RL78/G23, RL78/G14

LoRaWAN® stack reference guide

Introduction

This application note describes information to use the LoRaWAN® stack and its APIs.

Target Device

MCU: RL78/G23 (R7F100GSN, R7F100GLG) or RL78/G14 (R5F104ML)
Transceiver: Semtech SX1261 or SX1262

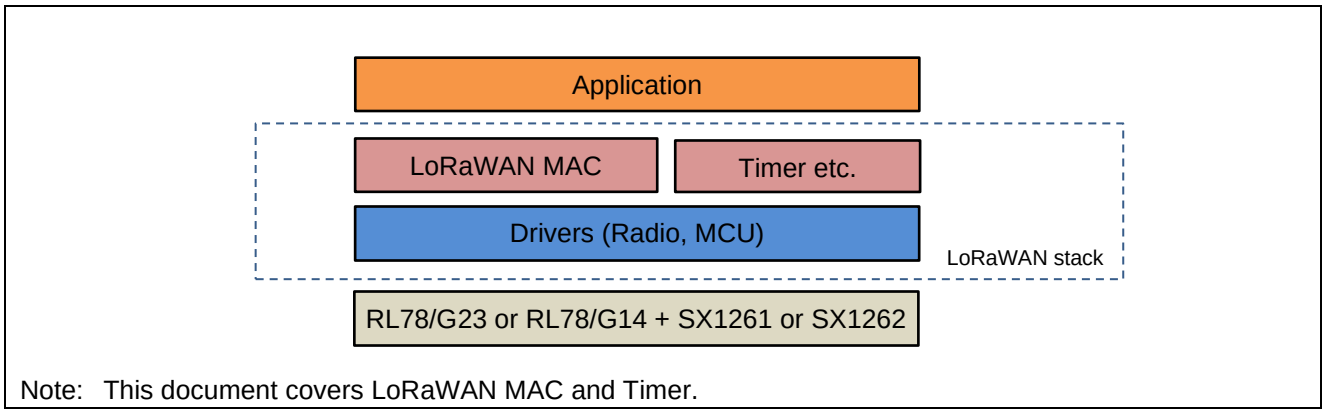
Contents

1. Overview

This application note contains API references and other information to use the LoRaWAN stack. The LoRaWAN stack includes LoRaWAN interface which implements LoRaWAN protocol (Class A/B/C) specified in the specification version 1.0.3 and 1.0.4.
The LoRaWAN interfaces are described in Section 2. The Timer interface is described in Section 3. Application can also use Timer APIs, although they are used in the stack.

1.1 LoRaWAN stack block diagram

Figure 1.1 shows a block diagram of the LoRaWAN Stack.



Note: This document covers LoRaWAN MAC and Timer.

Figure 1.1 LoRaWAN stack block diagram

1.2 Directories (informative)

Table 1-1 shows a basic concept of what kind of codes each directory includes. This is just informative.

Table 1-1 Directories

Directories	Description
apps	Application code
boards	Board specific codes
boards/mcu	MCU drivers
mac	LoRaWAN MAC stack
radio	Radio driver for LoRa®
peripherals	Security related codes
system	Utility APIs, etc.

1.3 Resource usage (informative)

Table 1-2, Table 1-3, and Table 1-4 shows the resources used by the LoRaWAN stack and drivers.

Table 1-2 Resources

MCU Peripherals (RL78/G23 - R7F100GSN)
- 32 bit Interval Timer - Timer Array Unit (TAU) : Unit 0 Channle2 (TAU02) - External Interrupt: INTP11(P77) - Serial Array Unit (SAU): Unit 1 Channel 1 (CSI11) - SCK11(P95), SI11(P96), SO11(P97) - I/O Port: P115, P46, P106, P42

Table 1-3 Resources

MCU Peripherals (RL78/G23 - R7F100GLG)
- 32 bit Interval Timer - Timer Array Unit (TAU) : Unit 0 Channle2 (TAU02) - External Interrupt: INTP11(P77) - Serial Array Unit (SAU): Unit 1 Channel 1 (CSI11) - SCK11(P30), SI11(P50), SO11(P51) - I/O Port: P22, P76, P42, P73

Table 1-4 Resources

MCU Peripherals (RL78/G14 - R5F104ML)
- Real Time Clock(RTC) - Timer RJ - 12 bit Interval Timer - External Interrupt: INTP11(P77) - Serial Array Unit (SAU): Unit 3 Channel 1 (CSI31) - SCK31(P54), SI31(P53), SO31(P52) - I/O Port: P26, P02, P75, P03

1.4 Acronyms and abbreviations

Table 1-5 Acronyms and abbreviations

Acronyms	Description
ABP	Activation By Personalization
EIRP	Equivalent Isotropic Radiated Power
MCPS	MAC Common Part Sublayer
MLME	MAC Layer Management Entity
OTAA	Over-The-Air-Activation
RFU	Reserved for Future Use
BW	Modulation bandwidth
DR	Data rate
US915	US902-928MHz ISM Band
EU868	EU863-870MHz ISM Band
AS923	AS923MHz ISM Band
Group AS923-1	Composed of Asia countries having available frequencies in the 915-928MHz range
Group AS923-2	Composed of Asia countries having available frequencies in the 920-923MHz range
Group AS923-3	Composed of Asia countries having available frequencies in the 915-921MHz range
Group AS923-4	Composed of Asia countries having available frequencies in the 917-920MHz range
AS923-Japan	Perform ARIB STD-T108 regulation for using in Japan

1.5 Related Documentation

	Document No.	Title	Author	Language
[1]	R11AN0227EJ	Radio Driver Reference Guide	Renesas Electronics	English
[2]	R11AN0393JJ	Radio Driver for Japan Radio Regulations	Renesas Electronics	Japanese

2. LoRaWAN interface

This section describes the LoRaWAN stack interfaces.

2.1 Macros

This section includes the following enumeration types.

2.1.1 Data rate

This section includes Data rate index definitions. Actual parameters are defined for each region.

Table 2-1 Data rate

Macro	Value	Description
DR_0	0	DR0 AS923: LoRa®: SF12 - BW125 kHz (250 bps) EU868: LoRa®: SF12 - BW125 kHz (250 bps) US915: LoRa®: SF10 - BW125 kHz (980 bps)
DR_1	1	DR1 AS923: LoRa®: SF11 - BW125 kHz (440 bps) EU868: LoRa®: SF11 - BW125 kHz (440 bps) US915: LoRa®: SF9 - BW125 kHz (1760 bps)
DR_2	2	DR2 AS923: LoRa®: SF10 - BW125 kHz (980 bps) EU868: LoRa®: SF10 - BW125 kHz (980 bps) US915: LoRa®: SF8 - BW125 kHz (3125 bps)
DR_3	3	DR3 AS923: LoRa®: SF9 - BW125 kHz (1760 bps) EU868: LoRa®: SF9 - BW125 kHz (1760 bps) US915: LoRa®: SF7 - BW125 kHz (5470 bps)
DR_4	4	DR4 AS923: LoRa®: SF8 - BW125 kHz (3125 bps) EU868: LoRa®: SF8 - BW125 kHz (3125 bps) US915: LoRa®: SF8 - BW500 kHz (12500 bps)
DR_5	5	DR5 AS923: LoRa®: SF7 - BW125 kHz (5470 bps) EU868: LoRa®: SF7 - BW125 kHz (5470 bps) US915: RFU
DR_6	6	DR6 AS923: LoRa®: SF7 - BW250 kHz (11000 bps) EU868: LoRa®: SF7 - BW250 kHz (11000 bps) US915: RFU
DR_7	7	DR7 AS923: FSK: 50 kbps EU868: FSK: 50 kbps US915: RFU
DR_8	8	DR8 AS923: RFU EU868: RFU US915: LoRa®: SF12 - BW500 kHz (980 bps)
DR_9	9	DR9 AS923: RFU

		EU868: RFU US915: LoRa®: SF11 - BW500 kHz (1760 bps)
DR_10	10	DR10 AS923: RFU EU868: RFU US915: LoRa®: SF10 - BW500 kHz (3900 bps)
DR_11	11	DR11 AS923: RFU EU868: RFU US915: LoRa®: SF9 - BW500 kHz (7000 bps)
DR_12	12	DR12 AS923: RFU EU868: RFU US915: LoRa®: SF8 - BW500 kHz (12500 bps)
DR_13	13	DR13 AS923: RFU EU868: RFU US915: LoRa®: SF7 - BW500 kHz (21900 bps)
DR_14	14	DR14 AS923: RFU EU868: RFU US915: RFU
DR_15	15	DR15 AS923: RFU EU868: RFU US915: RFU

2.1.2 Transmission power

This subsection defines transmission power macros. Actual power levels corresponding to each value are region specific. MaxEIRP in the table is considered to be +16 dBm by default.

Table 2-2 Transmission power

Macro	Value	Description
TX_POWER_0	0	AS923: MaxEIRP EU868: MaxEIRP US915: +30 [dBm]
TX_POWER_1	1	AS923: MaxEIRP – 2 [dB] EU868: MaxEIRP – 2 [dB] US915: +28 [dBm]
TX_POWER_2	2	AS923: MaxEIRP – 4 [dB] EU868: MaxEIRP – 4 [dB] US915: +26 [dBm]
TX_POWER_3	3	AS923: MaxEIRP – 6 [dB] EU868: MaxEIRP – 6 [dB] US915: +24 [dBm]
TX_POWER_4	4	AS923: MaxEIRP – 8 [dB] EU868: MaxEIRP – 8 [dB] US915: +22 [dBm]
TX_POWER_5	5	AS923: MaxEIRP – 10 [dB] EU868: MaxEIRP – 10 [dB] US915: +20 [dBm]

TX_POWER_6	6	AS923: MaxEIRP – 12 [dB] EU868: MaxEIRP – 12 [dB] US915: +18 [dBm]
TX_POWER_7	7	AS923: MaxEIRP – 14 [dB] EU868: MaxEIRP – 14 [dB] US915: +16 [dBm]
TX_POWER_8	8	AS923: RFU EU868: RFU US915: +14 [dBm]
TX_POWER_9	9	AS923: RFU EU868: RFU US915: +12 [dBm]
TX_POWER_10	10	AS923: RFU EU868: RFU US915: +10 [dBm]
TX_POWER_11	11	AS923: RFU EU868: RFU US915: +8 [dBm]
TX_POWER_12	12	AS923: RFU EU868: RFU US915: +6 [dBm]
TX_POWER_13	13	AS923: RFU EU868: RFU US915: +4 [dBm]
TX_POWER_14	14	AS923: RFU EU868: RFU US915: +2 [dBm]
TX_POWER_15	15	AS923: RFU EU868: RFU US915: RFU

2.1.3 Battery level

This subsection defines the battery level indication macros. Values from 2 (0x02) to 253 (0xFD) are available for board specific battery levels and there's no macro defined for the values.

Table 2-3 Battery level

Macro	Value	Description
BAT_LEVEL_EXT_SRC	0x00	External power source is used
BAT_LEVEL_EMPTY	0x01	Battery is empty
BAT_LEVEL_FULL	0xFE	Battery is full
BAT_LEVEL_NO_MEASURE	0xFF	Battery measurement is not available

2.1.4 Stack settings

This subsection defines stack configurations macros. Macros in this subsection need to be defined in the project build option.

Table 2-4 Macros for the stack setting

Macro	Description
REGION_AS923	Enable AS923 feature [LoRaWAN 1.0.4 only] Enables all groups of AS923 (AS923-1, AS923-2, AS923-3, AS923-4, and AS923-1 for Japan)
REGION_EU868	Enable EU868 feature
REGION_US915	Enable US915 feature
LORAMAC_CLASSB_ENABLED	Enable class B function
LORAWAN_VERSION_1_0_4	Support LoRaWAN protocol version 1.0.4 and LoRaWAN Regional Parameters RP002-1.0.3. (LORAWAN_VERSION_1_0_3 cannot be specified simultaneously.)
LORAWAN_VERSION_1_0_3	Support LoRaWAN protocol version 1.0.3 and LoRaWAN 1.0.3 Regional Parameters Revision A. (It can be omitted, i.e. default version is 1.0.3.)

2.1.5 Configuration

Parameters available for configuration in the LoRaWAN stack are defined in “LoRaMacConfig.h” and “LoRaMacClassBConfig.h”.

Table 2-5 Macros for the stack configuration (LoRaMacConfig.h)

Macro	Range	Default	Description
LORAMAC_ENABLE_CCA_DEFAULT	false, true	true	Default configuration for AS923 CCA operation
LORAMAC_STACK_PROCTIMEMS_RX1ON	> 0	8 (*1)(*2) 9 (*1)(*3)	Processing time from Tx done to RxWindow1 start
LORAMAC_STACK_PROCTIMEMS_RX2ON	> 0	8 (*1)(*2) 9 (*1)(*3)	Processing time from Tx done to RxWindow2 start
CLASSB_STACK_PROCTIMEMS_BEACON_ACQISITION	> 0	7 (*1)	Processing time from beacon timing to beacon window start for beacon acquisition
CLASSB_STACK_PROCTIMEMS_BEACON	> 0	10 (*1)	Processing time from beacon timing to beacon window start for beacon tracking
CLASSB_STACK_PROCTIMEMS_PING_SLOT_SHORT_PERIOD	> 0	3 (*1)	Processing time from ping slot timing to ping slot window start for short periodicity
CLASSB_STACK_PROCTIMEMS_PING_SLOT_LONG_PERIOD	> 0	7 (*1)	Processing time from ping slot timing to ping slot window start for long periodicity
CLASSB_BEACON_SYSTIMEDELTA_MAXERROR	> 0	(int32_t) (CLASSB_BEACON_INTERVAL * BOARD_CLOCK_ERROR_PPM / 1000000)	Maximum error of system time in a beacon interval [msec]. BOARD_CLOCK_ERROR_PPM defined in ‘board.h’ is used as the typical clock error [ppm].
CLASSB_BEACON_SYSTIMEDELTA_MINERROR	< 0	(int32_t) (CLASSB_BEACON_SYSTIMEDELTA_MAXERROR * (-1))	Minimum error of system time in a beacon interval [msec].
CLASSB_BEACON_SYSTIMEDELTA_INITERROR	-	0	Initial error of system time in a beacon interval [msec].

*1) In case CPU clock is 8MHz, *2) In case of RL78/G23, *3) In case of RL78/G14

Table 2-6 Macros for the stack configuration (LoRaMacClassBConfig.h)

Macro	Range	Default	Description
CLASSB_PING_SLOT_PERIODICITY_DEFAULT	4 - 7	7	Default ping slot periodicity
CLASSB_PING_SLOT_PERIODICITY_RFSLEEP_THRESHOLD	4 - 7	4	Threshold for the ping slot periodicity to select RF sleep mode. If the ping slot periodicity is set to the value of this threshold or more, the cold sleep is used for RF sleep mode. Otherwise, the warm sleep is used.

2.2 Enumerations

This section includes the following enumeration types.

Table 2-7 Enumerations

Types	Description
DeviceClass_t	LoRaWAN device class
LoRaMacRegion_t	Region and band
LoRaMacStatus_t	The status of the requested MAC services
LoRaMacEventInfoStatus_t	The status of the events of the MAC services
Mcps_t	MAC data service type
Mlme_t	MAC management service type
Mib_t	MAC Information Base (MIB) type
LoRaMacRxSlot_t	MAC receive window type
ActivationType_t	Activation type
LoRaMacErrorNotificationStatus_t	The status of the error notified by MacErrorNofity callback

2.2.1 DeviceClass_t

This type is enumeration containing LoRaWAN device class definitions.

Table 2-8 DeviceClass_t

Enumerator	Description
CLASS_A	LoRaWAN device class A
CLASS_B	LoRaWAN device class B
CLASS_C	LoRaWAN device class C

2.2.2 LoRaMacRegion_t

This type is enumeration containing MAC region and frequency band.

Table 2-9 LoRaMacRegion_t

Enumerator	Description
LORAMAC_REGION_EU868	Europe, band 868MHz
LORAMAC_REGION_US915	North America, band 915MHz
LORAMAC_REGION_AS923	Asia, band 923MHz, group AS923-1
LORAMAC_REGION_AS923_2	Asia, band 923MHz, group AS923-2
LORAMAC_REGION_AS923_3	Asia, band 923MHz, group AS923-3
LORAMAC_REGION_AS923_4	Asia, band 923MHz, group AS923-4
LORAMAC_REGION_AS923_JPN	Asia, band 923MHz, group AS923-1 for Japan

2.2.3 LoRaMacEventInfoStatus_t

This type is enumeration containing the status of the operation of a MAC service.

Table 2-10 LoRaMacEventInfoStatus_t

Enumerator	Description
LORAMAC_EVENT_INFO_STATUS_OK	Service performed successfully
LORAMAC_EVENT_INFO_STATUS_ERROR	An error occurred during the execution of the service
LORAMAC_EVENT_INFO_STATUS_TX_TIMEOUT	A Tx timeout occurred
LORAMAC_EVENT_INFO_STATUS_RX1_TIMEOUT	An Rx timeout occurred on receive window 1
LORAMAC_EVENT_INFO_STATUS_RX2_TIMEOUT	An Rx timeout occurred on receive window 2
LORAMAC_EVENT_INFO_STATUS_RX1_ERROR	An Rx error occurred on receive window 1
LORAMAC_EVENT_INFO_STATUS_RX2_ERROR	An Rx error occurred on receive window 2
LORAMAC_EVENT_INFO_STATUS_JOIN_FAIL	An error occurred in the join procedure
LORAMAC_EVENT_INFO_STATUS_JOIN_NONCE_FAIL	An error occurred in the join procedure (AppNonce(JoinNonce) value is same as the one previously received from the LoRaWAN server)
LORAMAC_EVENT_INFO_STATUS_DOWNLINK_REPEATED	A frame with an invalid downlink counter was received. The downlink counter of the frame was equal to the local copy of the downlink counter of the node.
LORAMAC_EVENT_INFO_STATUS_TX_DR_PAYLOAD_SIZE_ERROR	The MAC could not retransmit a frame since the MAC decreased the data rate. The payload size is not applicable for the data rate
LORAMAC_EVENT_INFO_STATUS_DOWNLINK_TOO_MANY_FRAMES_LOSS	The node has lost more than maximum number of lost frames
LORAMAC_EVENT_INFO_STATUS_ADDRESS_FAIL	An address error occurred
LORAMAC_EVENT_INFO_STATUS_MIC_FAIL	Message integrity check failure
LORAMAC_EVENT_INFO_STATUS_BEACON_LOCKED	Successfully received a beacon frame
LORAMAC_EVENT_INFO_STATUS_BEACON_NOT_FOUND	Beacon acquisition failure
LORAMAC_EVENT_INFO_BEACON_LOST	Could not receive a beacon frame

2.2.4 LoRaMacStatus_t

This type is enumeration containing MAC status. This indicates the result of requested MAC service.

Table 2-11 LoRaMacStatus_t

Enumerator	Description
LORAMAC_STATUS_OK	Service started successfully
LORAMAC_STATUS_BUSY	Error - Processing request
LORAMAC_STATUS_SERVICE_UNKNOWN	Error - Unknown request
LORAMAC_STATUS_PARAMETER_INVALID	Error - Invalid parameter
LORAMAC_STATUS_FREQUENCY_INVALID	Error - Unacceptable frequency for the region
LORAMAC_STATUS_DATARATE_INVALID	Error - Unacceptable data rate for the region
LORAMAC_STATUS_FREQ_AND_DR_INVALID	Error - Unacceptable frequency and data rate for the region
LORAMAC_STATUS_NO_NETWORK_JOINED	Error - Device is not in a LoRaWAN network
LORAMAC_STATUS_LENGTH_ERROR	Error - FOpts and payload length is too long
LORAMAC_STATUS_REGION_NOT_SUPPORTED	Error - Specified region is not supported
LORAMAC_STATUS_SKIPPED_APP_DATA	Error - Only MAC commands have been sent. Need to retry sending application data
LORAMAC_STATUS_DUTYCYCLE_RESTRICTED	Error - Transmission was aborted due to duty cycle restriction
LORAMAC_STATUS_NO_CHANNEL_FOUND	Error - Data rate not supported by any channel
LORAMAC_STATUS_NO_FREE_CHANNEL_FOUND	Error - No free channel found by carrier sense
LORAMAC_STATUS_BUSY_BEACON_RESERVED_TIME	Error - Transmission was aborted due to overlap with beacon reserved time
LORAMAC_STATUS_BUSY_PING_SLOT_WINDOW_TIME	Error - Transmission was aborted due to an overlapping ping slot
LORAMAC_STATUS_BUSY_UPLINK_COLLISION	Error - Transmission was aborted due to overlap with beacon operation timing
LORAMAC_STATUS_MC_GROUP_UNDEFINED	Error - Multicast groups are undefined
LORAMAC_STATUS_ERROR	Error - Undefined error
LORAMAC_STATUS_RADIO_FAIL	Error - Radio driver initialization failure
LORAMAC_STATUS_RADIO_PARAMETER_INVALID	Error - Radio parameter configuration is invalid

2.2.5 Mcps_t

This type is enumeration containing MAC data services. These are used to request a service or indicate the service of the response.

Table 2-12 Mcps_t

Enumerator	Description
MCPS_UNCONFIRMED	Unconfirmed Data frame
MCPS_CONFIRMED	Confirmed Data frame

2.2.6 Mlme_t

This type is enumeration of MAC management services.

Table 2-13 Mlme_t

Enumerator	Description
MLME_JOIN	Initiates the Over-the-Air activation
MLME_LINK_CHECK	Issues MAC command “LinkCheckReq“ for connectivity validation
MLME_DEVICE_TIME	Issues MAC command “DeviceTimeReq“ to request current GPS time
MLME_PING_SLOT_INFO	Issues MAC command “PingSlotInfoReq” to notify ping slot information
MLME_BEACON_ACQUISITION	Initiates beacon acquisition
MLME_SCHEDULE_UPLINK	Indicates that the application shall perform an uplink as soon as possible.
MLME_BEACON	Indicates whether a beacon was successfully received
MLME_BEACON_LOST	Indicates MAC layer fails to receive a beacon for 120 minutes

2.2.7 Mib_t

This type is enumeration containing the LoRaWAN MAC Information Base (MIB). These are used to get or to set the parameters in LoRaWAN stack.

Table 2-14 Mib_t

Enumerator	Description
MIB_DEVICE_CLASS	LoRaWAN device class
MIB_NETWORK_ACTIVATION	Activation type
MIB_DEV_EUI	Device EUI (DevEUI)
MIB_APP_EUI	Application EUI (AppEUI)
MIB_ADR	Adaptive data rate
MIB_NET_ID	Network identifier (NetID)
MIB_DEV_ADDR	End-device address (DevAddr)
MIB_APP_KEY	Application key (AppKey) for OTAA
MIB_GEN_APP_KEY	Application root key for multicast
MIB_NWK_SKEY	Network session key (NwkSKey)
MIB_APP_SKEY	Application session key (AppSKey)
MIB_MC_NWK_S_KEY_n	Network session key for multicast (McNwkSKey). “n” means the index of the multicast group and an integer in 0 - 3.
MIB_MC_APP_S_KEY_n	Application session key for multicast (McAppSKey). “n” means the index of the multicast group and an integer in 0 - 3.
MIB_PUBLIC_NETWORK	Network type, public or private
MIB_CHANNELS	LoRaWAN channels
MIB_RX2_CHANNEL	Receive window 2 channel
MIB_RX2_DEFAULT_CHANNEL	Default receive window 2 channel
MIB_CHANNELS_NB_TRANS	Maximum number of frame retransmission for unconfirmed uplink transmission.
MIB_RECEIVE_DELAY_1	Receive delay 1 in [ms]
MIB_RECEIVE_DELAY_2	Receive delay 2 in [ms]
MIB_JOIN_ACCEPT_DELAY_1	Join accept delay 1 in [ms]
MIB_JOIN_ACCEPT_DELAY_2	Join accept delay 2 in [ms]
MIB_CHANNELS_DATARATE	Data rate of a channel
MIB_CHANNELS_TX_POWER	Transmission power of a channel

MIB_CHANNELS_DEFAULT_TX_POWER	Default transmission power of a channel
MIB_ABP_LORAWAN_VERSION	LoRaWAN version
MIB_PING_SLOT_DATARATE	Ping slot data rate
MIB_PING_SLOT_PERIODICITY	Ping slot periodicity
MIB_AS923_ENABLE_CCA	CCA configuration for AS923 in bool. CCA enabled when true (default), and CCA disabled otherwise
MIB_IS_CERT_FPORT_ON	LoRaWAN certification FPort handling state
MIB_DEV_NONCE	Device nonce (DevNonce)
MIB_APP_NONCE	Application nonce (AppNonce) (or Join nonce (JoinNonce))
MIB_MAX_DCYCLE	Maximum duty cycle
MIB_RX1_DROFFSET	RX1 data rate offset (RX1DRoffset)
MIB_MAX_EIRP	Maximum allowed Effective Isotropic Radiated Power (EIRP)
MIB_DOWNLINK_DWELLTIME	Downlink dwell time (Maximum downlink transmit duration)
MIB_UPLINK_DWELLTIME	Uplink dwell time (Maximum uplink transmit duration)
MIB_DOWNLINK_FCNT	Downlink frame counter
MIB_UPLINK_FCNT	Uplink frame counter

2.2.8 LoRaMacRxSlot_t

This type is enumeration of the receive window type.
Table 2-15 LoRaMacRxSlot_t

Enumerator	Description
RX_SLOT_WIN_1	Receive window 1
RX_SLOT_WIN_2	Receive window 2
RX_SLOT_WIN_CLASS_C	Receive window 2 for Class C, continuous listening.
RX_SLOT_WIN_CLASS_C_MULTICAST	Class C multicast downlink window
RX_SLOT_WIN_CLASS_B_MULTICAST_SLOT	Class B multicast ping slot window
RX_SLOT_WIN_CLASS_B_PING_SLOT	Class B unicast ping slot window
RX_SLOT_NONE	No active reception window

2.2.9 ActivationType_t

This type is enumeration of end device activation types.

Table 2-16 ActivationType_t

Enumerator	Description
ACTIVATION_TYPE_NONE	None
ACTIVATION_TYPE_ABP	ABP
ACTIVATION_TYPE_OTAA	OTAA

2.2.10 LoRaMacErrorNotificationStatus_t

This type is enumeration of error notified by MacErrorNofity callback.

Table 2-17 LoRaMacErrorNotificationStatus_t

Enumerator	Description
LORAMAC_ERROR_NOTIFICATION_STATUS_RADIO_CHECK_FAIL_RX_CFG	Radio RX parameter configuration failure

2.3 Structure Types

This section describes the following types.

Table 2-18 Structure Types

Types	Description
McpsReq_t	MCPS-Request primitive
McpsReqUnconfired_t	MCPS-Request for an unconfirmed frame
McpsReqConfirmed_t	MCPS-Request for a confirmed frame
McpsConfirm_t	MCPS-Confirm primitive
McpsIndication_t	MCPS-Indication primitive
MlmeReq_t	MLME-Request primitive
MlmeReqJoin_t	MLME-Request for Join service
MlmeConfirm_t	MLME-Confirm primitive
MlmeIndication_t	MLME-Indication primitive
MlmeReqPingSlotInfo_t	MLME-Request(ping slot info service)
MibParam_t	MIB parameters related MIB type MIB_*
MibRequestConfirm_t	MIB-RequestConfirm primitive
LoRaMacTxInfo_t	Tx information
RxChannelParams_t	Receive window channel parameters
McChannelSetup_t	Multicast channel parameter
McRxParams_t	Multicast reception parameter
LoRaMacCtxs_t	MAC operational context
BeaconInfo_t	Beacon information

2.3.1 McpsReq_t

This type is structure containing MAC MCPS-Request

Table 2-19 McpsReq_t

Member type	Member	Description	
Mcps_t	Type	MCPS-Request type. See 2.2.5 Mcps_t.	
union sMcpsReq::uMcpsParam	Req	Parameters for each MCPS-Request set in Type. See the below in detail.	
RequestReturnParams_t	ReqReturn	Return parameter of MCPS-Request.	
		TimerTime_t DutyCycleWaitTime	Report the time in milliseconds which an application must wait before it is possible to send the next uplink.

Note: uMcpsParam is union containing MCPS-Request parameters.

Table 2-20 uMcpsParam

Member type	Member	Description
McpsReqUnconfirmed_t	Unconfirmed	Parameters for an unconfirmed frame. See 2.3.2 McpsReqUnconfirmed_t.
McpsReqConfirmed_t	Confirmed	Parameters for a confirmed frame. See 2.3.3 McpsReqConfirmed_t

2.3.2 McpsReqUnconfirmed_t

This type is structure containing MAC MCPS-Request for an unconfirmed frame.

Table 2-21 McpsReqUnconfirmed_t

Member type	Member	Description
uint8_t	fPort	Frame port number. Must be set if the payload is not empty. As for application specific frame, set the value within the range of 1 to 223.
Void *	fBuffer	Pointer to the buffer of the frame payload
uint16_t	fBufferSize	Size of the frame payload [0..250]
int8_t	Datarate	Uplink data rate if ADR is off. See 2.1.1 Data rate.

2.3.3 McpsReqConfirmed_t

This type is structure containing MAC MCPS-Request for a confirmed frame.

Table 2-22 McpsReqConfirmed_t

Member type	Member	Description
uint8_t	fPort	Frame port number. Must be set if the payload is not empty. As for application specific frame, set the value within the range of 1 to 223.
Void *	fBuffer	Pointer to the buffer of the frame payload
uint16_t	fBufferSize	Size of the frame payload [0..250]
int8_t	Datarate	Uplink data rate if ADR is off. See 2.1.1 Data rate.
uint8_t	NbTrials	Maximum number of trials to transmit the frame, if the MAC layer cannot receive an acknowledgment. The stack tries to send at least once even if this value is 0, and maximum 8 times even if this value is over 8.

2.3.4 McpsConfirm_t

This type is structure containing MAC MCPS-Confirm.

Table 2-23 McpsConfirm_t

Member type	Member	Description
Mcps_t	McpsRequest	Holds the previously performed MCPS-Request
LoRaMacEventInfoStatus_t	Status	Status of the operation
uint8_t	Datarate	Uplink data rate
int8_t	TxPower	Transmission power
bool	AckReceived	Set if an acknowledgement was received
uint8_t	NbRetries	Provides the number of retransmissions
TimerTime_t	TxTimeOnAir	The transmission time on air of the frame
uint32_t	UpLinkCounter	The uplink counter value related to the frame
uint32_t	Channel	The uplink channel index related to the frame

2.3.5 McpsIndication_t

This type is structure containing MAC MLME-Request for the join service.

Table 2-24 McpsIndication_t

Member type	Member	Description
Mcps_t	McpsIndication	MCPS-Indication type
LoRaMacEventInfoStatus_t	Status	Status of the operation
uint8_t	Multicast	Multicast
uint8_t	Port	Application port
uint8_t	RxDatarate	Downlink data rate
uint8_t	FramePending	Frame pending status
uint8_t *	Buffer	Pointer to the received data stream
uint8_t	BufferSize	Size of the received data stream
bool	RxData	Indicates if data is available [true: available, false: unavailable]
int16_t	Rssi	RSSI of the received packet
int8_t	Snr	SNR of the received packet
LoRaMacRxSlot_t	RxSlot	Receive window
bool	AckReceived	Indicates if an acknowledgement was received [true: Received, false: Not received]
uint32_t	DownLinkCounter	The downlink counter value for the received frame
uint32_t	DevAddress	End-device address
bool	DeviceTimeAnsReceived	Indicates if DeviceTimeAns MAC command is contained in the received frame [true: Received, false: Not received]
TimerTime_t	ResponseTimeout	[LoRaWAN 1.0.4 only] Response timeout in milliseconds for a class B/C when a confirmed downlink has been received.

2.3.6 MlmeReq_t

This type is structure containing MAC MLME-Request structure.

Table 2-25 MlmeReq_t

Member type	Member	Description	
Mlme_t	Type	MLME_JOIN	Initiates the OTAA activation
		MLME_LINK_CHECK	Initiates MAC command “LinkCheckReq” to validate connectivity
		MLME_DEVICE_TIME	Initiates MAC command “DeviceTimeReq” to request current GPS time
		MLME_BEACON_ACQUISITION	Initiates beacon acquisition
		MLME_PING_SLOT_INFO	Initiates MAC command “PingSlotInfoReq” to notify ping slot information
union sMlmeReq::uMlmeParam	Req	MLME-Request parameters, see the following	
RequestReturnParams_t	ReqReturn	Return parameter of MLME-Request.	
		TimerTime_t DutyCycleWaitTime	Report the time in milliseconds which an application must wait before it is possible to send the next uplink.

Note: uMlmeParam is a union containing MLME-Request parameters.

Table 2-26 uMlmeParam

Member type	Member	Description
MlmeReqJoin_t	Join	Join-Request parameters, when Type is MLME_JOIN, use this member.
MlmeReqPingSlotInfo_t	PingSlotInfo	PingSlotInfoReq parameter. Set if MLME type is MLME_PING_SLOT_INFO. See 2.3.18 MlmeReqPingSlotInfo_t for details.

2.3.7 MlmeReqJoin_t

This type is structure containing MAC MLME-Request for the join service.

Table 2-27 MlmeReqJoin_t

Member type	Member	Description
Uint8_t	Datarate	Data rate used for JoinRequest

2.3.8 MlmeConfirm_t

This type is structure containing MAC MLME-Confirm primitive.

Table 2-28 MlmeConfirm_t

Member type	Member	Description
Mlme_t	MlmeRequest	Holds the previously performed MLME-Request
LoRaMacEventInfoStatus_t	Status	Status of the operation
TimerTime_t	TxTimeOnAir	The transmission time on air of the frame
uint8_t	DemodMargin	Demodulation margin. Contains the link margin [dB] of the last successfully received LinkCheckReq
uint8_t	NbGateways	Number of gateways which received the last LinkCheckReq
uint8_t	NbRetries	Number of retransmissions for confirmed uplink frame

2.3.9 MlmeIndication_t

This type is structure containing MAC MLME-Indication primitive.

Table 2-29 MlmeIndication_t

Member type	Member	Description	
Mlme_t	MlmeIndication	MLME-Indication type	
		MLME_SCHEDULE_UPLINK	The application shall perform an uplink as soon as possible.
		MLME_BEACON	Indicates whether a beacon was successfully received
		MLME_BEACON_LOST	Lost synchronization with Class B beacons. This indicates MAC layer fails to receive a beacon for 120 minutes.
LoRaMacEventInfoStatus_t	Status	Status for MLME_BEACON. Indicates whether a beacon was successfully received.	
BeaconInfo_t	BeaconInfo	Beacon information for MLME_BEACON indication on successful beacon reception.	

2.3.10 MibRequestConfirm_t

This type is structure containing MAC parameters. This is used to get or to set parameters in MAC.

Table 2-30 MibRequestConfirm_t

Member type	Member	Description
Mib_t	Type	MIB-Request type. See 2.2.7 Mib_t.
MibParam_t	Param	MAC parameters for each Type. See 2.3.11 MibParam_t.

2.3.11 MibParam_t

This type is union containing MIB parameters. Each member type is a data structure for MIB listed in 2.2.7 Mib_t.

Table 2-31 MibParam_t

Member type	Member	Description
DeviceClass_t	Class	LoRaWAN device class for MIB_DEVICE_CLASS. See 2.2.1 DeviceClass_t. Note: The device class shall not be changed from A to B before the beacon acquisition is succeeded and the ping slot periodicity is set to the intended value.
		CLASS_A Device class A
		CLASS_B Device class B
		CLASS_C Device class C
ActivationType_t	NetworkActivation	Activation type. When operating in the OTAA mode, this parameter is automatically configured upon successful Join procedure and shall not be manually configured by LoRaMacMibSetRequestConfirm().
uint8_t*	DevEui	Pointer to device EUI, which is equivalent to DevEUI in LoRaWAN 1.0.x specification.
uint8_t*	JoinEui	Pointer to join EUI, which is equivalent to AppEUI in LoRaWAN 1.0.x specification.
bool	AdrEnable	Activation state of ADR for MIB_ADR.
		true ADR is enabled
		false ADR is disabled
uint32_t	NetID	Network identifier for MIB_NET_ID.
uint32_t	DevAddr	End-device address for MIB_DEV_ADDR
uint8_t*	GenAppKey	Pointer to application root key for multicast.
uint8_t*	AppKey	Application key for MIB_APP_KEY
uint8_t *	NwkSKey	Pointer to Network session key for MIB_NWK_SKEY
uint8_t *	AppSKey	Pointer to Application session key for MIB_APP_SKEY
Uint8_t *	McAppSKeyN (N = 0 - 3)	Pointer to Application session key for MIB_MC_APP_S_KEY_n. “N” means the index of the multicast group.
Uint8_t *	McNwkSKeyN (N = 0 - 3)	Pointer to Network session key for MIB_MC_NWK_S_KEY_n. “N” means the index of the multicast group.
bool	EnablePublicNetwork	Enable or disable a public network for MIB_PUBLIC_NETWORK
		true Public network is enabled
		false Public network is disabled
ChannelParams_t*	ChannelList	LoRaWAN channels parameter list. See 2.3.16 ChannelParams_t
RxChannelParams_t	Rx2Channel	Channel parameters for the receive window 2 for MIB_RX2_CHANNEL. See 2.3.13 RxChannelParams_t.
RxChannelParams_t	Rx2DefaultChannel	Default channel parameters for the receive window 2 for MIB_RX2_DEFAULT_CHANNEL. See 2.3.13 RxChannelParams_t.
uint8_t	ChannelsNbTrans	Maximum number of retransmissions for Unconfirmed uplink transmission
uint32_t	MaxRxWindow	Maximum receive window duration for MIB_MAX_RX_WINDOW_DURATION. [ms]

		[0: continuous, others: timeout]
uint32_t	ReceiveDelay1	Receive delay 1 for MIB_RECEIVE_DELAY_1. [ms]. 1000 ms or larger is recommended.
uint32_t	ReceiveDelay2	Receive delay 2 for MIB_RECEIVE_DELAY_2. [ms]. This must be ReceiveDelay1 + 1s.
uint32_t	JoinAcceptDelay1	Join accept delay 1 for MIB_JOIN_ACCEPT_DELAY_1 [ms]
uint32_t	JoinAcceptDelay2	Join accept delay 2 for MIB_JOIN_ACCEPT_DELAY_2 [ms]
Int8_t	ChannelsDatarate	Data rate of a channel for MIB_CHANNELS_DATARATE
int8_t	ChannelsDefaultTxPower	Default TX power for MIB_CHANNELS_DEFAULT_TX_POWER [Range: TX_POWER_0 .. TX_POWER_15]
int8_t	ChannelsTxPower	TX power for MIB_CHANNELS_TX_POWER [Range: TX_POWER_0 .. TX_POWER_15]
Version_t	AbpLrWanVersion	LoRaWAN version. Set when operating in the ABP mode.
bool	EnableCca	Configuration for MIB_AS923_ENABLE_CCA. Enable (true) or disable (false) CCA for AS923.
int8_t	PingSlotDatarate	Ping slot data rate
uint8_t	PingSlotPeriodicity	Ping slot periodicity [Range: 4 - 7] NOTE: Ping slot periodicity is set to the default value at the timing as follows: - LoRaMacMlmeRequest of MLME_JOIN is issued - Beacon acquisition fails - The device class is changed from class B to class A
bool	isCertPortOn	LoRaWAN certification FPort(=224) handling state. Enable (true) or disable (false) certification FPort.
uint16_t	devNonce	Device nonce (DevNonce)
uint32_t	appNonce	Application nonce (AppNonce) (or Join nonce (JoinNonce))
uint8_t	maxDcycle	Maximum duty cycle [Range: 0 - 15]
uint8_t	rx1DrOffset	RX1 data rate offset (RX1DRoffset) [Range: 0 - 15]
uint8_t	maxEirp	Maximum EIRP
uint8_t	downlinkDwellTime	Downlink dwell time [Range: 0, 1]
uint8_t	uplinkDwellTime	Uplink dwell time [Range: 0, 1]
uint32_t	downlinkFCnt	Downlink frame counter
uint32_t	uplinkFCnt	Uplink frame counter

2.3.12 LoRaMacTxInfo_t

This type is structure containing MAC Tx information.

Table 2-32 LoRaMacTxInfo_t

Member type	Member	Description
unt8_t	MaxPossibleApplicationDataSize	Maximum size of application data payload that can be sent in the next uplink transmission
uint8_t	CurrentPayloadSize	The current payload size, dependent on the current data rate

2.3.13 RxChannelParams_t

This type is structure containing MAC receive window channel parameters.

Table 2-33 RxChannelParams_t

Member type	Member	Description
uint32_t	Frequency	Frequency in Hz
uint8_t	Datarate	Data rate. See 2.1.1 Data rate.

2.3.14 McChannelSetup_t

This type is structure containing multicast channel parameters to set up.

Table 2-34 McChannelSetup_t

Member type	Member	Description
AddressIdentifier_t	GroupID	Multicast group ID [Range: 0 - 3]
uint32_t	Address	Multicast group address
uint8_t	*McKeyE	Pointer to encrypted multicast key which is delivered from application server.
uint32_t	FCountMin	Minimum count of multicast frame counter
uint32_t	FCountMax	Maximum count of multicast frame counter

2.3.15 McRxParams_t

This type is union containing multicast reception parameters.

Table 2-35 McRxParams_t

Member type	Member	Description
struct ClassB	uint32_t Frequency	Frequency in Hz
	int8_t DataRate	Data rate. See 2.1.1 Data rate.
	uint8_t Periodicity	Periodicity of multicast slots (4 to 7). Actual interval between multicast slots is $(0.96 * 2^{Periodicity})$ seconds.
struct ClassC	uint32_t Frequency	Frequency in Hz
	int8_t DataRate	Data rate. See 2.1.1 Data rate.

2.3.16 ChannelParams_t

This type is structure containing channel parameters.

Table 2-36 ChannelParams_t

Member type	Member	Description
uint32_t	Frequency	Frequency in Hz
uint32_t	Rx1Frequency	Alternative frequency for RX window in Hz
union DrRange_t	DrRange.Value	Byte access to the following bits of Data rate definition
	DrRange.Fields.Min : 4	Minimum data rate
	DrRange.Fields.Max : 4	Maximum data rate
uint8_t	Band	Band index

2.3.17 BeaconInfo_t

Class B beacon information structure.

Table 2-37 BeaconInfo_t

Member type	Member	Description
SysTime_t	Time	Elapsed time since January 6, 1980 00:00:00 UTC (start of the GPS epoch)
uint32_t	Frequency	Frequency in Hz
uint8_t	Datarate	Data rate
int16_t	Rssi	RSSI
int8_t	Snr	SNR
uint8_t	Param	[LoRaWAN 1.0.4 only] Precision of beacon's transmit time (0-3)
bool	isValidGwSpecific	Indicate whether GwSpecific (in below) is available
		true GwSpecific is available
		false GwSpecific is unavailable
struct sGwSpecific GwSpecific	InfoDesc	Information descriptor
	Info[6]	Information field

2.3.18 MlmeReqPingSlotInfo_t

Structure for PingSlotInfoReq configurations.

Table 2-38 MlmeReqPingSlotInfo_t

Member type	Member	Description
PingSlotInfo_t	PingSlot	Union data structure for PingSlotInfoReq

Table 2-39 PingSlotInfo_t

Member type	Member	Description
uint8_t	Value	Parameter for byte access
struct sInfoField Fields	Periodicity: 3	Periodicity of ping slots (4 to 7). Actual interval between ping slots is $(0.96 * 2^{Periodicity})$ seconds.
	RFU: 5	(Reserved)

2.4 LoRaWAN APIs

This section contains the following functions.

Table 2-40 LoRaWAN APIs

function	Description
LoRaMacInitialization	Initialize the MAC layer.
LoRaMacMlmeRequest	The Mac Layer Management Entity handles management service.
LoRaMacMcpsRequest	The Mac Common Part Sublayer handles data services
LoRaMacMibGetRequestConfirm	The Mac Information Base service to get attribute of the Mac layer.
LoRaMacMibSetRequestConfirm	The Mac Information Base service to set attribute of the Mac layer.
LoRaMacQueryTxPossible	Queries Mac if it is possible to send the next frame with given payload size.
LoRaMacProcess	Process the interruption.
LoRaMacChannelAdd	Add a new channel.
LoRaMacChannelRemove	Remove a channel.
LoRaMacStart	Start MAC.
LoRaMacStop	Stop MAC.
LoRaMacIsBusy	Check if MAC is busy.
LoRaMacMcChannelSetup	Configure multicast channel.
LoRaMacMcChannelDelete	Delete multicast channel.
LoRaMacMcChannelGetGroupId	Get multicast channel group ID.
LoRaMacMcChannelGetAddress	Get multicast address.
LoRaMacMcChannelSetupRxParams	Configure reception parameters of multicast channel.

2.4.1 LoRaMacInitialization

LoRaMacStatus_t LoRaMacInitialization(LoRaMacPrimitives_t *primitives, LoRaMacCallback_t * callbacks, LoRaMacRegion_t region)		
This function initializes MAC layer. Event handler functions to be set in 'primitive' are mandatory and user has to implement them. Callback function to be set 'events' is optional.		
Parameters		
[IN] primitives	Pointer to a structure defining the MAC event handler functions. Must set all handler functions. See section 2.5 in detail.	
[IN] callbacks	Pointer to a structure defining the MAC callback functions. Do not set NULL to this pointer while callback function pointers (structure members) may be set to NULL. See section 2.6 for details.	
[IN] region	The region to start. Following regions are supported in this version.	
	LORAMAC_REGION_EU868	Europe, band 868MHz
	LORAMAC_REGION_US915	North America, band 915MHz
	LORAMAC_REGION_AS923	Asia, band 923MHz, group AS923-1
	LORAMAC_REGION_AS923_2	[LoRaWAN 1.0.4 only] Asia, band 923MHz, group AS923-2
	LORAMAC_REGION_AS923_3	[LoRaWAN 1.0.4 only] Asia, band 923MHz, group AS923-3,
	LORAMAC_REGION_AS923_4	[LoRaWAN 1.0.4 only] Asia, band 923MHz, group AS923-4
	LORAMAC_REGION_AS923_JPN	Asia, band 923MHz, group AS923-1 for Japan In this case, the regulatory function for Japan ARIB STD-T108 is enabled in RadioDriver. Refer to [2]

Return	
LORAMAC_STATUS_OK	Initialization finished successfully.
LORAMAC_STATUS_PARAMETER_INVALID	A parameter set in primitives is invalid.
LORAMAC_STATUS_REGION_NOT_SUPPORTED	The region set in region is not supported
LORAMAC_STATUS_RADIO_FAIL	Radio driver initialization failure.

2.4.2 LoRaMacMibGetRequestConfirmtest

LoRaMacStatus_t LoRaMacMibGetRequestConfirm(MibRequestConfirm_t *mibRequest)		
This function is the MAC information base service to get attributes of the Mac layer. See the sample code how to use this function to get the parameter.		
Parameters		
mibRequest	[IN] Type	MAC attribute type to get. See 2.2.7 Mib_t
	[OUT] Param	Parameters got from MAC. See 2.3.11 MibParam_t

Return	
LORAMAC_STATUS_OK	The request is finished successfully
LORAMAC_STATUS_SERVICE_UNKNOWN	Requested attribute is unknown.
LORAMAC_STATUS_PARAMETER_INVALID	Requested parameter is invalid.

2.4.3 LoRaMacMibSetRequestConfirm

LoRaMacStatus_t LoRaMacMibSetRequestConfirm(MibRequestConfirm_t *mibSet)		
This function is the MAC information base service to set attributes of the MAC layer. Attributes cannot be changed when MAC service is running, from sending a Join-request or a data frame to closing a final Rx window. MIB parameters set by this function are initialized to their default values upon calling LoRaMacInitialization().		
Parameters		
mibSet	[IN] Type	MAC attribute type to get. See 2.2.7 Mib_t
	[IN] Param	Parameters got from MAC. See 2.3.11 MibParam_t

Return	
LORAMAC_STATUS_OK	The request is finished successfully
LORAMAC_STATUS_SERVICE_UNKNOWN	Requested attribute is unknown
LORAMAC_STATUS_PARAMETER_INVALID	Requested parameter is invalid
LORAMAC_STATUS_BUSY	MAC is busy. Another service is running

2.4.4 LoRaMacMlmeRequest (MAC command)

LoRaMacStatus_t LoRaMacMlmeRequest(MlmeReq_t *mlmeRequest)

This function requests MAC MLME-Request. The MAC layer management entity handles management services. See 5.4 and 5.6 for the sample code how to use this function. When the process of the transmission and the reception finish, the callback function MacMlmeConfirm() in the LoRaMacPrimitives_t will be called. Except for MLME_JOIN, please do not call this function before activation.

Parameters

mlmeRequest	[IN] Type	MLME_JOIN		Initiates the Over-the-Air activation (*1)	
		MLME_LINK_CHECK		Requests to send LinkCheckReq command to validate connectivity	
		MLME_DEVICE_TIME		Requests to send the DeviceTimeReq command to request current GPS time (*1) (*2)	
		MLME_BEACON_ACQUISITION		Initiates beacon acquisition (*1) (*3)	
		MLME_PING_SLOT_INFO		Requests to send the PingSlotInfoReq command to notify ping slot information (*1)	
	[IN] Req	Join	Datarate	Referenced data rate used for JoinRequest command. Actually, decided according to the region. [Range: DR_0 .. DR_15]	
				AS923	This parameter is ignored. DR2 is used.
				EU868	This parameter is applied.
				US915	This parameter is ignored. DR0 or DR4 is used according to Tx channel.
		PingSlotInfo	Field.Periodicity	Periodicity of ping slots [Range: 4 to 7] See 2.3.18 for details.	

Return

See 2.2.4 LoRaMacStatus_t	

*1) The following MLME shall not be issued when the LoRaWAN stack operates in class B.
- MLME_JOIN
- MLME_DEVICE_TIME
- MLME_BEACON_ACQUISITION
- MLME_PING_SLOT_INFO

*2) If GPS time is received via MLME_DEVICE_TIME, the LoRaWAN stack will open a receive window to acquire a beacon frame around the calculated beacon frame reception timing. If not, the LoRaWAN stack will open a receive window to acquire a beacon frame up to 128 seconds.

2.4.5 LoRaMacMcpsRequest (Data message)

LoRaMacStatus_t LoRaMacMcpsRequest(McpsReq_t *mcpsRequest)				
This function requests MAC MCPS-Request. The Mac Common Part Sublayer handles data services. See 5.5 for the sample code sending Mac frame with this function. When the process of the transmission and the reception finish, the callback function MacMcpsConfirm() in the LoRaMacPrimitives_t will be called.				
Parameters				
	mcpsRequest	[IN] Type	MCPS_UNCONFIRMED	Unconfirmed Data frame
			MCPS_CONFIRMED	Confirmed Data frame
		[IN] Req	Unconfirmed	Parameters for Unconfirmed frame. See 2.3.2 McpsReqUnconfirmed_t
			Confirmed	Parameters for Confirmed frame. See 2.3.3 McpsReqConfirmed_t
Return				
	See 2.2.4 LoRaMacStatus_t			

2.4.6 LoRaMacQueryTxPossible

LoRaMacStatus_t LoRaMacQueryTxPossible(uint8_t size, LoRaMacTxInfo_t *txInfo)			
This function queries the MAC if it is possible to send the next frame with a given payload size. The MAC takes scheduled MAC commands into account and reports, when the frame can be sent or not.			
Parameters			
	[IN] size	Size of applicative payload to be sent next.	
	[OUT] txInfo	MaxPossiblePayload	Size of the applicative payload which can be processed (according to the configured data rate or the next data rate according to ADR)
		CurrentPayloadSize	The current payload size, dependent on the current data rate
Return			
	LORAMAC_STATUS_OK		Finish this function successfully
	LORAMAC_STATUS_PARAMETER_INVALID		Invalid parameter, txInfo is NULL.
	LORAMAC_STATUS_LENGTH_ERROR		Payload length exceeds acceptable length to send.

2.4.7 LoRaMacProces

void LoRaMacProcess (void)
This function process events that the MAC may hold. Application must periodically call this function in its main loop at an interval as short as possible.
Parameters
-
Return
-

2.4.8 LoRaMacChannelAdd

LoRaMacStatus_t LoRaMacChannelAdd(uint8_t id, ChannelParams_t params)		
This function adds a new channel. It is available for AS923 and EU868. In case of US915, LORAMAC_STATUS_PARAMETER_INVALID will be returned.		
Parameters		
[IN] id		Channel ID
params	[IN] Frequency	Frequency in Hz
	[IN] Rx1Frequency	Alternative frequency for RX Window 1
	[IN] DrRange.Fields.Min	Minimum DR for this channel
	[IN] DrRange.Fileds.Max	Maximum DR for this channel
Return		
LORAMAC_STATUS_OK		Parameter was added successfully
LORAMAC_STATUS_BUSY		MAC is busy. Another service is running
LORAMAC_STATUS_PARAMETER_INVALID		Requested parameter is NULL
LORAMAC_STATUS_FREQUENCY_INVALID		Unacceptable frequency for the region
LORAMAC_STATUS_DATARATE_INVALID		Unacceptable data rate for the region
LORAMAC_STATUS_FREQ_AND_DR_INVALID		Unacceptable frequency and data rate for the region

2.4.9 LoRaMacChannelRemove

LoRaMacStatus_t LoRaMacChannelRemove(uint8_t id)	
Remove a channel. It is available for AS923 and EU868. In case of US915, LORAMAC_STATUS_PARAMETER_INVALID will be returned.	
Parameters	
[IN] id	ID of the channel to remove
Return	
LORAMAC_STATUS_OK	Parameter was added successfully
LORAMAC_STATUS_BUSY	MAC is busy. Another service is running.
LORAMAC_STATUS_PARAMETER_INVALID	Requested parameter is NULL

2.4.10 LoRaMacStart

LoRaMacStatus_t LoRaMacStart(void)	
Starts MAC	
Parameters	
-	
Return	
LORAMAC_STATUS_OK	MAC was started successfully

2.4.11 LoRaMacStop

LoRaMacStatus_t LoRaMacStop(void)	
Stops MAC	
Parameters	
-	
Return	
LORAMAC_STATUS_OK	MAC was stopped successfully

2.4.12 LoRaMacIsBusy

bool LoRaMacIsBusy(void)
Check if MAC is busy
Parameters
-
Return
true: MAC is in busy false: MAC is in idle (not busy)

2.4.13 LoRaMacMcChannelSetup

LoRaMacStatus_t LoRaMacMcChannelSetup(McChannelSetup_t *setup)	
Configure multicast channel. If “setup->McKeyE” is set (e.g.; remote setup), MIB_GEN_APP_KEY must be set before calling this function. If “setup->McKeyE” is NULL (e.g.; local setup), MIB_MC_APP_S_KEY_n and MIB_MC_NWK_S_KEY_n must be set before starting multicast communication.	
Parameters	
[IN]setup	**Multicast channel to configure.** See 2.3.14 McChannelSetup_t.
Return	
LORAMAC_STATUS_OK	Parameter was added successfully
LORAMAC_STATUS_BUSY	MAC is busy. Another service is running.
LORAMAC_STATUS_PARAMETER_INVALID	Requested parameter is NULL.
LORAMAC_STATUS_MC_GROUP_UNDEFINED	Multicast group is not defined.

2.4.14 LoRaMacMcChannelDelete

LoRaMacStatus_t LoRaMacMcChannelDelete(AddressIdentifier_t groupID)	
Delete multicast channel.	
Parameters	
[IN]groupID	Multicast channel ID to delete. [Range: 0 - 3]
Return	
LORAMAC_STATUS_OK	Parameter was deleted successfully
LORAMAC_STATUS_BUSY	MAC is busy. Another service is running.
LORAMAC_STATUS_MC_GROUP_UNDEFINED	Specified multicast channel ID is not defined.

2.4.15 LoRaMacMcChannelGetGroupId

uint8_t LoRaMacMcChannelGetGroupId(uint32_t mcAddress)		
Look up the multicast group ID for a multicast address.		
Parameters		
	[IN]mcAddress	Multicast address
Return		
	groupID	Multicast group ID associated to the multicast address. Returns 0xFF if the multicast address is not found.

2.4.16 LoRaMacMcChannelGetAddress

LoRaMacStatus_t LoRaMacMcChannelGetAddress(AddressIdentifier_t groupID, uint32_t *mcAddress)		
Get multicast group address associated with the groupID.		
Parameters		
	[IN]groupID	Multicast group ID. [Range: 0 - 3]
	[OUT]mcAddress	Multicast group address associated with the groupID
Return		
	LORAMAC_STATUS_OK	Parameter was added successfully
	LORAMAC_STATUS_PARAMETER_INVALID	Requested parameter is NULL.
	LORAMAC_STATUS_MC_GROUP_UNDEFINED	Multicast group is not defined.

2.4.17 LoRaMacMcChannelSetupRxParams

LoRaMacStatus_t LoRaMacMcChannelSetupRxParams(AddressIdentifier_t groupID, DeviceClass_t mcClass, McRxParams_t *rxParams, uint8_t *status)		
Configures reception parameters for a multicast channel. When preparing multiple channels for class C, the frequency and data rate must be the same.		
Parameters		
	[IN]groupID	Multicast channel ID to be configured. [Range: 0 - 3]
	[IN]mcClass	Multicast session class. CLASS_B or CLASS_C can be set.
	[IN]rxParams	Reception parameters to set. See 2.3.15 McRxParams_t
	[OUT]status	Status mask. bit 7-5: RFU (b'000) bit4 : McGroup Undefined if set bit3 : Freq Error if set bit2 : DR Error if set bit1-0 : Multicast group (0…3)
Return		
	LORAMAC_STATUS_OK	Parameter was added successfully
	LORAMAC_STATUS_BUSY	MAC is busy. Another service is running.
	LORAMAC_STATUS_PARAMETER_INVALID	Requested parameter is invalid or NULL.
	LORAMAC_STATUS_MC_GROUP_UNDEFINED	Specified multicast group is not defined.

2.5 LoRaWAN primitive callback handler (LoRaMacPrimitives_t)

This type is a structure containing MAC events handler functions used to notify upper layers MAC primitive events. This structure includes the following member functions. These members must be set before calling LoRaMacInitialization().

Table 2-41 LoRaMacPrimitives_t

Member	Description
void (*MacMcpsConfirm)(McpsConfirm_t *)	Notify LoRaMacMcpsRequest() processed.
void (*MacMcpsIndication)(McpsIndication_t *)	Notify the payload received.
void (*MacMlmeConfirm)(MlmeConfirm_t *)	Notify LoRaMacMlmeRequest() was accepted.
void (*MacMlmeIndication)(MlmeIndication_t *)	Notify the uplink message requested.

2.5.1 MacMcpsConfirm

void (*MacMcpsConfirm)(McpsConfirm_t *mcpsConfirm)			
This function notify that LoRaMacMcpsRequest() processed.			
Parameters			
mcpsConfirm	[IN] McpsRequest	Requested MAC data service.	
		MCPS_UNCONFIRMED	Unconfirmed Data frame
		MCPS_CONFIRMED	Confirmed Data frame
	[IN] Status	Status of the operation of a MAC service. See 2.2.3 LoRaMacEventInfoStatus_t.	
	[IN] Datarate	Uplink data rate, DR_0.. DR_15. See 2.1.1 Data rate.	
	[IN] TxPower	Transmission Power, TX_POWER_0..TX_POWER_15. See 2.1.2 Transmission power	
	[IN] AckReceived	Received ack or not, true: received, false: not receive	
	[IN] NbRetries	Number of retransmissions	
	[IN] TxTimeOnAir	The transmission time on air of the frame in ms.	
	[IN] UpLinkCounter	The uplink counter value related to the frame	
	[IN] Channel	The uplink channel related to the frame.	
Return			
-			

2.5.2 MacMlmeConfirm

void (*MacMlmeConfirm)(MlmeConfirm_t *mlmeConfirm)				
This function notifies LoRaMacMlmeRequest() was completed				
Parameters				
	mlmeConfirm	[IN] MlmeRequest	MLME_JOIN	Completed OTAA process
			MLME_LINK_CHECK	Completed to request to send MAC command “LinkCheckReq”
			MLME_DEVICE_TIME	Completed to request to send MAC command “DeviceTimeReq”
			MLME_BEACON _ACQUISITION	Completed beacon acquisition
			MLME_PING_SLOT _INFO	Completed to requested to send MAC command “PingSlotInfoReq”
		[IN] Status	Status of the operation of a MAC service. See 2.2.3 LoRaMacEventInfoStatus_t.	
		[IN] TxTimeOnAir	The transmission time on air of the frame. [ms]	
		[IN] DemodMargin	Demodulation margin. Contains the link margin [dB] of LinkCheckReq last successfully received by the network.	
			0	0 dB or no margin
			1..254	1 dB .. 254 dB
			255	Reserved
		[IN] NbGateways	Number of gateways which received the last LinkCheckReq	
		[IN] NbRetries	Number of retransmissions for confirmed uplink transmission.	
Return				
	-			

2.5.3 MacMcpsIndication

void (*MacMcpsIndication)(McpsIndication_t *mcpsIndication)			
This function notifies the received payload			
Parameters			
mcpsIndication	This function notifies the received payload		
	[IN] McpsIndication	MCPS_UNCONFIRMED	Unconfirmed data frame
		MCPS_CONFIRMED	Confirmed data frame
	[IN] Status	Status of the operation of a MAC service. See 2.2.3 LoRaMacEventInfoStatus_t.	
	[IN] Port	The port that received message sent on.	
		0	MAC commands
		1..223	Application specific port
		224	Mac layer test protocol
		225..255	Reserved for standardized application extensions
	[IN] RxDatarate	Downlink data rate. DR_0..DR_15. See 2.1.1 Data rate.	
	[IN] FramePending	Indicates gateway has more data pending to be sent [0: No pending frame, 1: Pending frame exists]	
	[IN] Buffer	Pointer to the received data stream.	
	[IN] BufferSize	Size of the received data stream. 0..255 [bytes]	
	[IN] RxData	Indicates if data is available. [true: Available, false: Unavailable]	
	[IN] Rssi	RSSI of the received packet. [dBm]	
	[IN] Snr	SNR of the received packet [dBm]	
	[IN] RxSlot	Receive window for the received message. See 2.2.8 LoRaMacRxSlot_t	
	[IN] AckReceived	Indicates if an ack was received. [true: Received, false: Not received]	
	[IN] DownlinkCounter	The downlink counter value for the received frame	
	[IN] DevAddress	Device address (DevAddr)	
	[IN] DeviceTimeAns Received	Indicates if the “DeviceTimeAns” command is received. [true: Received, false: Not received]	
Return			
-			

2.5.4 MacMlmeIndication

void (*MacMlmeIndication)(MlmeIndication_t *mlmeIndication)			
This function notifies MAC MLME-Indication primitive			
Parameters			
mlmeIndication	[IN] MlmeIndication	MLME_SCHEDULE_UPLINK	The application shall perform an uplink as soon as possible. ** Don't send an uplink in this indication. Application has to send uplink message in its main loop.
		MLME_BEACON	Indicates whether a beacon was successfully received
		MLME_BEACON_LOST	Lost synchronization with Class B beacons. This indicates MAC layer fails to receive a beacon for 120 minutes.
	[IN] Status	Operation status in case of MLME_BEACON	
		LORAMAC_EVENT_INFO_STATUS_BEACON_LOCKED	Successfully received a beacon frame
		LORAMAC_EVENT_INFO_BEACON_LOST	Could not receive a beacon frame
	[IN] BeaconInfo	Beacon information	
Return			
-			

2.6 LoRaWAN callback handler (LoRaMacCallback_t)

This type is a structure containing MAC events handler functions used to notify upper layers. This structure includes following member functions. Application has to set pointers to user functions or NULL into the structure.

Table 2-42 LoRaMacCallback_t

Member	Description
uint8_t (*GetBatteryLevel)(void)	Pointer to callback function to be called when measured battery level is requested by MAC
void (*NvmContextChange)(uint32_t notifyMibFlags)	Pointer to callback function to be called when an attribute in one of the non-volatile contexts changed.
void (*MacProcssNotify)(void)	Pointer to callback function to be called when an interrupt request by the radio driver is processed.
void (*MacErrorNotify) (LoRaMacErrorNotificationStatus_t status)	Pinter to callback function to be called when an error is notified to application
float (*GetTemperatureLevel)(void)	Reserved. Set NULL.

2.6.1 GetBatteryLevel

uint8_t (*GetBatteryLevel)(void)	
This function will be called on reception of DevStatusReq command by MAC. Application has to measure the battery level in this function and return it. When the application assign NULL as this function, LoRaWAN stack return BAT_LEVEL_NO_MEASURE to the network.	
Parameters	
-	
Return	
BAT_LEVEL_EXT_SRC	Node is connected to an external power source
BAT_LEVEL_EMPTY	Battery level empty
2..253	Battery level, where 2 is the minimum
BAT_LEVEL_FULL	Battery level full
BAT_LEVEL_NO_MEASURE	The node was not able to measure the battery level

2.6.2 NvmContextChange

void (*NvmContextChange)(uint32_t notifyMibFlags)	
Called when the attribute(s) in the non-volatile context changes.	
Parameters	
notifyMibFlags	Changed attribute(s). One or more of the following parameters will be set. ● LORAMAC_NVM_MIBFLG_DEV_NONCE (0x00000001) DevNonce is changed. ● LORAMAC_NVM_MIBFLG_APP_NONCE (0x00000002) AppNonce (or JoinNonce) is changed. ● LORAMAC_NVM_MIBFLG_DOWNLINK_FCNT (0x00000004) Downlink frame counter is changed. ● LORAMAC_NVM_MIBFLG_UPLINK_FCNT (0x00000008) Uplink frame counter is changed. ● LORAMAC_NVM_MIBFLG_CHANNELS (0x00000010) Channel (frequency and data rate) information is changed. ● LORAMAC_NVM_MIBFLG_CHANNELS_MASK (0x00000020) Channel mask is changed. ● LORAMAC_NVM_MIBFLG_CHANNELS_DATARATE (0x00000040) Data rate is changed by ADR. ● LORAMAC_NVM_MIBFLG_CHANNELS_NB_TRANS (0x00000080) NbTrans is changed. ● LORAMAC_NVM_MIBFLG_CHANNELS_TXPOWER (0x00000100) TxPower is changed. ● LORAMAC_NVM_MIBFLG_MAX_DCYCLE (0x00000200) Maximum duty cycle is changed. ● LORAMAC_NVM_MIBFLG_RX1_DROFFSET (0x00000400) RX1DRoffset is changed. ● LORAMAC_NVM_MIBFLG_RX2_FREQUENCY (0x00000800) RX2 frequency is changed. ● LORAMAC_NVM_MIBFLG_RX2_DATARATE (0x00001000) RX2 data rate is changed. ● LORAMAC_NVM_MIBFLG_RECEIVE_DELAY_1 (0x00002000) RX1 received delay is changed. ● LORAMAC_NVM_MIBFLG_MAX_EIRP (0x00004000) Maximum EIRP is changed. ● LORAMAC_NVM_MIBFLG_DOWNLINK_DWELLTIME (0x00008000) Downlink dwell time is changed. ● LORAMAC_NVM_MIBFLG_UPLINK_DWELLTIME (0x00010000) Uplink dwell time is changed. ● LORAMAC_NVM_MIBFLG_PING_SLOT_DATARATE (0x00020000) PingSlot data rate is changed.
Return	
-	

2.6.3 MacProcessNotify

void (*MacProcessNotify)(void)	
Called when an interrupt request by the radio driver is processed. After this callback function is called, execute LoRaMacProcess(). Note that LoRaMacProcess() may not be called in this callback function.	
Parameters	
-	
Return	
-	

2.6.4 MacErrorNotify

void (*MacErrorNotify)(LoRaMacErrorNotificationStatus_t status)	
Called when an error is notified to applications	
Parameters	
status	Error status. This parameter can take of the following value. ● LORAMAC_ERROR_NOTIFICATION_STATUS_RADIO_CHECK_FAIL_RX_CFG Radio RX parameter configuration failure
Return	
-	

3. Timer

Timer provides timer event and a system time value.
For more details, please refer to [1].

4. Power saving

4.1 LoRaMacSetLowPower

LoRaMacStatus_t LoRaMacSetLowPower(void)	
This function sets MCU to the low power mode. Application can call this function when it can be in the low power mode. In this function, MCU will be in the low power mode if LoRaWAN is not busy. This function is not available for LoRaWAN Class C devices.	
Parameters	
-	
Return	
LORAMAC_STATUS_OK	MCU could be set to the low power mode and returned to the normal mode successfully.
LORAMAC_STATUS_BUSY	MCU cannot be set to the low power mode because of; - LoRaWAN is busy. - Application disallows MCU to be set to low power mode. (See below) - Device is in LoRaWAN Class C mode.
Additional explanation	
Application can allow/disallow MCU low power by using BoardIsLowPowerAllowed() function. It is called before MCU will be in the low power mode. [board.c] bool BoardIsLowPowerAllowed(void) Please make BoardIsLowPowerAllowed return true if MCU can be set to the low power mode, false if not.	

5. Sample codes (Informative)

Initialize the LoRaWAN MAC
The following code-snippet shows how to use the API to initialize the MAC.
#include "LoRaMac.h"

```
#include "Region.h"

static void McpsConfirm(McpsConfirm_t *);
static void McpsIndication(McpsIndication_t *);
static void MlmeConfirm(MlmeConfirm_t *);
static void MlmeIndication(MlmeIndication_t *);

static void AppNvmContextChange( uint32_t notifyMibFlags );

LoRaMacPrimitives_t LoRaMacPrimitives;
LoRaMacCallback_t LoRaMacCallbacks;

/* set mac primitives       */
LoRaMacPrimitives.MacMcpsConfirm = McpsConfirm;
LoRaMacPrimitives.MacMcpsIndication = McpsIndication;
LoRaMacPrimitives.MacMlmeConfirm = MlmeConfirm;
LoRaMacPrimitives.MacMlmeIndication = MlmeIndication;

/* set the getter of board information       */
LoRaMacCallbacks.GetBatteryLebel = BoardGetBatteryLevel;     /* maybe implemented for each board     */
LoRaMacCallbacks.GetTemperatureLevel = NULL;
LoRaMacCallbacks.NvmContextChange =  AppNvmContextChange;
LoRaMacCallbacks.MacProcessNotify = NULL;
LoRaMacCallbacks.MacErrorNotify = OnMacErrorNotify;

/* initialize MAC */
LoRaMacInitialization(&LoRaMacPrimitives, &LoRaMacCallbacks, LORAMAC_REGION_AS923);
```

5.1 Set a MIB parameter

The following code-snippet shows how to use the API to set the parameter DevAddr.
MibRequestConfirm_t mibReq;

```
LoRaMacStatus_t    status;

mibReq.Type = MIB_DEV_ADDR;
mibReq.Param.DevAddr = appLorawanSettings.devAddr;
status = LoRaMacMibSetRequestConfirm( &mibReq );
```

5.2 Get a MIB parameter

The following code-snippet shows how to use the API to get the parameter DevAddr.
MibRequestConfirm_t mibGet;

```
LoRaMacStatus_t    status;

mibGet.Type = MIB_DEV_ADDR;
stat = LoRaMacMibGetRequestConfirm(&mibGet);
```

5.3 Activation by Personalization (ABP)

The following code-snippet shows how to use the API to activate by personalization.
static uint8_t NwkSKey[] = { 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0x0f };

```
static uint8_t AppSKey[] = { 0x2B, 0x7E, 0x15, 0x16, 0x28, 0xAE, 0xD2, 0xA6, 0xAB, 0xF7, 0x15, 0x88, 0x09, 0xCF, 0x4F, 0x3C };
static uint32_t DevAddr = ( uint32_t )0x01020304;
MibRequestConfirm_t    mibReq;

mibReq.Type = MIB_ABP_LORAWAN_VERSION;
mibReq.Param.AbpLrWanVersion = 0x01000300;          // 0x01000300 = LoRaWAN 1.0.3
LoRaMacMibSetRequestConfirm( &mibReq );

mibReq.Type = MIB_NET_ID;
mibReq.Param.NetID = 0;
LoRaMacMibSetRequestConfirm( &mibReq );

mibReq.Type = MIB_DEV_ADDR;
mibReq.Param.DevAddr = DevAddr;
LoRaMacMibSetRequestConfirm( &mibReq );

mibReq.Type = MIB_NWK_SKEY;
mibReq.Param.NwkSKey = NwkSKey;
LoRaMacMibSetRequestConfirm( &mibReq );

mibReq.Type = MIB_APP_SKEY;
mibReq.Param.AppSKey = AppSKey;
LoRaMacMibSetRequestConfirm( &mibReq );

mibReq.Type = MIB_NETWORK_ACTIVATION;
mibReq.param.NetworkActivation = ACTIVATION_TYPE_ABP;
LoRaMacMibSetRequestConfirm( &mibReq );
```

5.4 Perform Join-Request

The following code-snippet shows how to use the API to perform a network join request.

```
static uint8_t    DevEui[] = {  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x09, 0x0F };
static uint8_t    AppEui[] = {  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0A, 0x0B };
static uint8_t    AppKey[] = {  0x2B, 0x7E, 0x15, 0x16, 0x28, 0xAE, 0xD2, 0xA6, 0xAB, 0xF7, 0x15, 0x88,
0x09, 0xCF, 0x4F, 0x3C };
MibRequestConfirm_t    mibReq;
MlmeReq_t     mlmeReq;

mibReq.Type = MIB_DEV_EUI;
mibReq.Param.DevEui = DevEUI;
LoRaMacMibSetRequestConfirm( &mibReq );

mibReq.Type = MIB_APP_EUI;
mibReq.Param.AppEui = AppEUI;
LoRaMacMibSetRequestConfirm( &mibReq );

mibReq.Type = MIB_APP_KEY;
mibReq.Param.AppKey = AppKey;
LoRaMacMibSetRequestConfirm( &mibReq );

mlmeReq.Type = MLME_JOIN;
mlmeReq.Req.Join.Datarate = DR_2;     // Ignored in case of AS923
if( LoRaMacMlmeRequest( &mlmeReq ) == LORAMAC_STATUS_OK )
{
  // Service started successfully. Waiting for the Mlme-Confirm event
}
```

5.5 Send unconfirmed data frame

The following code-snippet shows how to use the API to send an unconfirmed data frame.

```
uint8_t    myBuffer[] = { 1, 2, 3 };
McpsReq_t    mcpsReq;

mcpsReq.Type = MCPS_UNCONFIRMED;
mcpsReq.Req.Unconfirmed.fPort = 1;
mcpsReq.Req.Unconfirmed.fBuffer = myBuffer;
mcpsReq.Req.Unconfirmed.fBufferSize = sizeof( myBuffer );

if( LoRaMacMcpsRequest( &mcpsReq ) == LORAMAC_STATUS_OK )
{
  // Service started successfully. Waiting for the MCPS-Confirm event
}
```

5.6 Switching to Class B

The following code-snippet shows an example API sequence to switch an end device from Class A to Class B. Please make sure you have the same ping slot periodicity configuration on the device and server when you set the ping slot periodicity without publishing the MAC command PingSlotInfoReq.

```
MlmeReq_t    mlmeReq;
MibRequestConfirm_t    mibReq;

mlmeReq.Type = MLME_DEVICE_TIME;
if( LoRaMacMlmeRequest( &mlmeReq ) == LORAMAC_STATUS_OK )    // Request DeviceTimeAns
{
// Service started successfully. Waiting for the Mlme-Confirm event
}

mlmeReq.Type = MLME_BEACON_ACQUISITION
if( LoRaMacMlmeRequest( &mlmeReq ) == LORAMAC_STATUS_OK )    // Initiate beacon acquisition
{
 // Service started successfully. Waiting for the Mlme-Confirm event
}

mibReq.Type = MIB_PING_SLOT_PERIODICITY;
mibReq.Param.PingSlotPeriodicity = 5;
LoRaMacMibSetRequestConfirm( &mibReq );     // Set the ping slot periodicity to 5

mibReq.Type = MIB_DEVICE_CLASS;
mibReq.Param.Class = CLASS_B;
LoRaMacMibSetRequestConfirm(&mibReq);       // Switch the device class to Class B

// Device starts Class B operation, opening ping slots at the periodicity of 5.
// Make sure you have the same ping slot periodicity configuration on the device and server when you
// set the ping slot periodicity without publishing the MAC command PingSlotInfoReq.

// Notifies LoRaWAN server that currently it is operating in Class B
// LinkCheckReq command which requests response is used for example.

mlmeReq.Type = MLME_LINK_CHECK;
if( LoRaMacMlmeRequest( &mlmeReq ) == LORAMAC_STATUS_OK )    // Request LinkCheckAns
{
 // Service started successfully. Waiting for the Mlme-Confirm event
}
```

5.7 Timer

The following code-snippet shows how to use asynchronous timer.

```
static TimerEvent_t AsyncTimer;          /* timer event object      */
static OnAsyncTimerEvent(void);          /* timer event handler     */

int main(void)
{
        /* initialize Timer object  */
        TimerInit(&AsyncTimer, OnAsyncTimerEvent);
        TimerSetValue(&AsyncTimer, 25);                    /* set timeout value to 25ms       */

        /* start timer      */
        TimerStart(&AsyncTimer);
}
```

5.8 Timer time

The following code-snippet shows an example to use timer value functions.

```
/* example to repeat some process for 10ms      */
TimerTime_t startTime = TimerGetCurrentTime();

while (TimerGetElapsedTime(startTime) < 10) {
        /* code to repeat           */
}
```

Revision History

Rev.	Date	Description	
		Section	Summary
01.00	Jan. 31, 2019	-	Initial Release
01.10	Nov. 29, 2019	-	Changed target device from 'RL78/G14 (R5F104JJ)' to 'RL78/G14 (R5F104ML)'
		1.3	Changed CSI21 to CSI31 Deleted UART1 because UART1 is not necessary for the stack.
		2.2.5, 2.3.1, 2.4.4, 2.5.1, 2.5.2	Changed proprietary frame transmission and reception as 'reserved' (not supported)
		2.2.7	Added MIB_APP_KEY
		6	Remove this section due to the information in the section is for older version of the stack
02.10	July. 10, 2020	-	Supported for Class B and Multicast functions. Added related APIs, IBs, structures, enumerators and macros.
		2.1.4	Added LORAMAC_CLASSB_ENABLED to select whether to use Class B functions
		2.3.1	Added RequestReturnParams_t to McpsReq_t to report duty cycle wait time
		2.3.6	Added RequestReturnParams_t to MlmeReq_t to report duty cycle wait time
		2.4	Added LoRaMacStart and LoRaMacStop
		2.6	Added NvmContextChange, MacProcessNotify, MacErrorNotify
		3	Deleted description for Timer. Added reference document for Timer.
		5.7	Added sample code for Class B
03.00	Mar. 26, 2021	-	Supported RL78/G23 (R7F100GLG) as a target device
03.10	Sep 30, 2021	-	Supported for LoRaWAN protocol specified in the specification version 1.0.4.
		-	Supported RL78/G23 (R7F100GSN) as a target device
		1.4	Added region definition in Table 1-5; Group AS923-1,2,3,4, and AS923-Japan
		2.1.4	- Added LORAWAN_VERSION_1_0_3 and LORAWAN_VERSION_1_0_4 - Added description of REGION_AS923
		2.2	Deleted LoRaMacNvmCtxModule_t
		2.2.2	Added the enumerators; LORAMAC_REGION_AS923_2 LORAMAC_REGION_AS923_3 LORAMAC_REGION_AS923_4 LORAMAC_REGION_AS923_JPN
		2.2.3	Added the enumerator; LORAMAC_EVENT_INFO_STATUS_JOIN_NONCE_FAIL

		2.2.7 2.3.11	Added MIB (2.2.7) and the members of MibParam_t (2.3.11); MIB_CHANNELS, MIB_IS_CERT_FPORT_ON, MIB_DEV_NONCE, MIB_APP_NONCE, MIB_MAX_DCYCLE, MIB_RX1_DROFFSET, MIB_MAX_EIRP, MIB_DOWNLINK_DWELLTIME, MIB_UPLINK_DWELLTIME, MIB_DOWNLINK_FCNT, MIB_UPLINK_FCNT
		2.4.1	Added the enumerators which can be set in region parameter.
		2.6 2.6.2	Changed the argument of NvmContextChange() callback function.
03.12	Jan. 21, 2022	Table 1-2	Removed ‘Real Time Clock (RTC)’ for correction.
		Table 1-3	Removed ‘Real Time Clock (RTC)’ for correction.
		Table 2-2	Modified RFU to value of TX_POWER in case of US915, TX_POWER_10 to TX_POWER_14 for correction.
		Table 2-4	Added ‘AS923-1 for Japan’ for REGION_AS923. Added ‘LoRaWAN Regional Parameters RP002-1.0.3’ for LORAWAN_VERSION_1_0_4. Added ‘LoRaWAN 1.0.3 Regional Parameters Revision A’ for LORAWAN_VERSION_1_0_3.

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)

   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on

   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state

   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins

   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.
5. Clock signals

   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.
6. Voltage application waveform at input pin

   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses

   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products

   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
6. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.4.0-1 November 2017)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.

Trademarks

Semtech, the Semtech logo, LoRa, LoRaWAN and LoRa Alliance are registered trademarks or service marks, or trademarks or service marks, of Semtech Corporation and/or its affiliates.

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.