Interfacing with LoRaWAN

RM186 LoRa + BLE Module

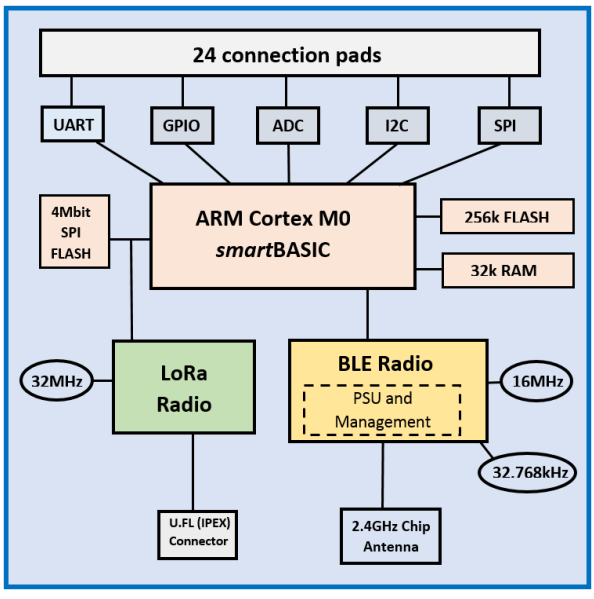
Application Note *v1.1*

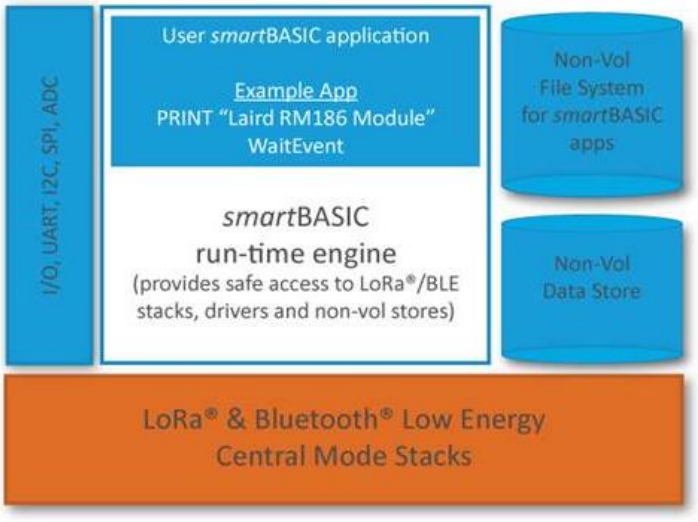
INTRODUCTION

The RM186 is a wireless communications module that combines a Nordic nRF51822_QFAC (256/16) BLE device and a Semtech SX1272 860 to 1020 MHz low power, long range transceiver. The RM186 is designed to function in the EU863-870MHz ISM band as opposed to the RM191 which is a very similar device designed for the US 902-928 MHz ISM band.

The RM186 can collect data from external sources by interfacing directly with a sensor over a UART, SPI or, I^2C serial communication channel or over a wireless BLE connection if a sensor is physically connected to a BLE peripheral device. Data can also be gathered internally using the analog or digital IO pins. Once the data has been collected and arranged in packets, it can then be transmitted to a remote LoRaWAN Gateway up to 16 kilometers away. From there it can be transmitted to a server or database over a TCP/IP link.

The purpose of this document is to describe how the RM186 interfaces with the LoRaWAN specification and what is happening inside the module.





Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0610

GATEWAYS AND SERVERS

LoRaWAN is a wireless standard developed by the LoRa Alliance. It enables low-power wide-area networking using low data rates and minimal power usage. The RM186 is designed to communicate with any LoRaWAN gateway that is within RF range, can receive a signal within the correct frequency range, and meets the requirements defined in the LoRaWAN specification.

The gateway does nothing with the data except to send it to a network server via a TCP/IP link. The network server must be running LoRaWAN-compatible software to handle the decryption of messages and authentication of devices. There are several vendors that provide such services.

It is possible that a packet could be received by a LoRa gateway that is not connected to the desired network. However, in this instance the data should be discarded by the network server as it cannot be decrypted correctly. The only network server that would be able to decrypt the data successfully is the one to which the RM186 is connected. This is explained in greater detail in the Connecting to the LoRaWAN Network section.

RM186 FREQUENCIES AND DUTY CYCLES

The RM186 module functions in the EU863-870 MHz ISM band, which is spilt into several sub-bands by the governing ETSI standard. Each sub-band has its own power and duty cycle requirements.

Each band also has a specific duty cycle restriction. If a specific band has used up all its duty cycle allowance and so cannot transmit a packet, then that packet can still be transmitted on a frequency in another band, provided that band still has available duty-cycle.

The sub-bands used in the RM186 are shown in the Table 1.

Table 1: RM1xx channel frequencies

Band	Edge Frequencies (MHz)		Duty Cycle (%)
0	865	868	1
1	868	868.6	1
2	868.7	869.2	0.1
3	869.4	869.65	10
4	869.7	869.7	1

It is mandatory for a LoRaWAN end-device to support the following three Band 1 frequencies: 868.1, 868.3, and 868.5 MHz. These frequencies are used during the OTAA network join procedure described below.

Additional frequencies can be added by the LoRaWAN gateway/server in any of the previously defined sub-bands.

If the RM186 successfully joins a network using OTAA, the LoRaWAN gateway/server transmits a JoinAccept message which could include a list of up to five additional frequencies on which the module could then transmit.

The LoRaWAN gateway/server could also configure additional frequencies using the LoRaWAN **NewChannelReq** command. This command, with the exception of the three mandatory channels mentioned above, could also modify existing frequency channels. Both of these actions are invisible to the user; they occur purely at the LoRaWAN stack level and no notification is provided regarding any changes.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

However, the list of available channels can be obtained using the following *smart*BASIC command:

LORAMACGetOption(LORAMAC_OPT_CHANNELLIST,var$)

This is a read-only command and is documented in the RM1xx *smart*BASIC extensions manual.

Note: This command only returns the list of enabled channels. There may be more channels configured, but if they are not enabled by the server through the **ChannelsMask** parameter then they are not listed. During the OTAA process the **ChannelsMask** is configured such that only the three mandatory Band 1 frequencies are enabled.

After joining, the ChannelsMask value can be modified as part of the LoRaWAN **JoinAccept, NewChannelReq**, and **LinkAdrReq** commands. Again, the user is not notified of any changes, however the current value of the ChannelsMask can be obtained through the following command:

LORAMACGetOption(LORAMAC_OPT_CHANNELMASK,var$)

As with the channel list, this is a read-only command.

Note: More details regarding all *smart*BASIC commands referenced in this application note can be found in the RM1xx *smart*BASIC Extensions Guide, found in the documentation tab of the RM1xx Series product page.

As well as the default frequency channels, there is also a requirement for a default second receive window frequency at 869.525 MHz. The purpose of this frequency channel is explained in the Data Reception below. It is a receive-only frequency and cannot be used to transmit data.

DATA RATE

The LoRaWAN specification states that the following data rates must be supported by the RM186.

Note: These figures are specific to CE operation with the RM186. The FCC rules and the RM191 vary.

Table 2: LoRaWAN data rates

Data Rate	Configuration	Physical Bit Rate (bit/s)
0	SF12 @ 125kHz	250
1	SF11 @ 125kHz	440
2	SF10 @ 125kHz	980
3	SF9 @ 125kHz	1760
4	SF8 @ 125kHz	3125
5	SF7 @ 125kHz	5470
6	SF7 @ 250kHz	11000
7	FSK:50kbps	50000

SF7 to SF12 refers to the spreading factor of the system. The spreading factor refers to how many chips of information represent each bit (or symbol) of payload data. The actual chip value is calculated by 2^x. For SF7 there are 2^7 (128) chips per symbol and for SF12 there are 2^{12} (4096) chips per symbol.

This explains why the physical bit rate is heavily dependent on the spreading factor. The higher the spreading factor, the more chips that must be sent for each bit of information and so, consequently, the longer the time it takes to transmit the data. This is very important when you look at the potential data throughput of a module.

At data rate 0, it takes approximately 1.6 seconds to transmit a packet with a 16-byte payload. This means that you are not able to send another packet in that band for approximately 160 seconds.

At data rate 5, that same packet takes approximately 66 milliseconds to transmit. Therefore, the module would be able to transmit another packet at that frequency or another frequency in that band after about 6.5 seconds.

The default configured data rate for the RM186 is data rate 5, SF7@125kHz. At any time, you can reconfigure this value using the **LORAMACSetOption(LORAMAC_OPT_DATA_RATE,var$)** command and read it back using the **LORAMACGetOption(LORAMAC_OPT_DATA_RATE,var$)** command. These are important commands as the server can modify the data rate at any time using the **LinkAdrReq** command.

As with any changes to the configuration of the frequency channels by the gateway/server, notification of changes to the data rate are not automatically passed back to the user. An event is thrown on receipt of an ADR command however; you would then have to call the *smart*BASIC **LORAMACGetOption** command to obtain the latest configuration.

Note: Increasing the data rate reduces the maximum range of the module by making the signal more susceptible to noise. At high spreading factors, the receiver can receive data at much lower signal strengths than it can with low spreading factors.

LoRaWAN Events

A series of events are raised by the LoRaWAN stack to indicate the success or failure of a specific task. These events are then routed through the RM186 firmware and finally output as *smart*BASIC events.

The events listed in Table 3 are referenced later in this document.

Table 3: LoRaWAN events

Event	Description
EVLORAMACJOINING	A JoinRequest is sent to the gateway/server.
EVLORAMACJOINED	The JoinRequest is successfully received by the gateway/server and the subsequent JoinAccept is received by the RM186. The module is now connected to the network.
EVLORAMACJOINFAILED	The JoinRequest failed.
EVLORAMACTXCOMPLETE	Sent when an uplink packet is transmitted. The actual timing of this event varies depending on the type of packet transmitted. In the case of a confirmed packet, this is sent at the same time as the EVLORAMACRXCOMPLETE event.
EVLORAMACRXCOMPLETE	A downlink packet is received by the RM186. This event is only sent if the uplink packet is configured as a confirmed packet.
EVLORAMACRXTIMEOUT	Only valid with confirmed uplink or LinkCheck packets. For confirmed packets, it is sent after the RM186 fails to receive confirmation of the same uplink packet after the configured number of transmission attempts. For Link Checks it is sent if the LinkCheck doesn't receive a response. A LinkCheck is not retransmitted.
EVLORAMACTXDONE	A signal from the radio indicating that a packet was transmitted. All receive timings are taken from this event.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

Event	Description
EVLORAMACNOSYNC	Notification that a receive window has closed without receiving a sync pulse.
EVLORAMACADR	Notification that and ADR command has been received from the gateway.

CONNECTING TO THE LORAWAN NETWORK

Before the RM186 can transmit any data to a LoRaWAN network it must first be connected to that network. As the RF link between the RM186 and a gateway is over a secure channel, this connection process involves the exchange of certain keys between both ends of the link.

This can either be achieved by calculating the keys afresh every connection or by configuring the RM186 and gateway server with the key information in advance.

Over-the-Air Authentication

The recommended method of connecting an RM186 to a network is by using the Over-the-Air Authentication (OTAA) method. With this method the RM186 must be configured with certain IDs so that the Network Session Key (NwkSKey) and the Application Session Key (AppSKey) can be calculated. These IDs are also transmitted to the server as part of the JoinRequest command so that the NwkSKey and AppSKey can be calculated and stored there. These keys are unique between a module and a server.

The following IDs must be configured for OTAA:

- **Application Identifier (AppEUI) –** This is a global application ID in IEEE EUI64 which uniquely identifies the application provider of the module.
- **End-device Identifier (DevEui) –** This is a global end-device ID in IEEE EUI64 which uniquely identifies the module/end-device.
- **Application Key (AppKey) –** This is an AES-128 application key specific to the module which is used to derive the session keys NwkSKey and AppSKey.

These values can be stored and read back from the RM186 using the commands listed in Table 4.

Table 4: OTAA IDs

ID	Data Length (Bytes)	Write Command	Read Command
AppEui	8	at+cfgex 1010 “xxxxxxxx” (For firmware versions prior to 18.4.1.0 the AppEui Id is 1000.)	at+cfgex 1010?
DevEui	8	N/A- Configured during production	ati 25
		at+cfgex 1011 “xxxxxxx” (For firmware versions prior to 18.4.1.0 the DevEui Id is 1001.)	at+cfgex 1011?
AppKey	16	at+cfgex 1012 “xxx…xxx” (For firmware versions prior to 18.4.1.0 the AppKey Id is 1002.)	Write Only

Note: The **xxxx** above represents a hexadecimal value. For example:

at+cfgex 1010 "12c87fc0da000001"

The IDs configured using the **at+cfgex** command only take effect after a module reset. This is not performed automatically as part of the command; it must be manually initiated.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

Note: There are two options for configuring the DevEui. The device is initially configured with a global DevEui during production. However, it is possible to override this value with a local definition. When it comes to selecting a value, the code uses the local value first if it is available. If not, it reverts to the global value. The **ati 25** command only returns the global value and **at+cfgex 1011?** only returns the local value.

On receipt of a **LORAMACJoin(LORAMAC_JOIN_BY_REQUEST)** command the module first checks that all the required parameters have been configured. If not, the join request is cancelled and an error message is returned.

Note: More details regarding all the *smart*BASIC commands referenced in this application note can be found in the RM1xx *smart*BASIC Extensions Guide, found in the documentation tab of the RM1xx Series product page.

If everything is working, then the join request is transmitted to the gateway/server. If successful, an **EVLORAMACJOINED** event is passed to the user. This event is not received for at least five seconds after the transmission of the join request command due to the preprogrammed delays in the system. This is explained in the Data Reception section and the expected delays are defined in Table 6 below.

If the JoinRequest fails, the firmware continues to resend the JoinRequest in accordance with the Duty Cycle restrictions until a JoinAccept is received. In firmware version 18.4.1.0, the JoinRequest is transmitted at data rate DR5. So with a duty cycle setting of 1% and a transmit time of 66 milliseconds, a new request can be sent in a little over six seconds.

Activation by Personalization

When using Activating by Personalization, the NwkSKey and the AppSkey must be configured on both the RM186 and any server with which it might communicate.

- **Network Session Key (NwkSKey):** Specific to the end device. Used by the module and gateway/server to calculate the checksum of all data messages. Also used to encrypt and decrypt the payload field of MAC only data messages.
- **Application Session Key (AppSKey):** Specific to the end device. Used by the module and gateway/server to encrypt and decrypt the payload data. It may also be used to calculate the optional payload checksum.
- **End Device Address (DevAddr):** 32 bits (4 bytes) that identifies the module within the current network.

Table 5: Personalization IDs

ID	Data Length (Bytes)	Write Command	Read Command
NwkSKey	16	at+cfgex 1013 “xxx.xxxx” (For firmware versions prior to 18.4.1.0 the NwkSKey Id is 1003.)	Write Only
AppSKey	16	at+cfgex 1014 “xxx.xxx” (For firmware versions prior to 18.4.1.0 the AppSKey Id is 1004.)	Write Only
DevAddr	4	at+cfgex 1015 “xxxx” (For firmware versions prior to 18.4.1.0 the DevAddr Id is 1005.)	at+cfgex 1005?

Note: The **xxxx** above represents a hexadecimal value. For example:

at+cfgex 1013 "2b7e151628aed2a6abf7158809cf4f3c"

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

As with the OTAA, once a **LORAMACJoin(LORAMAC_JOIN_BY_PERSONALIZATION)** is received by the module, it checks that the above IDs are configured. If not, an error message is returned and the request is cancelled.

When you join a network using the personalization method, there is no handshaking required between the RM186 and the server. Both sides of the link should be configured with the same key values and the RM186 is assumed to have joined the network as soon as the command is sent. The module is ready to start transmitting data to the gateway immediately.

If using the Multitech gateway in Network Server mode, refer to the Conduit mLinux: LoRa Use with Third-Party Devices webpage for details on how to configure the gateway.

DATA TRANSMISSION

Transmitting data to the server is a simple task. Call the *smart*BASIC **LoramacTxData** command containing the data you wish to send to the server. Everything else is handled by the firmware.

As with OTAA, data packets are subject to the same duty cycle restrictions. The LoRaWAN stack searches all the enabled frequency channels to determine which, if any, have duty cycle available. If there are any available, it randomly selects one of those channels and transmits the packet to the server. If not, the module waits for the first frequency channel to become available and then transmits the packet.

As before, you may determine when a packet is due to be transmitted by using the **LORAMACGetOption(LORAMAC_OPT_NEXT_TX,var$)** command. But again, the packet must first be loaded before this command outputs the correct value.

The full transmit cycle consists of a successful uplink message to the server and the successful reception of the downlink message. During this period, you cannot load another packet for transmission. More information regarding the downlink message is provided in the Data Reception section.

If successful and depending on whether the confirmed option is selected, an EVLORAMACRXCOMPLETE and/or an EVLORAMACTXCOMPLETE are raised. For the case of the confirmed packet, these events are sent at the same time – after the reception of the downlink packet. Therefore, either can be used to indicate the successful completion of the data transmission in the case of a confirmed packet. The events are basically duplicated. For non-confirmed packets only the EVLORAMACTXCOMPLETE is sent.

If the transmission fails, the confirmed/not confirmed option determines what happens next. If the confirmed option is selected and the confirmation is not received, the module attempts to retransmit the packet. By default, the module attempts to send a specific packet eight times, i.e. the initial attempt and seven reattempts. The number of retries can be configured using the **LORAMACGetOption(LORAMAC_OPT_MAX_RETRIES,var$)** command. The number of retries cannot exceed eight. If the number of retries is reached without a receiving a response, then an EVLORAMACRXTIMEOUT event is thrown.

During this retransmit process the LoRaWAN stack automatically decrements the data rate after every two failed transmissions. So, depending on the starting data rate, the data rate at the end of this process could be much lower than at the beginning, potentially causing this process to take longer than might be expected. The module tends towards decreasing the data rate as this has the effect of increasing the range of the module.

If an EVLORAMACRXTIMEOUT is received, it is up to the user how to proceed. You can either try to join the network again or try sending another packet.

This resend process is carried out behind the scenes. In firmware version 18.4.1.0, two new events were added to provide feedback to the user as to what is happening. If enabled, the EVLORAMACTXDATA event is thrown each time a packet is transmitted by the radio. Also, if a receive window is closed without receiving a sync pulse an

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

EVLORAMACNOSYNC event is thrown. Using these events, you are able to monitor when the module has transmitted a data packet but not received the expected downlink. A new *smart*BASIC command has also been created, LoramacSetDebug(), to aid with debugging the system. This command provides transmit and receive waveforms output on configurable SIO pins. Please refer to the corresponding app not for details.

If no confirmation is required, the EVLORAMACTXCOMPLETE is still received and the next packet can be transmitted. As far as the RM186 is concerned, all is well. It continues to send packets. Therefore, it is always advisable to select the confirm option or use the Link Check command to keep tabs on the strength of the connection.

DATA RECEPTION

After each transmitted uplink packet on a Class A LoRaWAN device, the module must listen for a downlink packet, which may or may not be sent from the gateway There are two possible receive windows in which this downlink packet could be transmitted, illustrated in Figure 1.

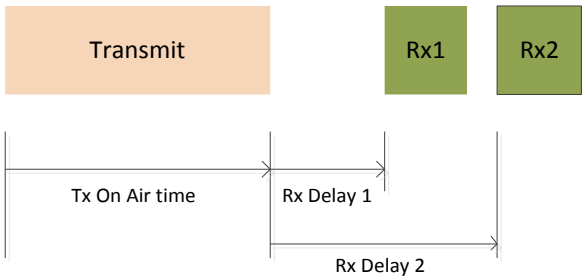


Figure 1: Receive window timings

The actual length of the delays (Table 6) is dependent on whether the RM186 is transmitting a join request or a normal data packet.

Table 6: Receive delay times

Packet Type	Rx Delay 1	Rx Delay 2
Join Request	5 seconds	6 seconds
Data Packet	1 second	2 seconds

In the first receive window, the gateway transmits on the same frequency as the uplink message. In the second receive window, the gateway transmits on the default 869.525 MHz at data rate DR0.

If the downlink packet is transmitted in the first window and is successfully received by the module, then the second window is not required and does not open.

If the downlink packet is not received during the first Receive window, then the second Receive window is opened and the module listens on 869.525 MHz. A likely scenario is that the gateway ran into duty cycle issues on the uplink frequency channel and had no time remaining for that band.

As shown in Figure 1, all the timings are taken from the end of the uplink transmission. This is the EVLORAMACTXDONE event. At the designated interval, the RM186 turns on the receiver for a short period and listens for a sync message. If the signal is not detected, the receiver is switched off and the RM186 then waits for the second window to open and repeat the process.

Americas: +1-800-492-2320
Europe: +44-1628-858-940
Hong Kong: +852 2923 0600

If the sync message is detected in any of the receive windows, then the RM186 receiver remains on until the full downlink packet is received. In this way the RM186 saves power by only having the receiver switched on for the minimum amount of time.

Note: If the sync message is transmitted outside this initial small receive window, the downlink packet is missed. The LoRa gateway has no knowledge of this fact. As far as the gateway is concerned, the packet was sent. It is not retransmitted in the second window.

Again, the uplink packet option (confirmed or not confirmed) determines the RM186's response to this. If confirmed is selected, then the same uplink packet is resent, following the procedure outlined above.

Receive timings are hardcoded. There are no configuration options available to the user to modify these timings. The RM186 timings meet those of the LoRaWAN specification. It is the gateway's responsibility to send the downlink packet at the correct time.

LINK CHECK

A module can transmit a Link Check request to the server. This command contains no payload.

The Link Check response from the server contains an indication of the signal strength of the last Link Check request received by the server and the number of gateways that have received that request.

REFERENCES

- Lora Alliance - LoRaWAN Specification – The current version is available from the LoRa Alliance website.
- LoRaWAN Regional Parameters – The current version is available from the LoRa Alliance website.
- User Guide – RM1xx Series *smart*BASIC Extensions (documentation tab of RM1xx product page)
- User Guide - *smart*BASIC Core Functionality (documentation tab of RM1xx product page)
- Application Note – Connecting to Multitech Conduit Gateway (documentation tab of RM1xx product page)
- http://www.multitech.net/developer/software/lora/conduit-mlinux-lora-communication/conduit-mlinux-lora-use-third-party-devices/

REVISION HISTORY

Version	Date	Notes	Approver
1.0	20 May 2016	Initial Release	Colin Anderson
1.1	10 Oct 2016	Updates to several ID values	Colin Anderson