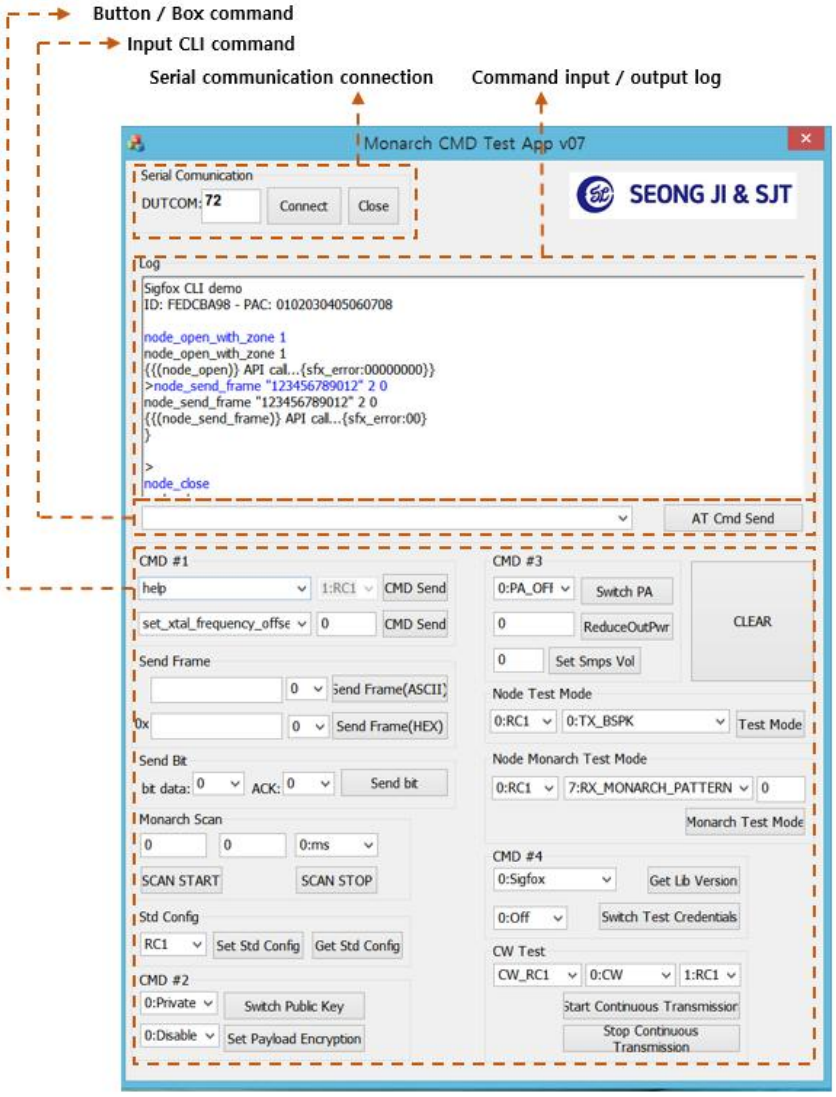
Test program Description (MONARCH_CMD_Test_App_vxx.exe)

This program is for controlling and debugging the SRM100A

You may use another serial communication terminal. However, this program is designed for the SRM100A so that you can enter commands easily.

Serial communication

- Baud Rate : 115200 bps
- Data bits: 8
- Stop bits: 1
- Parity: None



[Fig. Monarch Command Tool program**(v07)**]

CLI command set

A typical serial terminal emulator can also be used to control the EVK instead of the proposed test SW. In that case the following parameters should be used:

- Speed : 115200 bps
- Data bits: 8
- Stop bits: 1
- Parity: None
- String that specifies the number and types of arguments the command accepts.
- The argument specifiers are:
  * - u: one-byte unsigned integer.
  * - v: two-byte unsigned integer
  * - w: four-byte unsigned integer
  * - s: one-byte signed integer
  * - b: string. The argument can be entered in ascii by using quotes, for example: "foo". Or it may be entered in hex by using curly braces, for example: { 08 A1 f2 }. There must be an even number of hex digits, and spaces are ignored.
  * - n: indicates this is a 'n'ested command. The action points to a table of subcommands. If used, this must be the only specifier. It also adds one to the total argument count of the complete command.
  * - l eight-byte unsigned integer
  * Integer arguments can be either decimal or hexadecimal.
  * A 0x prefix indicates a hexadecimal integer. Example: 0x3ed.

Table 1 CLI commands

Name	Arg	Arguments description	Description
node_close		None	Closes the Sigfox library, resetting its state
node_open	u	**rc**: pointer to sfx_rc_t type representing the RC number (RC1=1, RC2=2, RC3C=3, RC4=4, RC5=5, RC6=6 or RC7=7).	
node_open_with_zone	u		
node_get_info		None	
node_get_version		**None**	
node_send_bit	uuu	**bit_value**: bit value to send (0 or 1) **tx_repeat**: tx repeat value (default : 2) **initiate_downlink_flag**: wait for a response after transmitting. (0 or 1)	This function is used to send a single bit. It is mainly used when the node seeks downlink data (and not to transmit).
node_send_frame	buu	**cust_data**: pointer to the data to transmit ex) ASCII : "12345678" Hexa : {0102030405060708} **tx_repeat**: tx repeat value (default : 2) **initiate_downlink_flag**: wait for a response after transmitting. (0 or 1)	DM00365435.pdf Please refer to page 9 of the "DM00365435.pdf" file
node_execute_monarch_scan	uuu	**rc_capability**: rc 7 6 5 4 3 2 1 bit 6 5 4 3 2 1 0 **time**: scan time **time_unit**: 0: ms, 1:sec, 2:min, 3:hour	Execute Monarch scan. rc_capability, time, unit
node_stop_monarch_scan		None	This function stops any ongoing RC scan
node_set_std_config	wwwv	**config_word1**: ch1 ~ 32 for RC2,4 **config_word2:** ch33~64 for RC2,4 **config_word3:** ch65~86 for RC2,4 **timer_enable :** (0,1) for RC2,4	DM00365435.pdf Please refer to page 10 of the "DM00365435.pdf" file
node_get_std_config		none	Get std_config value.
start_continuous_transmission	wu	**frequency**: Frequency at which the signal has to be generated **type**: Type of modulation to use in continuous mode (SFX_NO_MODULATION=0 SFX_DBPSK_100BPS=1 SFX_DBPSK_600BPS=2)	Executes a continuous wave or modulation depending on the parameter type
stop_continuous_transmission		None	Stop the current continuous transmission
node_test_mode	uu	**rc** : pointer to sfx_rc_t type representing the RC number (0, 1, 2, 3, 4, 5, 6 or 7) **test_mode** : (SFX_TEST_MODE_TX_BPSK =0 SFX_TEST_MODE_TX_PROTOCOL =1 SFX_TEST_MODE_RX_PROTOCOL =2 SFX_TEST_MODE_RX_GFSK =3 SFX_TEST_MODE_RX_SENSI =4 SFX_TEST_MODE_TX_SYNTH =5 SFX_TEST_MODE_TX_FREQ_DISTRIBUTION =6 SFX_TEST_MODE_TX_BIT=11 SFX_TEST_MODE_PUBLIC_KEY=12 SFX_TEST_MODE_NVM=13)	Sigfox test mode rc : 0 : RC1 1 : RC2 2 : RC3A 3 : RC3C 4 : RC4 5 : RC5 6 : RC6 7 : RC7
node_monarch_test_mode	uuu	**rc** : pointer to sfx_rc_t type representing the RC number (0, 1, 2, 3, 4, 5, 6 or 7). **test_mode** : (SFX_TEST_MODE_RX_MONARCH_PATTERN_LISTENING_SWEEP=7 SFX_TEST_MODE_RX_MONARCH_PATTERN_LISTENING_WINDOW=8 SFX_TEST_MODE_RX_MONARCH_BEACON=9 SFX_TEST_MODE_RX_MONARCH_SENSI=10) **rc_capability**: rc 7 6 5 4 3 2 1 bit 6 5 4 3 2 1 0	Sigfox monarch test mode rc : 0 : RC1 1 : RC2 2 : RC3A 3 : RC3C 4 : RC4 5 : RC5 6 : RC6 7 : RC7
switch_public_key	u	**key** : private=0, public=1	Switch device on public or private key.

switch_test_credentials	u	**credentials :** 1 : test ID,PAC 0 : module ID, PAC	Set test credentials 1=On, 0=Off
set_payload_encryption	u	**enc** : encryption enable : 1 disable : 0	Payload encription
node_send_oob		None	Send OOB frame
switch_pa	u	**pa** : set external power amplifiler (1 if a PA – RC2 or RC4 , 0 if no PA – RC1,RC3,RC5,RC6,RC7).	Instructs the library to configure the S2-LP for a external PA (Power Amplifier).
set_rssi_offset	u	**rssi_value** : Rssi offset value in dB	Set an RSSI offset for the RSSI. Very useful if the RF frontend has an LNA or to calibrate the RSSI measurement.
get_rssi_offset		None	Get the RSSI offset for the RSSI
set_frequency_offset	w	**xtal** : xtal value in Hz	Sets the Vender frequency of the S2-LP in Hertz (default is 50MHz).
get_frequency_offset		None	Get Vender frequency
get_id		None	ID stored in the current node
get_pac		None	PAC stored in the current node
get_rcz		None	RCZ stored in the current node
get_lib_version	u	**lib_ver** : 0 : Sigfox, 1 : MCU_API 2 : RF_API, 5 : MONARCH_API 6 : DEVICE_CONFIG_API	Get version of specified module
set_rcz	u	**rc**: pointer to sfx_rc_t type representing the RC number (RC1=1, RC2=2, RC3C=3, RC4=4, RC5=5, RC6=6 or RC7=7).	Set rc
get_swver		None	Get software version
sleep		None	Sleep mode(To wake up, make the WAKEUP pin falling edge)
ble_get_mac		None	Return MAC address
ble_set_beacon_data	b	**advertising_data :** Max 21byte	Set the advertising data. hex: *ble_set_beacon_data {000102030405060708090a0b0c0d0e0f1011121314}* *string:* *ble_set_beacon_data "123456789012345678901"*
ble_send_noti_Character	b	**notification_data :** Max 20byte	Set the notification data. hex: *ble_send_noti_Character {000102030405060708090a0b0c0d}* *string:* *ble_send_noti_Character "12345678901234"*
ble_set_read_Character	b	**read_data :** Max 20byte	Set the read data. (Same as notification data) hex: *ble_set_read_Character {000102030405060708090a0b0c0d}* *string:* *ble_set_read_Character "12345678901234"*
ble_start	uvv	**adv_type**: 0: Connectable undirected advertising 1: Connectable directed advertising 2: Scannable undirected advertising 3: Non connectable undirected advertising **Advertising_Interval_Max**: 32(20.000 ms)~ 16384(10240.000 ms) **Advertising_Interval_Min**: 32(20.000 ms)~ 16384(10240.000 ms)	Start ble for the option. In the connected mode, the write value is output to Serial.. *modified_event : 0x00 0x00…(8byte)*
ble_set_tx_power_lvl	uu	**high_power** : 0-disalbe 1-enable **level** : 0: -14 dBm (High Power) 1: -11 dBm (High Power) 2: -8 dBm (High Power) 3: -5 dBm (High Power) 4: -2 dBm (High Power) 5: 2 dBm (High Power) 6: 4 dBm (High Power) 7: 8 dBm (High Power)	Set the power of tx
ble_test_tx	uuu	**Frequency**: 0(2042MHz)~39(2480MHz) **Length**:0-255 **Payload**: 0: Pseudo-Random bit sequence 9 1: Pattern of alternating bits '11110000' 2: Pattern of alternating bits '10101010'	Start ble tx test.

		3: Pseudo-Random bit sequence 15 4: Pattern of All '1' bits 5: Pattern of All '0' bits 6: Pattern of alternating bits '00001111' 7: Pattern of alternating bits '0101'	
ble_test_rx	u	**Frequency**: 0(2042MHz)~39(2480MHz)	Start ble rx test.
ble_test_stop		None	Stop ble test. Returns the number of received packets.
ble_tone_start	u	**Frequency**: 0(2042MHz)~39(2480MHz)	Start the ble tone test.
ble_tone_stop		None	Stop the ble tone test.
ble_reset		None	Reset the ble.

Getting started

Preparations

In order to test the basic functions of SRM100A, please first check followings.

1 ID & PAC

ID & PAC is the information for identifying the Sigfox device and registering to Sigfox network. You can read this information from SRM100A as below.

1.1 ID

This can be read by the command, "*get_id*"

1.2 PAC

This can be read by the command, "*get_pac*"

2 Sigfox network

This document assumes that you have purchased the SRM100A EVB, and your location is in Sigfox coverage. In this case, just send the ID & PAC information to your local contact person who is in charge of SRM100A. Seongji will do the network registration with this information, and let you know. Then you can access the Sigfox network.

Note.

The process above is only for SRM100A EVB. In general case, you should contact to your local Sigfox SO(Service Operator), in order to register your device to Sigfox network.

If the Sigfox network is not available for any reason, you can consider to use the Sigfox network emulator. Please refer to the link, https://build.sigfox.com/sdr-dongle for more information.

3 Evaluation Board

After power on, please press the reset button one time.

4 Etc

For basic information about Sigfox, please refer to this link, https://www.sigfox.com/.

Use cases

The purpose of this section is to help you understand how to use the basic function of the SRM100A with example. (With this understanding, please have a discussion with Seongji about CLI procedure and parameters at the beginning of target product development.)

Each case shows an example and each example can be done by '*Monarch CMD Test Application*' or by '*CLI commands through terminal program*'.

For the CLI commands in the following sections, please refer to the clause, 'CLI command set' .

1 Case : Sigfox monarch feature is not required. (i.e. Only 1 RC is used.)

There are two types of uplink messages

- Message with downlink requests
- Message without downlink requests

cf. Uplink message : SRM100A -> Sigfox network
Downlink message : Sigfox network -> SRM100A

Downlink message can be triggered only by uplink message. That is, after sending uplink message including downlink request, downlink message can be received.

1.1 Uplink only

1.1.1 RC 1/6/7

Tx procedure in RC 1/6/7 is as follows

1.1.1.1 Open with zone

1.1.1.2 Send frame

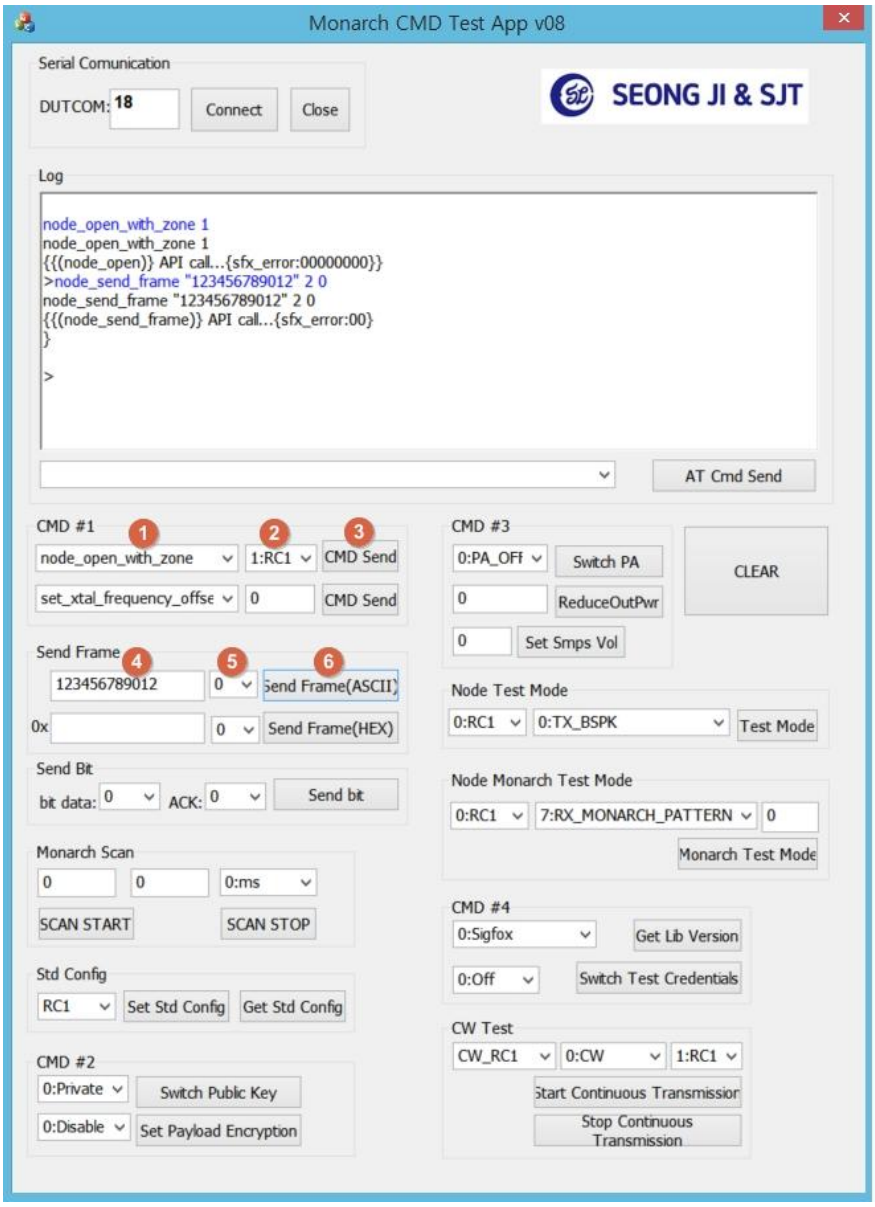
1.1.1.3 Close zone

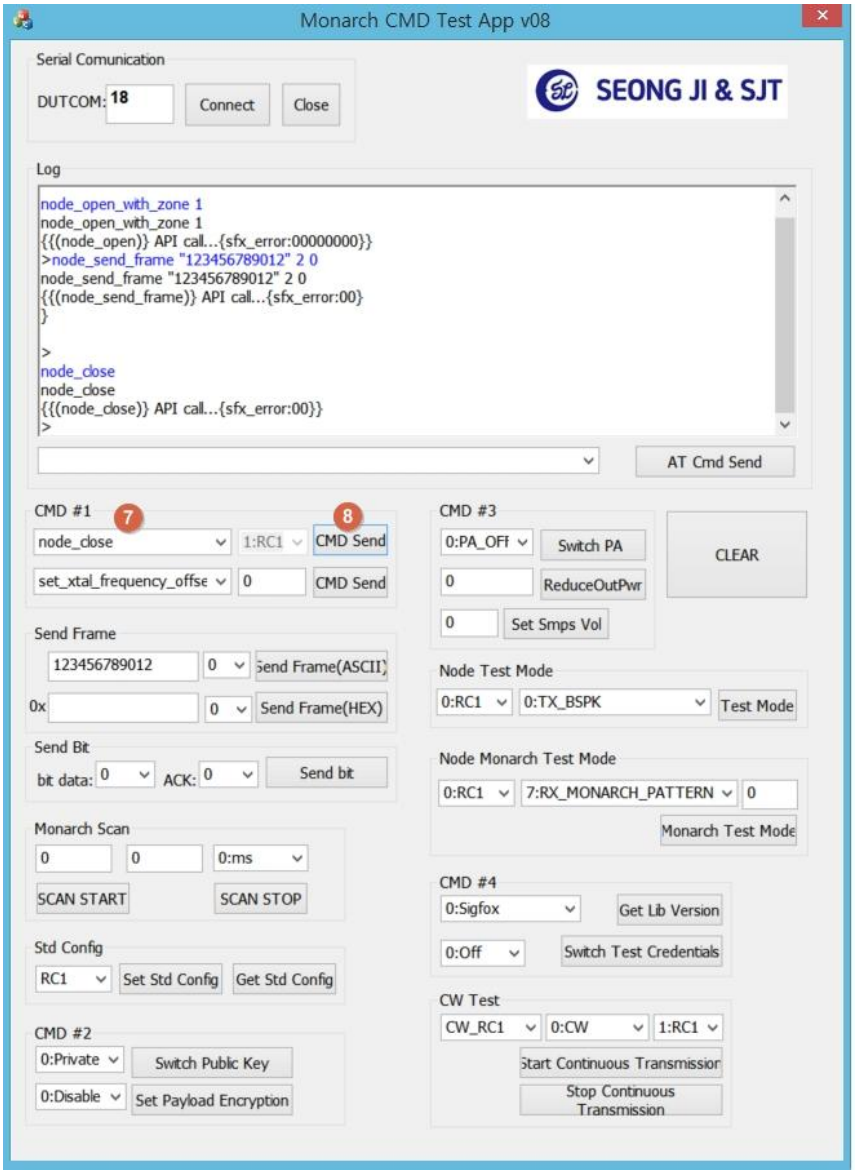
1.1.1.4 Ex. Transmit 12 bytes(ASCII) (RC1)

Each example is presented in two ways. One uses 'Monarch CMD Test Application', the other uses 'CLI commands through terminal program'.

A. Monarch CMD Test Application

: You can do this test, according to numbering in the figure below.





B. CLI commands through terminal program

(Blue text : Input, Purple text : Response)

```
node_open_with_zone 1<CR><LF>
node_open_with_zone 1<CR><LF>
{{{node_open)} API call...{sfx_error:00000000}}<CR><LF>
>
node_send_frame "123456789012" 2 0<CR><LF>
node_send_frame "123456789012" 2 0<CR><LF>
{{{node_send_frame)} API call...{sfx_error:00}<CR><LF>
}<LF><CR>>
node_close<CR><LF>
node_close<CR><LF>
{{{node_close)} API call...{sfx_error:00}}<CR><LF>
>
```

1.1.2 RC 2/4

Tx procedure in RC 2/4 is as follows.

1.1.2.1 Open with zone

1.1.2.2 Set standard configuration

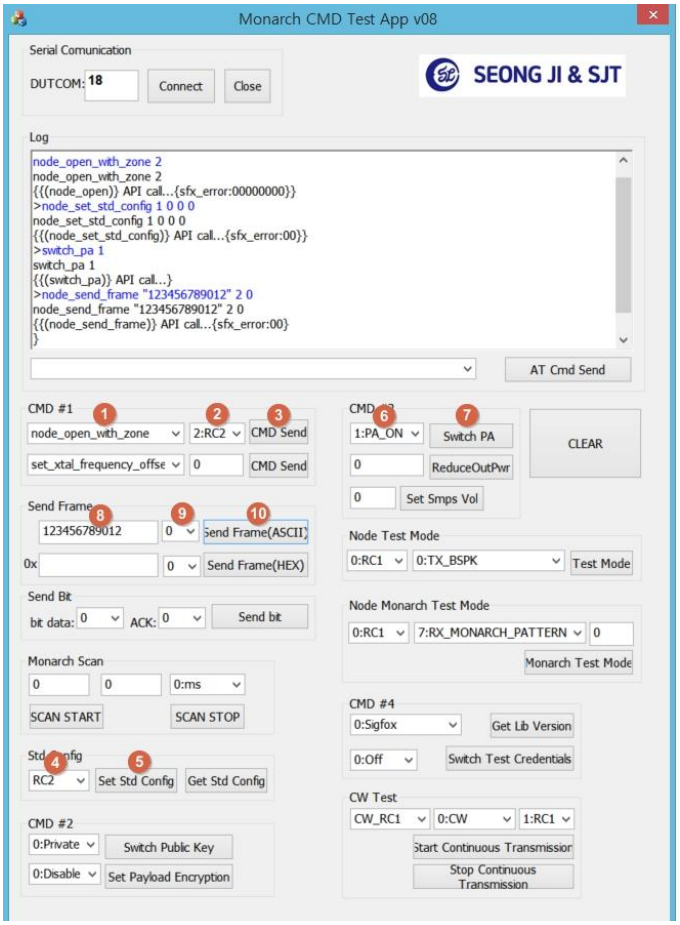
1.1.2.3 Set amplifier

1.1.2.4 Send frame

1.1.2.5 Close zone

1.1.2.6 Ex. Transmit 12 bytes(ASCII) (RC2)

A. Monarch CMD Test Application (Regarding 'Close zone', refer to 1.1.1.4)



B. CLI commands through terminal program

```
node_open_with_zone 2<CR><LF>
node_open_with_zone 2<CR><LF>
{{(node_open)} API call...{sfx_error:00000000}}<CR><LF>
>
node_set_std_config 1 0 0 0<CR><LF>
node_set_std_config 1 0 0 0<CR><LF>
{{(node_set_std_config)} API call...{sfx_error:00}}<CR><LF>
>
switch_pa 1<CR><LF>
switch_pa 1<CR><LF>
{{(switch_pa)} API call...}<CR><LF>
>
node_send_frame "123456789012" 2 0<CR><LF>
node_send_frame "123456789012" 2 0<CR><LF>
{{(node_send_frame)} API call...{sfx_error:00}<CR><LF>
}<LF><CR>>
node_close<CR><LF>
node_close<CR><LF>
{{(node_close)} API call...{sfx_error:00}}<CR><LF>
>
```

* RC4 case : "node_set_std_config 0 0x40000000 0 0<CR><LF>" at step 2).

1.1.3 RC 3/5

Tx procedure in RC 3/5 is as follows.

1.1.3.1 Open with zone

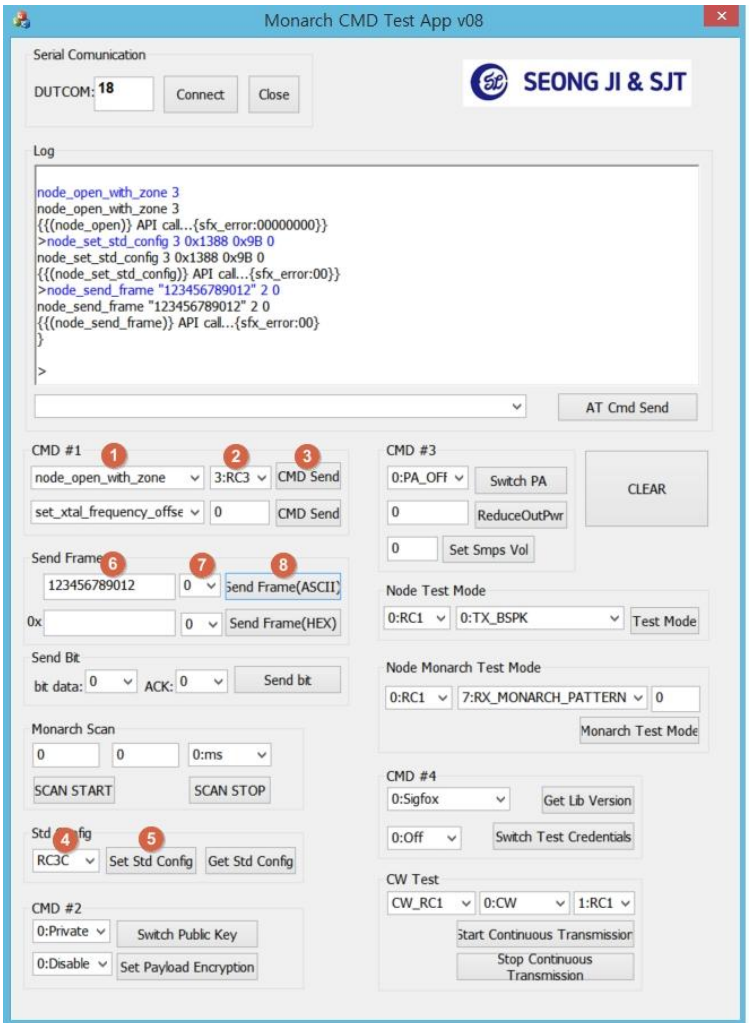
1.1.3.2 Set standard configuration

1.1.3.3 Send frame

1.1.3.4 Close zone

1.1.3.5 Ex.: Transmit 12 byte(ASCII) (RC3)

A. Monarch CMD Test Application (Regarding 'Close zone', refer to 1.1.1.4)



B. CLI commands through terminal program

```
node_open_with_zone 3<CR><LF>
node_open_with_zone 3<CR><LF>
{{(node_open)} API call...{sfx_error:00000000}}<CR><LF>
>
node_set_std_config 3 0x1388 0x9B 0<CR><LF>
node_set_std_config 3 0x1388 0x9B 0<CR><LF>
{{(node_set_std_config)} API call...{sfx_error:00}}<CR><LF>
>
node_send_frame "123456789012" 2 0<CR><LF>
node_send_frame "123456789012" 2 0<CR><LF>
{{(node_send_frame)} API call...{sfx_error:00}<CR><LF>
}<LF><CR>>
node_close<CR><LF>
node_close<CR><LF>
{{(node_close)} API call...{sfx_error:00}}<CR><LF>
>
```

1.2 Uplink and Downlink

1.2.1 RC 1/6/7

Tx & Rx procedure in RC 1/6/7 is as follows.

1.2.1.1 Open with zone

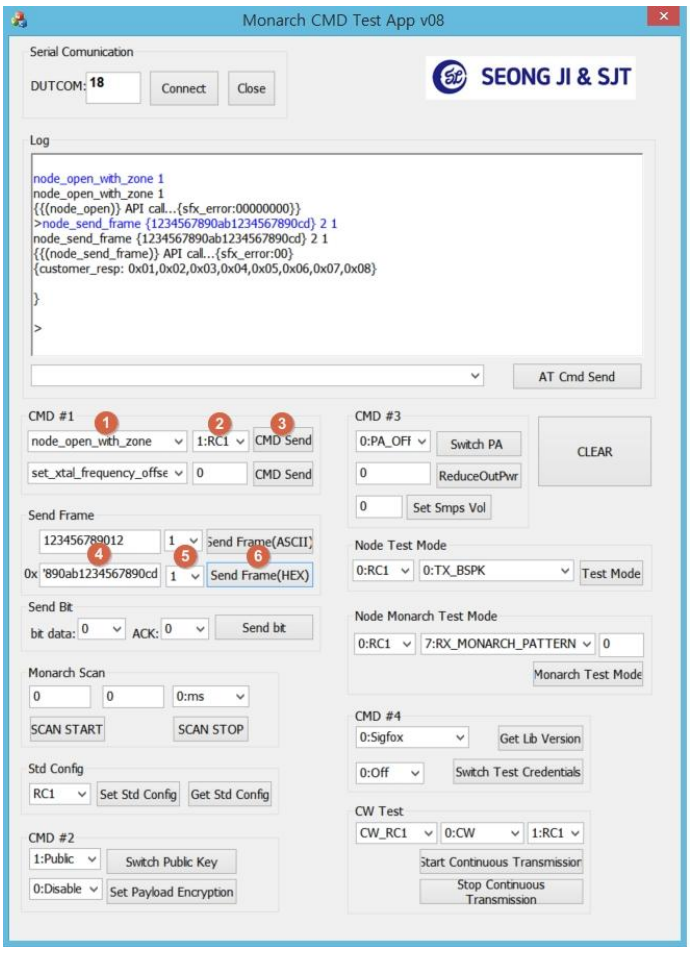
1.2.1.2 Send frame

1.2.1.3 Receive data

1.2.1.4 Close zone

1.2.1.5 Ex. Transmit 12 bytes(Hexa) and receive message (RC1)

A. Monarch CMD Test Application (Regarding 'Close zone', refer to 1.1.1.4)



B. CLI commands through terminal program

```
node_open_with_zone 1<CR><LF>
node_open_with_zone 1<CR><LF>
{{(node_open)} API call...{sfx_error:00000000}}<CR><LF>
>
node_send_frame {1234567890ab1234567890cd} 2 1<CR><LF>
node_send_frame {1234567890ab1234567890cd} 2 1<CR><LF>
{{(node_send_frame)} API call...{sfx_error:00}<CR><LF>
{customer_resp: 0x0A,0x00,0x0F,0x06,0xE0,0x16,0x01,0x01}<LF><CR>}<LF><CR>>
node_close<CR><LF>
node_close<CR><LF>
{{(node_close)} API call...{sfx_error:00}}<CR><LF>
>
```

* Rx data

{customer_resp : 0x0A,0x00,0x0F,0x06,0xE0,0x16,0x01,0x01}<LF><CR>}<LF><CR>>

* If you use SDR dongle, you must execute "*switch_public_key 1* " before step 1).

1.2.2 RC 2/4

Tx & Rx procedure in RC 2/4 is as follows.

1.2.2.1 Open with zone

1.2.2.2 Set standard configuration

1.2.2.3 Set amplifier

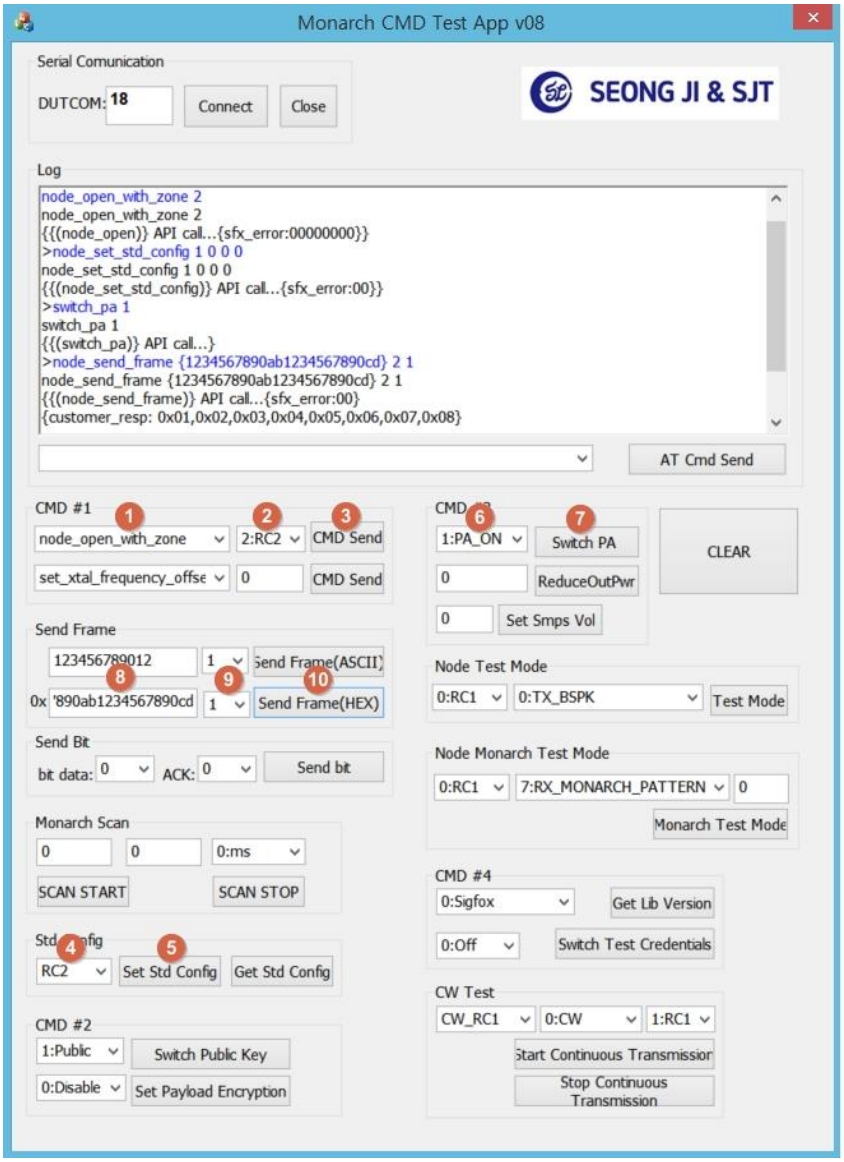
1.2.2.4 Send frame

1.2.2.5 Receive data

1.2.2.6 Close zone

1.2.2.7 Ex.: Transmit 12 bytes(Hexa) and receive message (RC2)

A. Monarch CMD Test Application (Regarding 'Close zone', refer to 1.1.1.4)



B. CLI commands through terminal program

```
node_open_with_zone 2<CR><LF>
node_open_with_zone 2<CR><LF>
{{(node_open)} API call...{sfx_error:00000000}}<CR><LF>
>
node_set_std_config 1 0 0 0<CR><LF>
node_set_std_config 1 0 0 0<CR><LF>
{{(node_set_std_config)} API call...{sfx_error:00}}<CR><LF>
>
switch_pa 1<CR><LF>
switch_pa 1<CR><LF>
{{(switch_pa)} API call...}<CR><LF>
>
node_send_frame {1234567890ab1234567890cd} 2 1<CR><LF>
node_send_frame {1234567890ab1234567890cd} 2 1<CR><LF>
{{(node_send_frame)} API call...{sfx_error:00}<CR><LF>
{customer_resp: 0x0A,0x00,0x0F,0x06,0xE0,0x16,0x01,0x01}<LF><CR>}<LF><CR>>
node_close<CR><LF>
node_close<CR><LF>
{{(node_close)} API call...{sfx_error:00}}<CR><LF>
>
```

* Rx data

{customer_resp : 0x0A,0x00,0x0F,0x06,0xE0,0x16,0x01,0x01}<LF><CR>}<LF><CR>>

* If you use SDR dongle, you must execute "*switch_public_key 1*" before step 1)

* RC4 case : "node_set_std_config 0 0x40000000 0 0<CR><LF>" at step 2).

1.2.3 RC 3/5

Tx & Rx procedure in RC 3/5 is as follows.

1.2.3.1 Open with zone

1.2.3.2 Set standard configuration

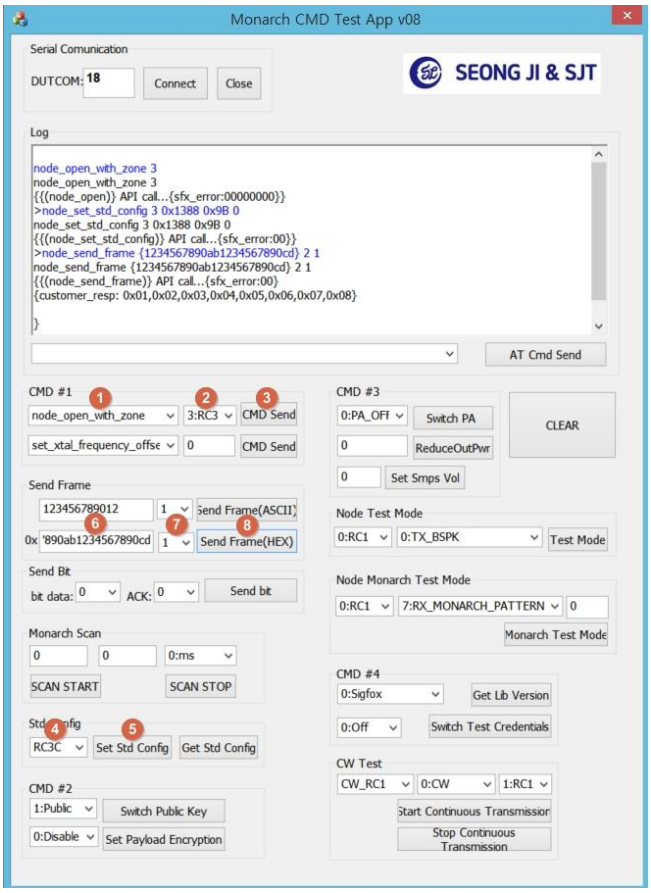
1.2.3.3 Send frame

1.2.3.4 Receive data

1.2.3.5 Close zone

1.2.3.6 Ex.: Transmit 12 bytes(Hexa) and receive message (RC3)

A. By using Monarch CMD Test App



B. By using CLI commands through terminal program

```
node_open_with_zone 3<CR><LF>
node_open_with_zone 3<CR><LF>
{{{node_open}} API call...{sfx_error:00000000}}<CR><LF>
>
node_set_std_config 3 0x1388 0x9B 0<CR><LF>
node_set_std_config 3 0x1388 0x9B 0<CR><LF>
{{{node_set_std_config}} API call...{sfx_error:00}}<CR><LF>
>
node_send_frame {1234567890ab1234567890cd} 2 1<CR><LF>
node_send_frame {1234567890ab1234567890cd} 2 1<CR><LF>
{{{node_send_frame}} API call...{sfx_error:00}<CR><LF>
{customer_resp: 0x0A,0x00,0x0F,0x06,0xE0,0x16,0x01,0x01}<LF><CR>}<LF><CR>>
node_close<CR><LF>
node_close<CR><LF>
{{{node_close}} API call...{sfx_error:00}}<CR><LF>
>
```

* Rx data

{customer_resp : 0x0A,0x00,0x0F,0x06,0xE0,0x16,0x01,0x01}<LF><CR>}<LF><CR>>

* If you use SDR dongle, you must execute '*switch_public_key 1*' before step 1).

2 Case : Sigfox monarch feature is required.
(i.e. Automatic RC change(Roaming) is required.)

When roaming is mandatory, Sigfox monarch feature is necessary. In the other cases, monarch feature is not required.

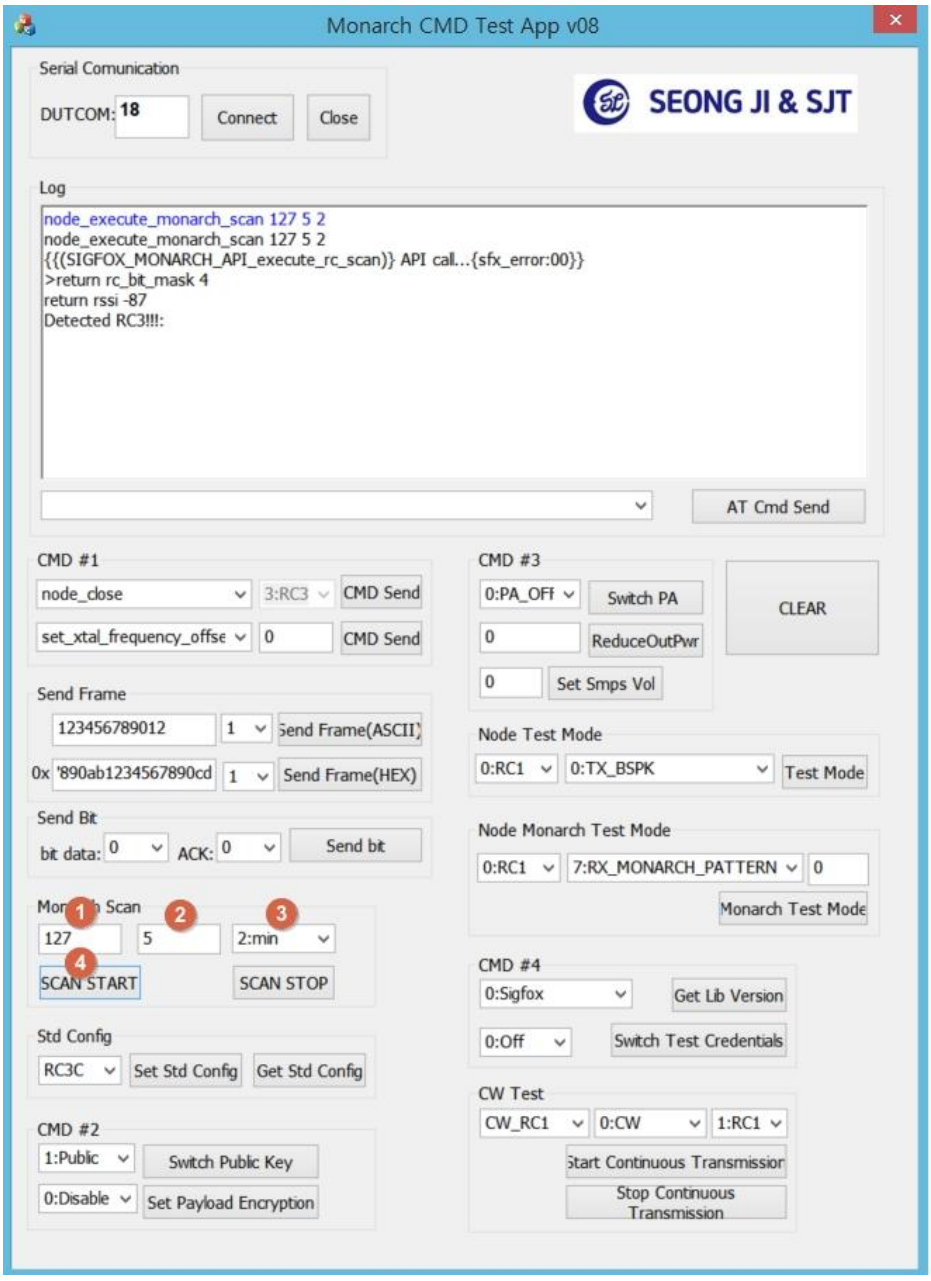
In case of using Sigfox monarch feature, you need to do RC scan first. This is the procedure to detect the RC available in your region. (Please make sure first that the Sigfox base station is broadcasting monarch beacon signal. You may ask this to your local Sigfox SO(Service Operator).)

2.1 RC scan

2.1.1 Start RC scan (from RC1 to RC7)

2.1.1.1 Ex. RC scan (RC1 is detected.)

A. By using Monarch CMD Test App



B. By using CLI commands through terminal program

```
node_execute_monarch_scan 127 5 2<CR><LF>
node_execute_monarch_scan 127 5 2<CR><LF>
{{(SIGFOX_MONARCH_API_execute_rc_scan)} API call...{sfx_error:00}}<CR><LF>
>return rc_bit_mask 1<CR><LF>
return rssi -60<CR><LF>
Detected RC1!!!:<CR><LF>
```

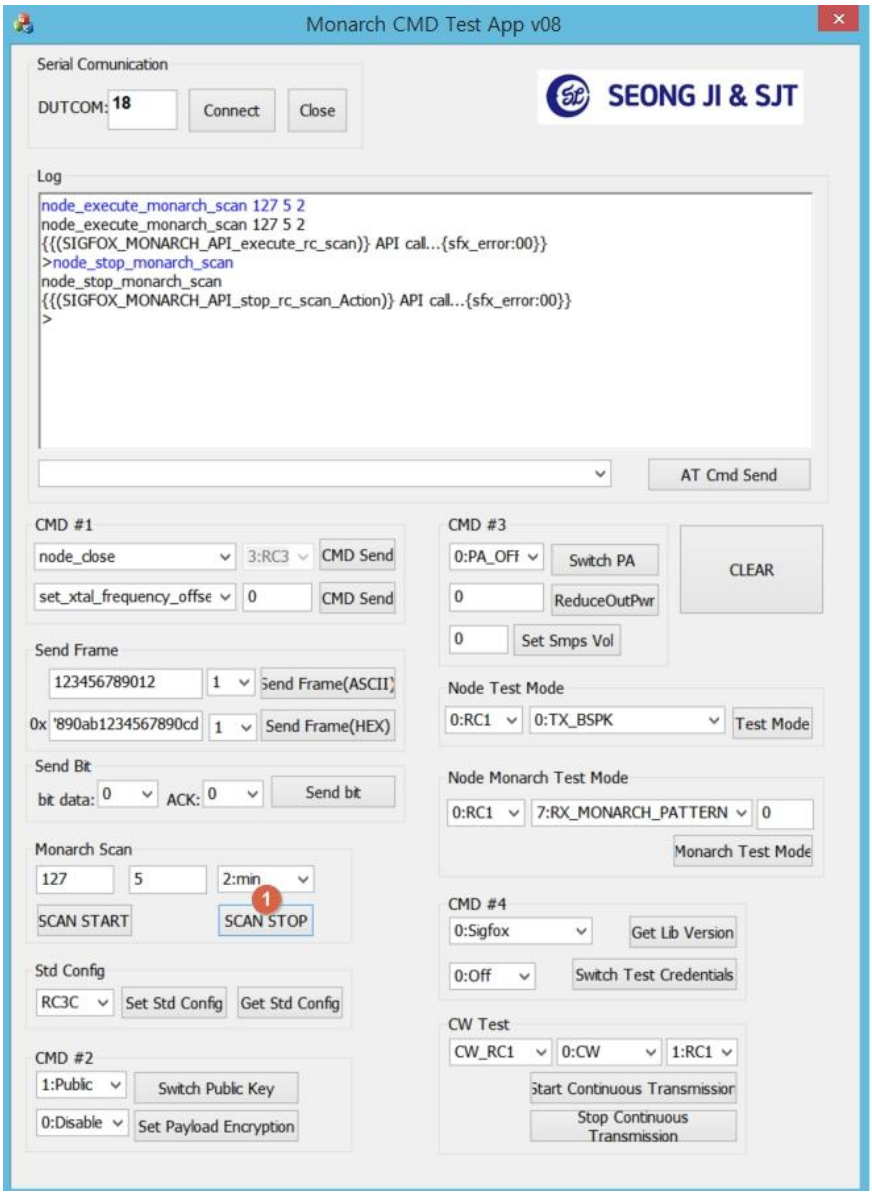
If RC scan is successful (any RC is detected), you can go to 2.2 or 2.3.

2.1.2 Stop RC scan

Basically, RC scan is completed when any RC is detected or RC scan timer is expired. This command is used when you want to stop RC scan, during Monarch scan is in progress.

2.1.2.1 Ex. Stop RC scan

A. By using Monarch CMD Test App



B. By using CLI commands through terminal program

```
node_stop_monarch_scan<CR><LF>
node_stop_monarch_scan<CR><LF>
{{{SIGFOX_MONARCH_API_stop_rc_scan_Action)} API call...{sfx_error:00}}<CR><LF>
>
```

2.2 Uplink only

2.2.1 RC 1/6/7

Same as 1.1.1.

2.2.2 RC 2/4

Same as 0.

2.2.3 RC 3/5

Same as B

2.3 Uplink and Downlink

2.3.1 RC 1/6/7

Same as 1.2.1.

2.3.2 RC 2/4

Same as 1.2.2.

2.3.3 RC 3/5

Same as 0.

3 Sigfox error cases

3.1 node_open error (60)

```
node_close<CR><LF>
node_close<CR><LF>
{{(node_close)} API call...{sfx_error:00}}<CR><LF>
>
node_send_frame "123456789012" 2 0<CR><LF>
node_send_frame "123456789012" 2 0<CR><LF>
{{(node_send_frame)} API call...{sfx_error:60}<CR><LF>
}<LF><CR>>
```

3.2 node_set_std_config error (60)

```
node_open_with_zone 3<CR><LF>
node_open_with_zone 3<CR><LF>
{{(node_open)} API call...{sfx_error:00000000}}<CR><LF>
>
node_send_frame "123456789012" 2 0<CR><LF>
node_send_frame "123456789012" 2 0<CR><LF>
{{(node_send_frame)} API call...{sfx_error:60}<CR><LF>
}<LF><CR>>
```

3.3 RC scan error (B5)

```
node_execute_monarch_scan 127 5 2
node_execute_monarch_scan 127 5 2
<CR><LF>
<CR><LF>
{{(SIGFOX_MONARCH_API_execute_rc_scan)} API call...{sfx_error:00}}<CR><LF>
>return rc_bit_mask 4<CR><LF>
return rssi -86<CR><LF>
Detected RC3!!!:<CR><LF>
node_stop_monarch_scan<CR><LF>
node_stop_monarch_scan<CR><LF>
{{(SIGFOX_MONARCH_API_stop_rc_scan_Action)} API call...{sfx_error:B5}}<CR><LF>
>
```

3.4 No RC is detected.

```
node_execute_monarch_scan 127 5 2<CR><LF>
node_execute_monarch_scan 127 5 2<CR><LF>
{{(SIGFOX_MONARCH_API_execute_rc_scan)} API call...{sfx_error:00}}<CR><LF>
>return rc_bit_mask 0<CR><LF>
return rssi 0<CR><LF>
```

3.5 The message could not be transmitted because the channel was busy. (7E)

This is only for RC3 and RC5.

```
node_open_with_zone 3<CR><LF>
node_open_with_zone 3<CR><LF>
{{(node_open)} API call...{sfx_error:00000000}}<CR><LF>
>
node_set_std_config 3 0x1388 0x9B 0<CR><LF>
node_set_std_config 3 0x1388 0x9B 0<CR><LF>
{{(node_set_std_config)} API call...{sfx_error:00}}<CR><LF>
>
node_send_frame "123456789012" 2 0<CR><LF>
node_send_frame "123456789012" 2 0<CR><LF>
{{(node_send_frame)} API call...{sfx_error:7E}}<CR><LF>
}<LF><CR>>
```

3.6 The message was not received. (9B)

```
node_send_frame {1234567890ab1234567890cd} 2 1<CR><LF>
node_send_frame {1234567890ab1234567890cd} 2 1<CR><LF>
{{(node_send_frame)} API call...{sfx_error:9B}}<CR><LF>
}<LF><CR>>
```

4 BLE

This section explains how to test the BLE function supported by SRM100A. SRM100A supports only peripheral mode.

For test convenience(explanation), 'nRF Connect(Android app.)' is used as central device at sub-clause 4.1, and 'Serial Bluetooth Terminal(Android app.)' is used as central device at sub-clause 4.2.

4.1 Advertising (Not connectable)

There are two types of advertising mode. One is non-connectable(beacon) and the other is connectable. Advertising can be done via the command, 'ble_start'.

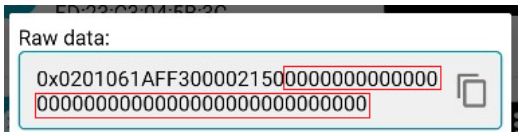
For testing, the Android application, 'nRF Connect' is used as a central device.

4.1.1 Default advertising

- Command : ble_start [adv_type] [Advertising_Interval_Max] [Advertising_Interval_Min]

  Ex. ble_start 3 1600 1600

  -> This advertising message can be seen at the central(nRF Connect) as below.



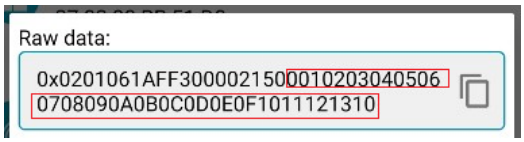
4.1.2 Modifying advertising payload

- Command : ble_set_beacon_data [payload] //Max 21 bytes

  Ex. ble_set_beacon_data {000102030405060708090a0b0c0d0e0f1011121310}

  ble_start 3 1600 1600

  -> This advertising message can be seen at the central(nRF Connect) as below.



4.2 Advertising & Connection

4.2.1 UUID Information

The default UUID is as below.

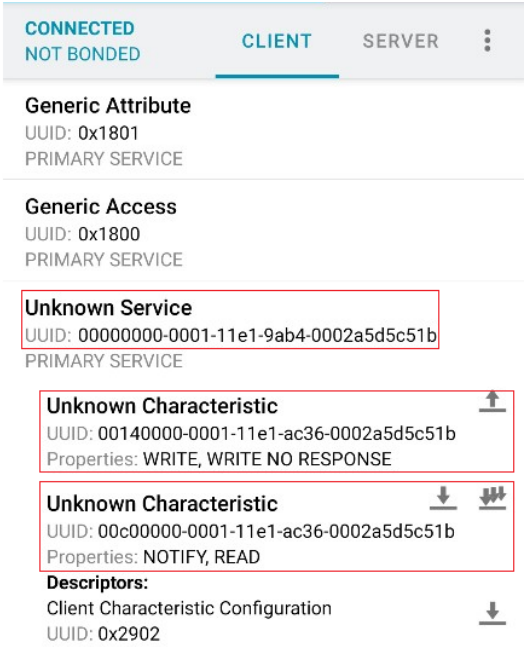
Service UUID : 00000000-0001-11e1-9ab4-0002a5d5c51b

Read characteristic UUID : 00c00000-0001-11e1-ac36-0002a5d5c51b

Write characteristic UUID : 00140000-0001-11e1-ac36-0002a5d5c51b

Ex. ble_start 0 1600 1600

-> The information below will be seen at the central(nRF Connect)



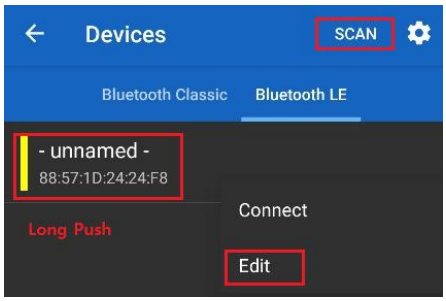
4.2.2 Data Send / Receive

After connection, data send/receive is possible..

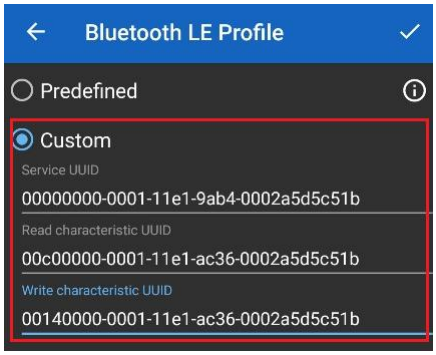
For testing, the Android application, 'Serial Bluetooth Terminal" is used as a central device.

■ Connecting with the central ('Serial Bluetooth Terminal')

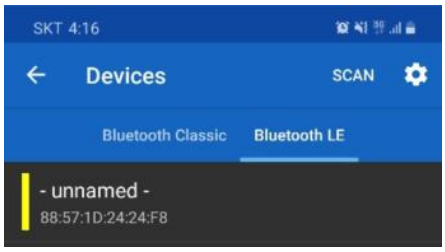
- Scan -> Select device -> Edit



- Check Custom



- Select the device, and then connected.



- Send command : ble_send_noti_character [data] //Max 20bytes

  Ex. ble_send_noti_Character {31323334353637383930313233343536373839 30}

  -> The data(green text) below will be seen at the central ('Serial Bluetooth Terminal')

Then, input the string, '12345678901234567890' like the bottom and push button. The message will be sent to the peripheral(SRM100A) and the message below will be seen at the peripheral(SRM100A), via serial port.

modified_event : 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39 0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39 0x30

Notice

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications.

SEONG JI INDUSTRIAL MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE.

SEONG JI INDUSTRIAL disclaims all liability arising from this information and its use. Use of SEONG JI INDUSTRIAL devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless SEONG JI INDUSTRIAL from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any SEONG JI INDUSTRIAL intellectual property rights unless otherwise stated.