Product Specifications

LoRa Wireless Communication Module

LM-130 series

VER: 1.1

GlobalSat WorldCom Corporation
16F., No. 186, Jian 1st Rd, Zhonghe Dist.,
New Taipei City 23553, Taiwan
Tel: 886.2.8226.3799/ Fax: 886.2.8226.3899
service@globalsat.com.tw
www.globalsat.com.tw

USGlobalSat Incorporated
14740 Yorba Court Chino, CA 91710
Tel: 888.323.8720 / Fax: 909.597.8532
sales@usglobalsat.com
www.usglobalsat.com

Revision History

Rev. No.	Change History	Issue Date	Remark
1.0	Initiation	2016.3.14	Preliminary
1.1	Modify UART INTERFACE description & add AAT1 Restore of AT command	2016.4.6	

Product Description

The Globalsat LM-130 is a LoRaWAN^{TM} compliant RF module, using Semtech SX1276 transceiver features the LoRa® long range modem, which provides long-range, low data rate IoT connectivity to sensors, electronic meter reading, geolocation devices, industrial monitoring and control, home and building automation, long range irrigation systems, and all kinds of M2M equipments. It works as the end-node devices in the LoRaWAN^{TM} infrastructure.

Product feature

- LoRaWAN^{TM} compliant
- Ultra-high sensitive receiving ability by LoRa spread spectrum modulation technology
- Long-distance transmission (1KM to 10KM)
- Multi-channel, dual data buffer (each 256 Bytes)
- Instant wake up over the air
- Built-in watchdog
- LoRa/FSK/GFSK/OOK modulation, 2-way half –duplex communication, strong anti-interfere
- Maximal output power100mW(20dBm), output power adjustable between 5-20dBm
- Easily use, auto exchange on communication & transceiver
- Tuning free
- Accord FCC,ETSI standard

Hardware Specifications

Chipset	SEMTECH LoRa SX1276
Antenna	IPex RF Connector
Frequency	862-870MHz (Model: LM-130E) 902-928MHz (Model: LM-130H)
Transmission Power	100mW (max.)
Transmission Media	UART
UART	Baud Rate : 57600bps Parity: 8N1
Operation Voltage	3.0V~6V
Current Consumption	Receiving: 21 mA (typical) Transmitting: 125 mA (typical) Sleeping: 5 uA (typical)
Transmission Distance	1KM~10KM (0.81Kbps)
Receiving Sensitivity	-132dBm@0.81Kbps
Operation Temperature	-40°C~ 85°C
Humidity	5%~95% (Non-condensing)
Dimension	25mm x 18mm (PCBA)

Pin Definition

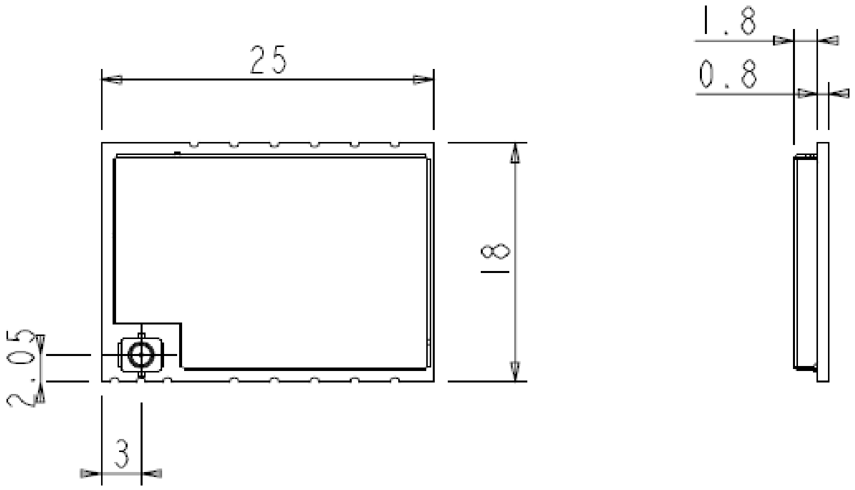
No.	Pin	Definition	Description
1	GND	GND	Ground
2	RF_IO	Input/Output	RF input / output
3	GND	GND	Ground
4	PIO_9	Reserved	Reserved for extension (I2C_SDA)
5	PIO_10	Reserved	Reserved for extension (I2C_SCL)
6	PIO_2	Reserved	Reserved for extension (UART2_TX)
7	PIO_3	Reserved	Reserved for extension (UART2_RX)
8	PIO_11	Reserved	Reserved for extension (UART3_TX)
9	PIO_12	Reserved	Reserved for extension (UART3_RX)
10	NRST	Input	RESET，LOW ACTIVE
11	LoRa_EN	Input	MODULE POWER ENABLE，HIGH ACTIVE “HI” = 0.91~6 Vdc ,”Low” = 0 ~ 0.38 Vdc
12	VDD	Input	**3.0-6.0 Vdc**
13	GND	GND	Grand
14	GND	GND	Grand
15	PIO_8	Reserved	Reserved for extension (ADC)
16	PIO_7	Reserved	Reserved for extension (UART3_CTS)
17	PIO_6	Reserved	Reserved for extension (UART3_RTS)
18	PIO_5	Reserved	Reserved for extension (USB_DM)
19	PIO_4	Reserved	Reserved for extension (USB_DP)
20	JTAG_TCK	Input	JTAG Interface
21	JTAG_TMS	Input	JTAG Interface
22	UART1_RX	Input	UART Input port
23	UART1_TX	Output	UART Output port
24	PIO_1	Reserved	Reserved for extension
25	PIO_0	Reserved	Reserved for extension

GPIO level except LoRa_EN :

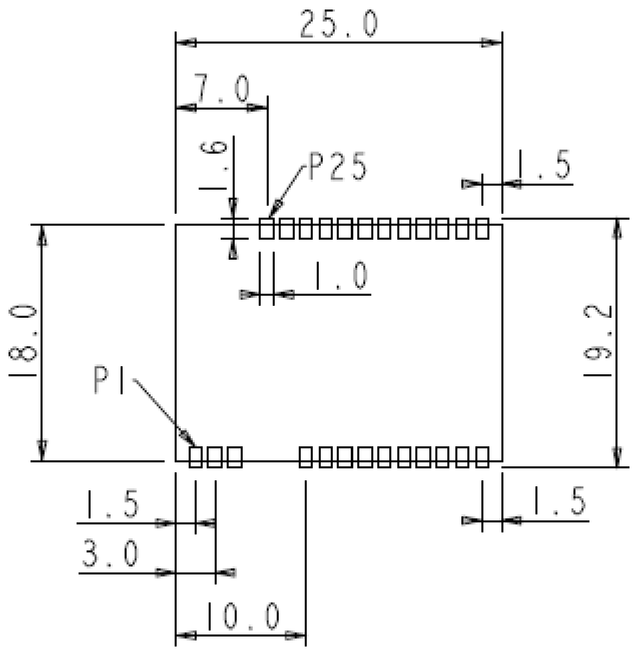
INPUT ”HI” = 1.96~3.1 Vdc, ”Low” = 0~0.84 Vdc

OUTPUT “HI” = 2.1~2.8 Vdc, ”Low” = 0~0.7 Vdc

Product Size



Recommend Layout



Configuration

Activation of an end-device can be achieved in two ways, either via Over-The-Air Activation (OTAA) when an end-device is deployed or reset, or via Activation By Personalization (ABP) in which the two steps of end-device personalization and activation are done as one step.

■ Over-the-Air Activation

For over-the-air activation, end-devices must follow a join procedure prior to participating in data exchanges with the network server. An end-device has to go through a new join procedure every time it has lost the session context information. The join procedure requires the end-device to be personalized with the following information before its starts the join procedure: a globally unique end-device identifier (DevEUI), the application identifier (AppEUI), and an AES-128 key (AppKey).

■ Activation by Personalization

Under certain circumstances, end-devices can be activated by personalization. Activation by personalization directly ties an end-device to a specific network by-passing the join request join accept procedure.
Activating an end-device by personalization means that the DevAddr and the two session keys NwkSKey and AppSKey are directly stored into the end-device instead of the DevEUI, AppEUI and the AppKey. The end-device is equipped with the required information for participating in a specific LoRa network when started. Each device should have a unique set of NwkSKey and AppSKey. Compromising the keys **of one device shouldn‘t compromise the security of the communications of other devices.**

Operation Mode

Bi-directional end-devices (Class A): End-devices of Class A allow for bi-directional communications whereby each end-device's uplink transmission is followed by two short downlink receive windows. The transmission slot scheduled by the end-device is based on its own communication needs with a small variation based on a random time basis (ALOHA-type of protocol). This Class A operation is the lowest power end-device system for applications that only require downlink communication from the server shortly after the end-device has sent an uplink transmission. Downlink communications from the server at any other time will have to wait until the next scheduled uplink.

UART INTERFACE

All of the LM-130 module’s settings and commands are transmitted over UART by using the ASCII interface.

All commands need to be terminated with <CR><LF> and any replies they generate will also be terminated by the same sequence.

Any setting change set by AT command will take effect only after execute “AAT1 Save” and “AAT1 Reset” command.

The settings for the UART interface are 57600 bps, 8 bits, no parity, 1 Stop bit, no flow control.

AT command

Command	Description
AAT1 UpdateFW	Upgrade the LM-130 module firmware. Response ***ok*** after entering the command
AAT1 Save	All parameters are saved. Response ***ok*** after parameters are saved.
AAT1 FwVersion	Show up firmware version.
AAT1 Reset	Resets and restarts the LM-130 module. Response ***ok*** after entering the command
AAT1 Restore	All parameters turn into default setting.
AAT1 SLEEP	Put LM-130 into sleep mode. Input 0xFF by UART to wake up LM-130 to leave sleep mode. Response ***ok*** after entering the command

Command	Parameter Description
AAT2 DevAddr=[parameter1]	[parameter1]: 4-byte hexadecimal number representing the device address, from 00000000 – FFFFFFFF Response: ***ok*** if address is valid ***invalid_param*** if parameter1 is not valid This command configures the module with a 4-byte unique network device address [parameter1]. The [parameter1] must be unique to the current network. This must be directly set solely for activation by personalization devices. This parameter must not be set before attempting to join using over-the-air activation because it will be overwritten once the join process is over.
AAT2 DevAddr=?	Response: 4-byte hexadecimal number representing

	the device address, from 00000000 to FFFFFFFF. This command will return present end-device address of the module.
Command	Parameter Description
AAT2 DevEui=[parameter1]	[parameter1]: 8-byte hexadecimal number representing the device EUI Response: ***ok*** if address is valid ***invalid_param*** if parameter1 is not valid This command sets the globally unique device identifier for the module. The identifier must be set by the host MCU. The module contains a pre-programmed unique EUI and can be retrieved using user provided EUI can be configured using the AAT2 DevEui command.
AAT2 DevEui=?	Response: 8-byte hexadecimal number representing the device EUI. This command returns the globally unique end-device identifier, as set in the module.

Command	Parameter Description
AAT2 AppEui=[parameter1]	[parameter1]: 8-byte hexadecimal number representing the application EUI Response: ***ok*** if address is valid ***invalid_param*** if parameter1 is not valid This command sets the application identifier for the module.
AAT2 AppEui=?	Response: 8-byte hexadecimal number representing the application EUI. This command will return the application identifier for the module. The application identifier is a value given to the device by the network.

Command	Parameter Description
AAT2 NwkSKey=[parameter1]	[parameter1]: 16-byte hexadecimal number representing the network session key Response: ***ok*** if address is valid ***invalid_param*** if parameter1 is not valid This command sets the network session key for the module. This key is 16 bytes in length, and should be modified with each session between the module and network. The key should remain the same until the communication session between devices is terminated.
AAT2 NwkSKey=?	Reponse: [parameter1]: 16-byte hexadecimal

	number representing the network session key This command sets the network session key for the module.

Command	Parameter Description
AAT2 AppSKey=[parameter1]	[parameter1]: 16-byte hexadecimal number representing the application session key Response: ***ok*** if address is valid ***invalid_param*** if parameter1 is not valid This command sets the application session key for the module. This key is unique, created for each occurrence of communication, when the network requests an action taken by the application.
AAT2 AppSKey=?	Response: [parameter1]: 16-byte hexadecimal number representing the application session key This command sets the application session key for the module.

Command	Parameter Description
AAT2 AppKey=[parameter1]	[parameter1]: 16-byte hexadecimal number representing the application key Response: ***ok*** if address is valid ***invalid_param*** if parameter1 is not valid This command sets the application key for the module. The application key is used to identify a grouping over module units which perform the same or similar task.
AAT2 AppKey=?	Response: [parameter1]: 16-byte hexadecimal number representing the application key This command sets the application key for the module.

Command	Parameter Description
AAT2 ADR=[parameter1]	[parameter1]: 0: disable 1: enable Response: ***ok*** if address is valid ***invalid_param*** if parameter1 is not valid This command sets if the adaptive data rate (ADR) is to be enabled, or disabled. The server is informed about the status of the module’s ADR in every uplink frame it receives from the ADR field in

	uplink data packet. If ADR is enabled, the server will optimize the data rate and the transmission power of the module based on the information collected from the network.
AAT2 ADR=?	Response: 0: disable 1: enable This command will return the state of the adaptive data rate mechanism.

Command	Parameter Description
AAT2 JoinMode=[parameter1]	[parameter1]: 0: ABP mode 1: OTAA mode Response: ***ok*** if address is valid ***invalid_param*** if parameter1 is not valid This command informs the ***module activation type.***
AAT2 JoinMode=?	Response: 0: ABP mode 1: OTAA mode This command will return the ***activation type*** of module.

Command	Parameter Description
AAT2 reTx=[parameter1]	[parameter1]: decimal number representing the number of retransmissions for an uplink confirmed packet, from 0 to 10. Response: ***ok*** if address is valid ***invalid_param*** if parameter1 is not valid This command sets the number of retransmissions to be used for an uplink confirmed packet, if no downlink acknowledgment is received from the server.
AAT2 reTx=?	Response: decimal number representing the number of retransmissions, from 0 to 10. This command will return the currently configured number of retransmissions which are attempted for a confirmed uplink communication when no downlink response has been received.

Command	Parameter Description
AAT2 RxDelay1=[parameter1]	[parameter1]:decimal number representing the

	delay between the transmission and the first reception window in microseconds, from 100000 to 10000000. Response: ***ok*** if address is valid ***invalid_param*** if parameter1 is not valid This command will set the delay between the transmission and the first reception window to the [parameter1] in microseconds. The delay between the transmission and the second Reception window is calculated in software as the delay between the transmission and the first Reception window + 1000000 (us).
AAT2 RxDelay1=?	Response: decimal number representing the interval, in milliseconds, for rxdelay1. This command will return the interval, in microseconds, for rxdelay1.

Command	Parameter description
AAT2 Tx=[parameter1], [parameter2], [parameter3]	[parameter1]: decimal number representing the port number, from ***1*** to ***223***. [parameter2]: string representing the uplink payload type, either ***cnf*** or ***uncnf*** (cnf-confirmed, uncnf-unconfirmed) [parameter3]: hexadecimal value. The length of [parameter3] bytes capable of being transmitted are dependent upon the set data rate (please refer to the LoRaWAN^{TM} Specification for further details). Response: this command may reply with two responses. The first response will be received immediately is valid (ok reply received), a second reply will be received after the end of the uplink transmission. Please refer to the the LoRaWAN^{TM} Specification for further details. Response after entering the command: ● ok - If parameters and configurations are valid. ● Invalid_param – if parameters ([parameter1],[parameter2],[parameter3]) are not valid. ● Tx_ok - if uncnf radio tx return. ● Tx_noACK - if cnf radio tx return without ack. ● Tx_ok - if cnf radio tx return with ack ● Rx < parameter1> < parameter2>– if transmission was successful, [parameter1] port number, from 1 to 223; [parameter2] hexadecimal value that was received from the server.