LSM1x0A Sigfox CLI Command Interface Manual

Rev 1.1

SJIT

JAN. 19. 2024

Contents

History

Date	Contents	Version	
2022-01-28	Create	V1.0	
2024-01-19	Add content of Default RC Change the company name	V1.1	

1. AT command complete 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 : 9600 bauds
- Data bits: 8
- Stop bits: 1
- Parity: None

The following table gather all AT command available:

2. Sigfox RF Test Description

2.1 RF standard test

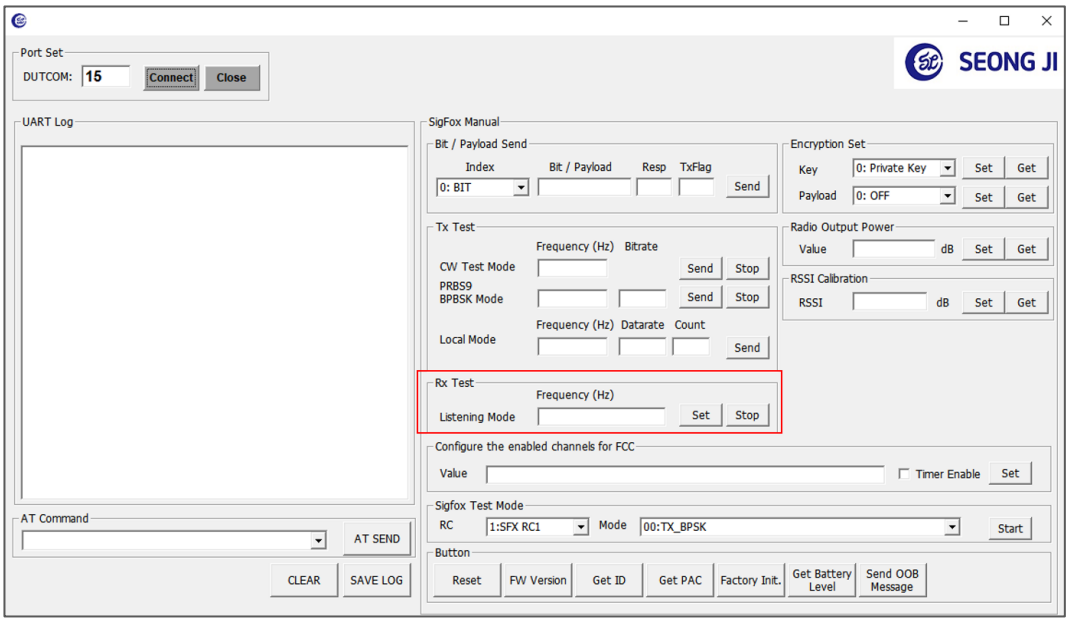
1) Input AT Command command to LSM1x0A used as RX

EX) AT+RL=869525000

Test Result

➔ if received success display "TEST PASSED"

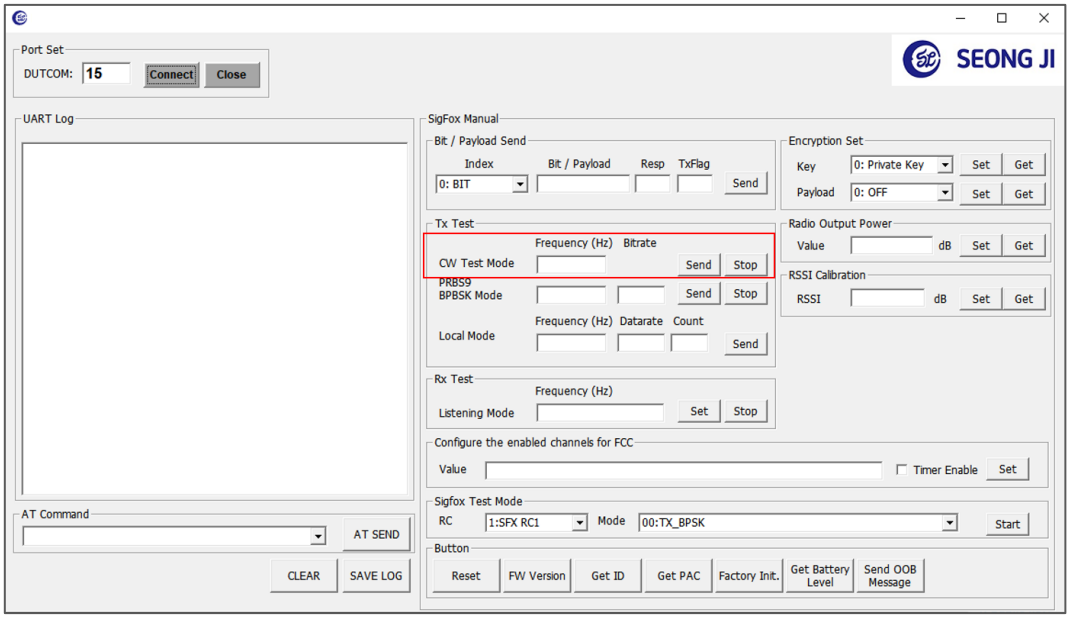
➔ if received fail display "Wait For End of Rx"



2) Input AT Command command to LSM1x0A used as TX

EX) AT+CW=868130000

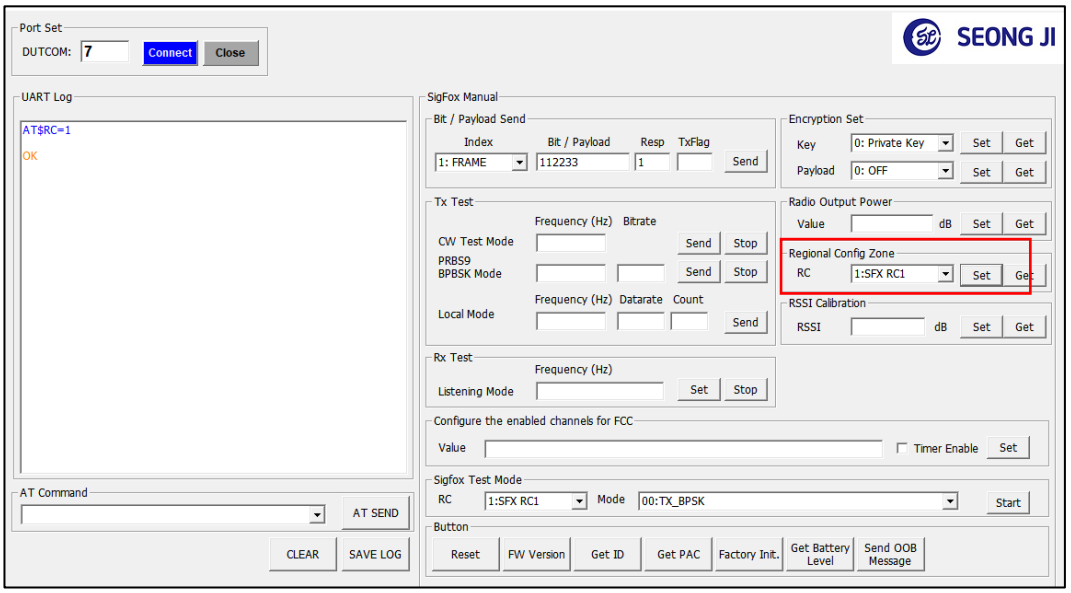
➔ Transmit frequency to Continuous wave



2.2 Backend test

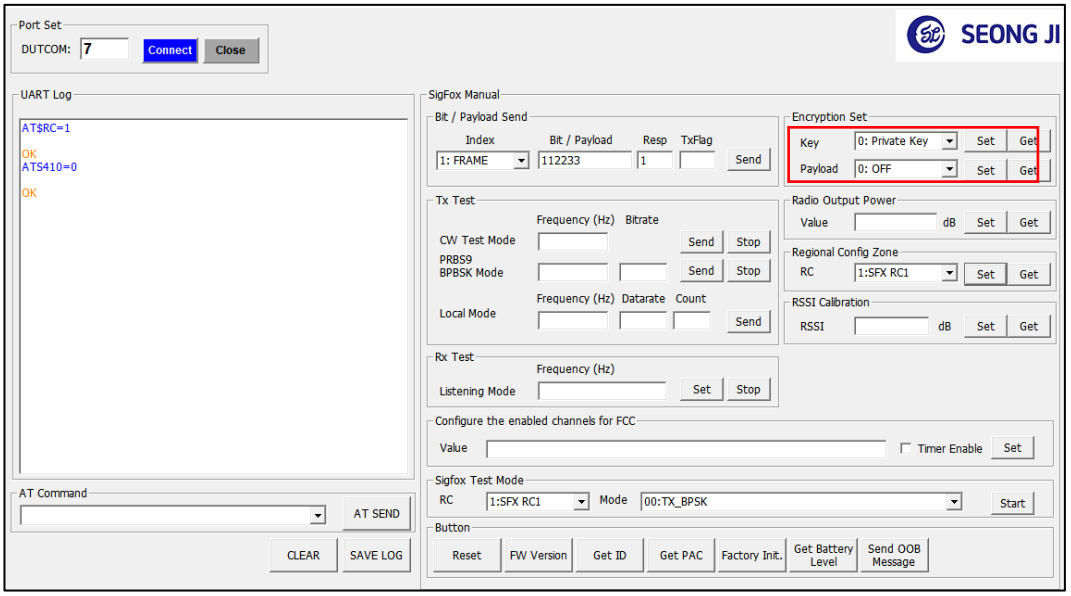
1) Select regional config zone

EX) AT$RC=1



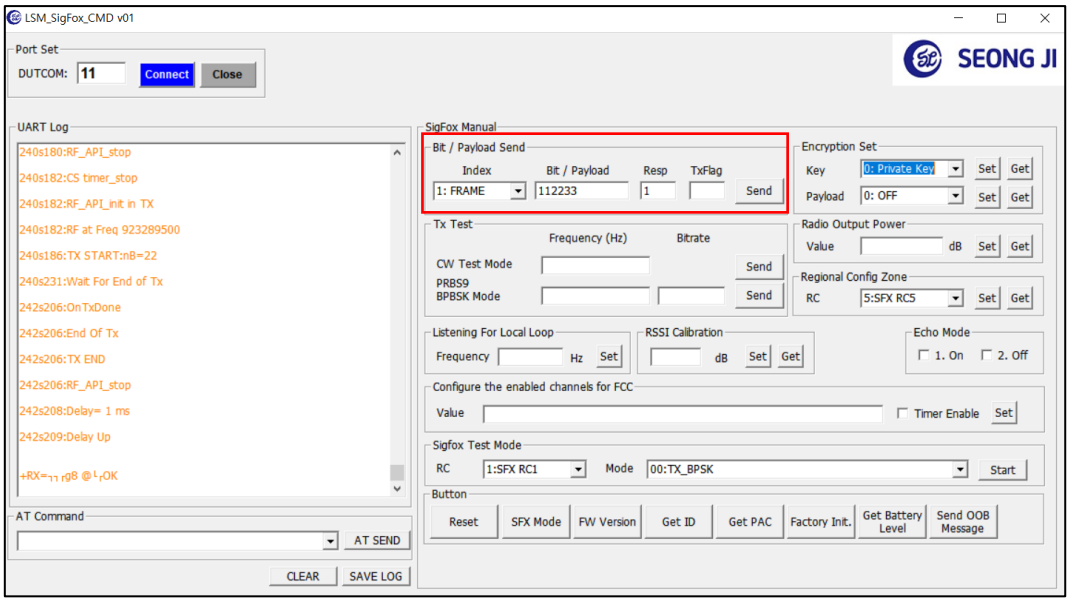
2) Key setting

EX) ATS410=0



2) Send dataa

EX) AT$SF=112233,1,1



3. Sigfox Command

Command	Name	Description
AT?	Help on all <CMD>	Help on All Commands Ex) AT? (CR)
ATZ	Reset	Trig a MCU reset. Ex) ATZ (CR)
ATE=mode	Echo mode	Not used except to set echo mode. <mode>: [0: echo ON, 1: echo OFF] Ex) ATE=1 (CR) ATE=? (CR) Get echo mode
AT+BAT=?	Battery level	Get the battery level (in mV). Ex) AT+BAT=? (CR)
AT+VL=level AT+VL=?	Verbose level	Set or Get the verbose level. <level>: [0: off, 1: Low, 2: Meddle, 3: High] Ex) AT+VL=3 (CR) AT+VL=? (CR) Get level
AT+MODE=mode AT+MODE=?	Mode Change	LoRa & Sigfox Mode Change. After a MCU reset. <mode>: [0: SigFox, 1: LoRa] Ex) AT+MODE=1 (CR) AT+MODE=? (CR) Get mode
AT$SSWVER=?	Software version	Get the Software version. Ex) AT$SSWVER=? (CR)
AT+VER=?	Firmware and library versions	Get the version of firmware and libraries. Ex) AT+VER=? (CR)
AT$RFS	Factory settings	Restores the factory setting. Ex) AT$RFS (CR)
AT$ID	Device ID	Get the 32-bit device ID. Ex) AT$ID (CR)

Command	Name	Description
AT$PAC	Device PAC	Get the 8-byte device PAC. Ex) AT$PAC (CR)
AT$SB=bit_value{,opt_responsewaited}{,opt_txflag}	Bit status	Send a bit to the Sigfox network. <bit_value>: [0 or 1] <opt_responsewaited> 0: no response waited (default) <opt_responsewaited> 1: response waited <opt_txflag> 0: one Tx frame sent <opt_txflag> 1: three Tx frame sent (default) Ex) AT$SB=0,1,1 (CR) AT$SB=1 (CR) sends bit 1 with no response waited. AT$SB=0,1 (CR) sends bit 0 with a response waited. AT$SB=0,1,1 (CR) sends bit 0 with a response waited and with three Tx frames sent.
AT$SF=payload{,opt_responsewaited}{,opt_txflag}	ASCII payload in bytes	Send a frame to the Sigfox network. <payload>: [12 bytes maximum in ASCII format (24 ASCII characters max)] <opt_responsewaited>: [0: no response waited (default)] <opt_responsewaited>: [1: response waited] <opt_txflag>: [0: one Tx frame sent] <opt_txflag>: [1: three Tx frames sent (default)] Ex) AT$SF=313245,1,1 (CR) AT$SF=313245 (CR) sends 0x31 0x32 0x45 payload with no response waited. AT$SF=313245,1 (CR) sends 0x31 0x32 0x45 payload with a response waited. AT$SF=313245,1,1 (CR) sends 0x31 0x32 0x45 payload with a response waited and with three Tx frames sent.

Command	Name	Description
AT$SH=payload_length, payload{,opt_responsewaited}{,opt_txflag}	Hexadecimal payload in bytes	Send a Hex frame to the Sigfox network. <payload_length>: [length in bytes] <payload>: [12 bytes maximum in hexadecimal format] <opt_responsewaited>: [0: no response waited (default)] <opt_responsewaited>: [1: response waited] <opt_txflag>: [0: one Tx frame sent] <opt_txflag>: [1: three Tx frames sent (default)] Ex) AT$SH=1,A,1 (CR) AT$SH=1,A (CR) sends 0x41 payload with no response waited. AT$SH=1,A,1 (CR) sends 0x41 payload with a response waited.
AT$CW=freq	Continuous wave(CW)	Start or stop a continuous unmodulated carrier for test. Run CW Test mode. <freq>: frequency (in Hz) Ex) AT$CW=868130000 (CR) AT$CW=0 (CR) Stop a CW
AT$PN=freq,bitrate	PRBS9 BPBSK test mode	Run PRBS9 BPBSK Test mode. Send a continuous modulated carrier for test. <freq>: frequency (in Hz) <bitrate>: 100 or 600 Ex) AT$PN=868130000,100 (CR) AT$PN=0 (CR) Stop a BPBSK
AT$RC=rc	Sigfox RC setting	Commands for changing and setting RC. <rc> RC2(default band) = 2 RC4 = 4 Ex) AT$RC=2

Command	Name	Description
AT$TM=rc,mode	Sigfox test mode	Start a Sigfox test mode. <rc> SFX_RC1 = 1 SFX_RC2 = 2 SFX_RC3C = 3C SFX_RC4 = 4 SFX_RC5 = 5 SFX_RC6 = 6 SFX_RC7 = 7 <mode> SFX_TEST_MODE_TX_BPSK = 0 SFX_TEST_MODE_TX_PROTOCOL = 1 SFX_TEST_MODE_RX_PROTOCOL = 2 SFX_TEST_MODE_RX_GFSK = 3 SFX_TEST_MODE_RX_SENSI = 4 SFX_TEST_MODE_TX_SYNTH = 5 SFX_TEST_MODE_TX_FREQ_DISTRIBUTION = 6 SFX_TEST_MODE_TX_BIT = 11 SFX_TEST_MODE_PUBLIC_KEY = 12 SFX_TEST_MODE_NVM = 13 Ex) AT$TM=1,0 (CR)
AT$RSSICAL=value AT$RSSICAL=?	RSSI value in dB	Set or Get the RSSI calibration value in dB. <value>: calibration value (in dB) Ex) AT$RSSICAL=0 (CR) AT$RSSICAL=? (CR)
AT$RL=freq	Listening for a data packet	Starts listening for a local loop. <freq>: frequency (in Hz) Stop by input 'X' Ex) AT$RL=869525000 (CR)
AT$SL=freq,datarate,count	Send local loop	Send TX packet up to count number for local test. <freq>: frequency (in Hz) <datarate>: data rate (in bps) <count>: send packets counter Ex) AT$SL=869525000,600,10 (CR)
ATS300	Out-of-band message	Send one keep-alive out-of-band message. Ex) ATS300 (CR)

Command	Name	Description
ATS302=power ATS302=?	Radio output power	Set or Get the radio output power. <power> : power (in dBm) Ex) ATS302=15 (CR) ATS302=? (CR) Get power
ATS400=<8_digit_word0><8_digit_word1><8_digit_word2>,timer_enable	Enabled channels for FCC	Configure the enabled channels for FCC. Ex) ATS400=00000000400000000000000,0 (CR)
ATS410=key ATS410=?	Encryption key	Set or Get the configuration of the device encryption key. <key>: [0: Use Private key, 1: Use Public key] Ex) ATS410=1 (CR) ATS410=? (CR) Get the encryption key
ATS411=mode ATS411=?	Payload encryption	Set or Get the device payload encryption mode. <mode>: [0:Payload Encryption OFF, 1:Payload Encryption ON} Ex) ATS411=1 (CR) ATS411=? (CR) Get payload encryption