Private LoRa® Stack

Reference Guide

Introduction

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

Target Device

MCU: Renesas RL78/G23 (R7F100GSN, R7F100GLG), RL78/G14 (R5F104ML), RL78/G22 (R7F102GGE),
RA2E1 (R7FA2E1A9xxFM), RA2L1 (R7FA2L1AB2DFP),
RA0E1 (R7FA0E1073CFJ) or RA0E2 (R7FA0E2094CFM)
Transceiver: Semtech SX1261 or SX1262

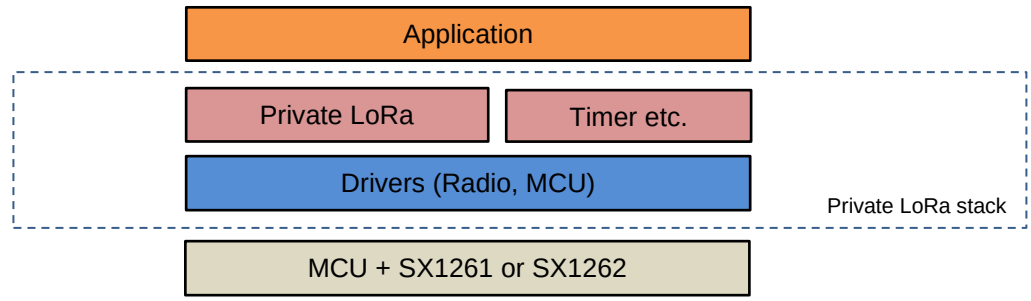
Contents

1. Overview

This application note contains API references and other information to use the Private LoRa stack.
The Private LoRa interfaces are described in chapter 2. The Timer interface is described in chapter 3.
Application can also use Timer APIs, although they are used in the stack.

1.1 Private LoRa Stack Block Diagram

Figure 1 shows a block diagram of the Private LoRa Stack.



Note: This document covers Private LoRa MAC and Timer.

Figure 1. Private LoRa Stack Block Diagram

1.2 Directories (informative)

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

Table 1. Directories

Directories	Description
src/apps	Application code.
src/boards	Board specific codes.
src/boards/mcu	MCU drivers (except RL78/G23 and RL78/G22).
src/prvloramac	Private LoRa stack.
src/radio	Radio driver for LoRa®.
src/peripherals	Security related codes.
src/system	Utility APIs, and so forth.
<ProjectDir>/src/smc_gen	MCU drivers for RL78/G23 and RL78/G22 generated by RL78 Smart Configurator. * <ProjectDir> is a folder for e2studio/CS+ project.

1.3 Resource Usage Example

Please refer to [3] for RL78 and [4] for RA in the following folder for the resource usage such as memory and peripherals.

Folder: `(package top)\documents\`

1.4 Acronyms and Abbreviations

Table 2. Acronyms and Abbreviations

Acronyms	Description
MCPS	MAC Common Part Sublayer
MLME	MAC Layer Management Entity
PSK	Pre-Shared Key
RFU	Reserved for Future Use
DR	Data rate
SF	Spreading factor
BW	Modulation bandwidth
AS1	Composed of Asia countries having available frequencies in the 915-928 MHz range
AS2	Composed of Asia countries having available frequencies in the 920-923 MHz range
AS3	Composed of Asia countries having available frequencies in the 915-921 MHz range
AS4	Composed of Asia countries having available frequencies in the 917-920 MHz range
JP	Perform ARIB STD-T108 regulation for using in Japan (without low duty cycle method)
JP-LDC	Perform ARIB STD-T108 regulation for using in Japan (with low duty cycle method)
EU	EU863-870 MHz Band
US	US902-928 MHz ISM Band
IN	IN865-867 MHz Band
AU	AU915-928 MHz Band
KR	KR920-923 MHz Band

1.5 Related Documentation

	Document No.	Title	Author	Language
[1]	R11AN0227	Radio Driver Reference Guide	Renesas Electronics	English
[2]	R11AN0834	Radio Driver Support Functions for Regional Radio Regulations	Renesas Electronics	English
[3]	R11AN0595	RL78/G23, RL78/G22, RL78/G14 LoRa®-based Wireless Software Package	Renesas Electronics	English
[4]	R11AN0596	RA2E1, RA2L1, RA0E1, RA0E2 LoRa®-based Wireless Software Package	Renesas Electronics	English
[5]	R11AN0937	Smart Configurator Usage for RL78 LoRa®-based Wireless Software Reference Guide	Renesas Electronics	English

2. Private LoRa Interface

This section describes the Private LoRa stack interfaces.

2.1 Macros

This section includes the following enumeration types.

2.1.1 Stack Settings

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

Table 3. Macros for Stack Setting

Macro	Description
PRIVATELORA_ENABLED	Enable Private LoRa feature.
RADIO_CFG_AS_ENABLED	Enable AS1, AS2, AS3, AS4, JP and JP-LDC feature.
RADIO_CFG_EU_ENABLED	Enable EU feature.
RADIO_CFG_US_ENABLED	Enable US feature.
RADIO_CFG_IN_ENABLED	Enable IN feature.
RADIO_CFG_AU_ENABLED	Enable AU feature.
RADIO_CFG_KR_ENABLED	Enable KR feature.
RP_USE_RADIO_CFG_CHECK	Enable the regulatory function for each region in Radio Driver. Refer to [2].

2.1.2 Configuration

Parameters available for configuration in the Private LoRa stack are defined in `PrivateLoRaConfig.h`.

Table 4. Macros for Stack Configuration

Macro	Description	
PRVLORA_CONFIG_REMOTE_DEVICE_MAXNUM	**Type:** uint8_t (1 – 3)	Default: 1
	Maximum number of remote devices. Note: the maximum value for RL78/G22 is 1.	
PRVLORA_CONFIG_INDIRECT_TX_QUEUE_MAXNUM	**Type:** uint8_t (1 – 3)	Default: 1
	Maximum number of the indirect transmission queue. Note: the maximum value for RL78/G22 is 1. Note: the maximum value depends on RAM.	
PRVLORA_CONFIG_SYSTEM_MAX_RX_ERROR	**Type:** uint32_t	Default: (RL78) 35 (RA2) 35 (RA0E1) 35 (RA0E2) 35
	System overall timing error in msec.	
PRVLORA_CONFIG_TRXADJUST_TX2RX	**Type:** uint32_t	Default: (RL78) 7 (RA2) 10 (RA0E1) 7 (RA0E2) 8
	Processing time from completion of transmission to start of reception. It is used to fine-tune the start timing of reception.	
PRVLORA_CONFIG_TRXADJUST_RX2TXRES_FIXED	**Type:** uint32_t	Default: (RL78) 9 (RA2) 11 (RA0E1) 8 (RA0E2) 9
	Processing time from completion of reception to just before starting response transmission. It is used to fine-tune the start timing of transmission.	

2.2 Enumerations

This section includes the following enumeration types.

Table 5. Enumerations

Types	Description
PrvLoRaRegion_t	Region and band.
PrvLoRaStatus_t	The status of the requested Private LoRa services.
PrvLoRaEventInfoStatus_t	The status of the events of the Private LoRa services.
PrvLoRaIb_t	Private LoRa Information Base (IB) type
PrvLoRaMlme_t	Private LoRa management service type.
PrvLoRaNotifyId_t	Private LoRa notification type.

2.2.1 PrvLoRaRegion_t

This enumeration type contains the following MAC region and frequency bands.

Table 6. PrvLoRaRegion_t

Enumerator	Description
PRVLORA_REGION_EU	Europe, band 868 MHz
PRVLORA_REGION_IN	India, band 865 MHz
PRVLORA_REGION_AS1	Asia, band 915-928 MHz
PRVLORA_REGION_AS2	Asia, band 920-923 MHz
PRVLORA_REGION_AS3	Asia, band 915-921 MHz
PRVLORA_REGION_AS4	Asia, band 917-920 MHz
PRVLORA_REGION_US	North America, band 915 MHz
PRVLORA_REGION_AU	Australia, band 915 MHz
PRVLORA_REGION_KR	South Korea, band 920 MHz
PRVLORA_REGION_JP	Japan, ARIB STD-T108 regulation (without low duty cycle method)
PRVLORA_REGION_JP_LDC	Japan, ARIB STD-T108 regulation (with low duty cycle method)

2.2.2 PrvLoRaStatus_t

This enumeration type contains Private LoRa status and indicates the result of requested Private LoRa service as follows.

Table 7. PrvLoRaStatus_t

Enumerator	Description
PRVLORA_STATUS_OK	Service started successfully.
PRVLORA_STATUS_ERROR	Error – Undefined error.
PRVLORA_STATUS_BUSY	Error – Processing request.
PRVLORA_STATUS_INACTIVE	Error – Private LoRa is inactive.
PRVLORA_STATUS_PARAMETER_INVALID	Error – Invalid parameter.
PRVLORA_STATUS_REQUSET_INVALID	Error – Invalid request.
PRVLORA_STATUS_NO_ REMOTE_DEVICE_ENTRY	Error – Remote device entry is not found.
PRVLORA_STATUS_NOT_SUPPORTED	Error – Request is not supported.
PRVLORA_STATUS_SERVICE_UNKNOWN	Error – Unknown request.
PRVLORA_STATUS_IB_ATTRIBUTE_INVALID	Error – Invalid access to specified IB.
PRVLORA_STATUS_LENGTH_ERROR	Error – Payload length is too long.
PRVLORA_STATUS_COMMAND_ERROR	Error – Command error.
PRVLORA_STATUS_INSUFFICIENT_MEMORY	Error – Insufficient memory.
PRVLORA_STATUS_DATARATE_INVALID	Error – Invalid data rate.

PRVLORA_STATUS_CHANNEL_INVALID	Error – Invalid channel.
PRVLORA_STATUS_RADIO_ERROR	Error – Radio driver initialization failure.
PRVLORA_STATUS_RADIO_CHANNEL_BUSY	Error – Channel is busy.
PRVLORA_STATUS_RADIO_DUTYCYCLE_RESTRICTED	Error – Transmission was aborted due to duty cycle restriction.
PRVLORA_STATUS_RADIO_PARAMETER_INVALID	Error – Radio parameter configuration is invalid.

2.2.3 PrvLoRaEventInfoStatus_t

This enumeration type contains the status of the operation of a Private LoRa service as follows.

Table 8. PrvLoRaEventInfoStatus_t

Enumerator	Description
PRVLORA_EVENTINFO_STATUS_OK	Service performed successfully.
PRVLORA_EVENTINFO_STATUS_ERROR	An error occurred during the execution of the service.
PRVLORA_EVENTINFO_STATUS_TX_TIMEOUT	A Tx timeout occurred.
PRVLORA_EVENTINFO_STATUS_TX_NOACK	ACK frame from remote device cannot be received.
PRVLORA_EVENTINFO_STATUS_TX_CANCELED	Tx request has been canceled.
PRVLORA_EVENTINFO_STATUS_TX_CHANNELBUSY	No free channel found by carrier sense.
PRVLORA_EVENTINFO_STATUS_TX_DUTYCYCLE_RESTRICTED	Transmission was aborted due to duty cycle restriction.
PRVLORA_EVENTINFO_STATUS_TX_RADIO_ERROR	Radio parameter is not supported.
PRVLORA_EVENTINFO_STATUS_KEYREQ_FAILED	An error occurred in the KeyReq procedure.

2.2.4 PrvLoRaIb_t

This enumeration type contains the information on the following Private LoRa Information Base (IB). These are used to get or to set the parameters in Private LoRa stack.

Table 9. PrvLoRaIb_t

Enumerator	Description
PRVLORA_IB_MACADDR	MAC address.
PRVLORA_IB_CHANNEL_ID	Channel ID to decide frequency. See below for the detail.
PRVLORA_IB_DR	Data rate ID to decide data rate. See below for the detail.
PRVLORA_IB_TXPOWER	Transmission power.
PRVLORA_IB_RXONWHENIDLE	Enable/disable reception in the idle state.
PRVLORA_IB_KEYREQ_PERMIT	Accept/reject `KeyReq` command frame.
PRVLORA_IB_TXCYCLE_TIME	Tx cycle information.
PRVLORA_IB_RADIO_CFG_CHECK_ENABLE	Enable/disable the regulatory function for each region in Radio Driver.

Available data rate IDs and channel IDs are defined for each region.

Table 10. Available Data Rate ID and Channel ID (AS1)

Data Rate ID	Configuration	Indicative physical bit rate	Channel ID	Frequency (MHz)	Steps (kHz)
0	LoRa®: SF12 – BW125 kHz	250 bps	0 – 63	915.2 – 927.8	200
1	LoRa®: SF11 – BW125 kHz	440 bps	0 – 63	915.2 – 927.8	200
2	LoRa®: SF10 – BW125 kHz	980 bps	0 – 63	915.2 – 927.8	200
3	LoRa®: SF9 – BW125 kHz	1760 bps	0 – 63	915.2 – 927.8	200
4	LoRa®: SF8 – BW125 kHz	3125 bps	0 – 63	915.2 – 927.8	200
5	LoRa®: SF7 – BW125 kHz	5470 bps	0 – 63	915.2 – 927.8	200
6	LoRa®: SF7 – BW250 kHz	11000 bps	0 – 31	915.3 – 927.7	400
7	FSK	50 kbps	0 – 63	915.2 – 927.8	200

Table 11. Available Data Rate ID and Channel ID (AS2)

Data Rate ID	Configuration	Indicative physical bit rate	Channel ID	Frequency (MHz)	Steps (kHz)
0	LoRa®: SF12 – BW125 kHz	250 bps	0 – 13	920.2 – 922.8	200
1	LoRa®: SF11 – BW125 kHz	440 bps	0 – 13	920.2 – 922.8	200
2	LoRa®: SF10 – BW125 kHz	980 bps	0 – 13	920.2 – 922.8	200
3	LoRa®: SF9 – BW125 kHz	1760 bps	0 – 13	920.2 – 922.8	200
4	LoRa®: SF8 – BW125 kHz	3125 bps	0 – 13	920.2 – 922.8	200
5	LoRa®: SF7 – BW125 kHz	5470 bps	0 – 13	920.2 – 922.8	200
6	LoRa®: SF7 – BW250 kHz	11000 bps	0 – 5	920.5 – 922.5	400
7	FSK	50 kbps	0 – 13	920.2 – 922.8	200

Table 12. Available Data Rate ID and Channel ID (AS3)

Data Rate ID	Configuration	Indicative physical bit rate	Channel ID	Frequency (MHz)	Steps (kHz)
0	LoRa®: SF12 – BW125 kHz	250 bps	0 – 28	915.2 – 920.8	200
1	LoRa®: SF11 – BW125 kHz	440 bps	0 – 28	915.2 – 920.8	200
2	LoRa®: SF10 – BW125 kHz	980 bps	0 – 28	915.2 – 920.8	200
3	LoRa®: SF9 – BW125 kHz	1760 bps	0 – 28	915.2 – 920.8	200
4	LoRa®: SF8 – BW125 kHz	3125 bps	0 – 28	915.2 – 920.8	200
5	LoRa®: SF7 – BW125 kHz	5470 bps	0 – 28	915.2 – 920.8	200
6	LoRa®: SF7 – BW250 kHz	11000 bps	0 – 13	915.3 – 920.5	400
7	FSK	50 kbps	0 – 28	915.2 – 920.8	200

Table 13. Available Data Rate ID and Channel ID (AS4)

Data Rate ID	Configuration	Indicative physical bit rate	Channel ID	Frequency (MHz)	Steps (kHz)
0	LoRa®: SF12 – BW125 kHz	250 bps	0 – 14	917.1 – 919.9	200
1	LoRa®: SF11 – BW125 kHz	440 bps	0 – 14	917.1 – 919.9	200
2	LoRa®: SF10 – BW125 kHz	980 bps	0 – 14	917.1 – 919.9	200
3	LoRa®: SF9 – BW125 kHz	1760 bps	0 – 14	917.1 – 919.9	200
4	LoRa®: SF8 – BW125 kHz	3125 bps	0 – 14	917.1 – 919.9	200
5	LoRa®: SF7 – BW125 kHz	5470 bps	0 – 14	917.1 – 919.9	200
6	LoRa®: SF7 – BW250 kHz	11000 bps	0 – 6	917.3 – 919.7	400
7	FSK	50 kbps	0 – 14	917.1 – 919.9	200

Table 14. Available Data Rate ID and Channel ID (JP)

Data Rate ID	Configuration	Indicative physical bit rate	Channel ID	Frequency (MHz)	Steps (kHz)
0	LoRa®: SF12 – BW125 kHz	250 bps	0 – 14	920.6 – 923.4	200
1	LoRa®: SF11 – BW125 kHz	440 bps	0 – 14	920.6 – 923.4	200
2	LoRa®: SF10 – BW125 kHz	980 bps	0 – 14	920.6 – 923.4	200
3	LoRa®: SF9 – BW125 kHz	1760 bps	0 – 14	920.6 – 923.4	200
4	LoRa®: SF8 – BW125 kHz	3125 bps	0 – 37	920.6 – 928.0	200
5	LoRa®: SF7 – BW125 kHz	5470 bps	0 – 37	920.6 – 928.0	200
6	LoRa®: SF7 – BW250 kHz	11000 bps	0 – 3	920.7 – 921.9	400
7	LoRa®: SF7 – BW250 kHz	11000 bps	0 – 13	922.7 – 927.9	400
8	LoRa®: SF7 – BW250 kHz	11000 bps	0 – 18	920.9 – 928.1	400
9	FSK	50 kbps	0 – 37	920.6 – 928.0	200

Table 15. Available Data Rate ID and Channel ID (JP-LDC)

Data Rate ID	Configuration	Indicative physical bit rate	Channel ID	Frequency (MHz)	Steps (kHz)
0	LoRa®: SF12 – BW125 kHz	250 bps	0 – 14	920.6 – 923.4	200
1	LoRa®: SF11 – BW125 kHz	440 bps	0 – 14	920.6 – 923.4	200
2	LoRa®: SF10 – BW125 kHz	980 bps	0 – 14	920.6 – 923.4	200
3	LoRa®: SF9 – BW125 kHz	1760 bps	0 – 14	920.6 – 923.4	200
4	LoRa®: SF8 – BW125 kHz	3125 bps	0 – 14	920.6 – 923.4	200
5	LoRa®: SF7 – BW125 kHz	5470 bps	0 – 14	920.6 – 923.4	200
6	FSK	50 kbps	0 – 14	920.6 – 923.4	200

Table 16. Available Data Rate ID and Channel ID (EU)

Data Rate ID	Configuration	Indicative physical bit rate	Channel ID	Frequency (MHz)	Steps (kHz)
0	LoRa®: SF12 – BW125 kHz	250 bps	0 – 27	863.1 – 868.5	200
1	LoRa®: SF11 – BW125 kHz	440 bps	0 – 27	863.1 – 868.5	200
2	LoRa®: SF10 – BW125 kHz	980 bps	0 – 27	863.1 – 868.5	200
3	LoRa®: SF9 – BW125 kHz	1760 bps	0 – 27	863.1 – 868.5	200
4	LoRa®: SF8 – BW125 kHz	3125 bps	0 – 27	863.1 – 868.5	200
5	LoRa®: SF7 – BW125 kHz	5470 bps	0 – 27	863.1 – 868.5	200
6	LoRa®: SF7 – BW250 kHz	11000 bps	0 – 11	863.2 – 867.6	400
7	FSK	50 kbps	0 – 27	863.1 – 868.5	200

Table 17. Available Data Rate ID and Channel ID (US)

Data Rate ID	Configuration	Indicative physical bit rate	Channel ID	Frequency (MHz)	Steps (kHz)
0	LoRa®: SF8 – BW125 kHz	3125 bps	0 – 41	920.9 – 927.5	600

Table 18. Available Data Rate ID and Channel ID (AU)

Data Rate ID	Configuration	Indicative physical bit rate	Channel ID	Frequency (MHz)	Steps (kHz)
0	LoRa®: SF8 – BW125 kHz	3125 bps	0 – 21	915.5 – 928.1	600

Table 19. Available Data Rate ID and Channel ID (IN)

Data Rate ID	Configuration	Indicative physical bit rate	Channel ID	Frequency (MHz)	Steps (kHz)
0	LoRa®: SF12 – BW125 kHz	250 bps	0 – 14	865.1 – 867.9	200
1	LoRa®: SF11 – BW125 kHz	440 bps	0 – 14	865.1 – 867.9	200
2	LoRa®: SF10 – BW125 kHz	980 bps	0 – 14	865.1 – 867.9	200
3	LoRa®: SF9 – BW125 kHz	1760 bps	0 – 14	865.1 – 867.9	200
4	LoRa®: SF8 – BW125 kHz	3125 bps	0 – 14	865.1 – 867.9	200
5	LoRa®: SF7 – BW125 kHz	5470 bps	0 – 14	865.1 – 867.9	200
6	FSK	50 kbps	0 – 14	865.1 – 867.9	200

Table 20. Available Data Rate ID and Channel ID (KR)

Data Rate ID	Configuration	Indicative physical bit rate	Channel ID	Frequency (MHz)	Steps (kHz)
0	LoRa®: SF12 – BW125 kHz	250 bps	0 – 14	865.1 – 867.9	200
1	LoRa®: SF11 – BW125 kHz	440 bps	0 – 14	865.1 – 867.9	200
2	LoRa®: SF10 – BW125 kHz	980 bps	0 – 14	865.1 – 867.9	200
3	LoRa®: SF9 – BW125 kHz	1760 bps	0 – 14	865.1 – 867.9	200
4	LoRa®: SF8 – BW125 kHz	3125 bps	0 – 14	865.1 – 867.9	200
5	LoRa®: SF7 – BW125 kHz	5470 bps	0 – 14	865.1 – 867.9	200
6	FSK	50 kbps	0 – 14	865.1 – 867.9	200

2.2.5 PrvLoRaMlme_t

This enumeration type contains the following Private LoRa management services.

Table 21. PrvLoRaMlme_t

Enumerator	Description
PRVLORA_MLME_KEY	Issues MAC command `KeyReq` to exchange the session key.
PRVLORA_MLME_DEVINFO	Issues MAC command `DevInfoReq` to get the remote device information.
PRVLORA_MLME_TXCYCLE	Issues MAC command `TxCycleReq` to request cyclic transmission to the remote device.

2.2.6 PrvLoRaNotifyId_t

This enumeration type contains the following Private LoRa notification type.

Table 22. PrvLoRaMlme_t

Enumerator	Description
PRVLORA_NOTIFY_UPDATE_REMOTEDEV	Remote device information is updated.

2.3 Structure Types

This section describes the following types.

Table 23. Structure Types

Types	Description
PrvLoRaIbRequest_t	Private LoRa IB parameters.
PrvLoRaTxOptions_t	Transmit options.
PrvLoRaMcpsReq_t	MCPS-Request primitive.
PrvLoRaMcpsCfm_t	MCPS-Confirm primitive.
PrvLoRaMcpsInd_t	MCPS-Indication primitive.
PrvLoRaMlmeReq_t	MLME-Request primitive.
PrvLoRaMlmeKeyReq_t	MLME-Request primitive – PRVLORA_MLME_KEY.
PrvLoRaMlmeDevInfoReq_t	MLME-Request primitive – PRVLORA_MLME_DEVINFO.
PrvLoRaMlmeTxCycleReq_t	MLME-Request primitive – PRVLORA_MLME_TXCYCLE.
PrvLoRaMlmeCfm_t	MLME-Confirm primitive.
PrvLoRaMlmeKeyCfm_t	MLME-Confirm primitive – PRVLORA_MLME_KEY.
PrvLoRaMlmeDevInfoCfm_t	MLME-Confirm primitive – PRVLORA_MLME_DEVINFO.
PrvLoRaMlmeTxCycleCfm_t	MLME-Confirm primitive – PRVLORA_MLME_TXCYCLE.
PrvLoRaMlmeInd_t	MLME-Indication primitive
PrvLoRaMlmeKeyInd_t	MLME-Indication primitive – PRVLORA_MLME_KEY
PrvLoRaMlmeTxCycleInd_t	MLME-Indication primitive – PRVLORA_MLME_TXCYCLE
PrvLoRaNotification_t	Private LoRa notification
PrvLoRaNotifyUpdatedRemoteDev_t	Private LoRa notification – PRVLORA_NOTIFY_UPDATE_REMOTEDEV

2.3.1 PrvLoRaIbRequest_t

This union type contains Private LoRa IB parameters. Each member type is a data structure for Private LoRa IB listed in ‘PrvLoRaIb_t’ enumeration.

Table 24. PrvLoRaIbRequest_t (union)

Member type	Member	Description	
uint8_t	macAddr[8]	[PRVLORA_IB_MACADDR] MAC address.	
uint8_t	channelId	[PRVLORA_IB_CHANNEL_ID] Channel ID to decide frequency. Available channel ID and actual frequencies are defined for each region. See 2.2.4.	
uint8_t	drIndex	[PRVLORA_IB_DR] Data rate ID to decide data rate. Available data rate is defined for each region. See 2.2.4.	
int8_t	txPower	[PRVLORA_IB_TXPOWER] transmission power in dB. Range: -128 – 127 Actual transmit power is determined by the regulation of each region and/or ability of the transceiver.	
bool	rxOnWhenIdle	[PRVLORA_IB_RXONWHENIDLE] Enable/disable reception in the idle state.	
		True	Enable reception.
		False	Disable reception.

bool	keyReqPermit	[PRVLORA_IB_KEYREQ_PERMIT] Accept/reject `KeyReq` command frame.	
		True	Accept `KeyReq` command.
		False	Reject `KeyReq` command.
PrvLoRaIbReqTxCycle_t	txCycle	[PRVLORA_IB_TXCYCLE_TIME] Tx cycle information.	
		uint8_t dstMacAddr[8]	Destination MAC address to send periodicity.
		uint32_t txCycleTime	Periodic time in sec. Range: 0, 10 – 131071 (0x1FFFF)
bool	radioCfgCheckEnable	[PRVLORA_IB_RADIO_CFG_CHECK_ENABLE] Enable/disable the regulatory function for each region in Radio Driver.	
		True	Enable the regulatory function.
		False	Disable the regulatory function.

2.3.2 PrvLoRaTxOptions_t

This union type contains information on Private LoRa transmit options.

Table 25. PrvLoRaTxOptions_t (union)

Member type	Member	Description
uint8_t	txOptValue	1Byte value of transmit options.
Struct {		(bit field)
uint8_t	AckRequest : 1	Request ACK to the destination.
uint8_t	SecEnable : 1	Encrypts data.
uint8_t	IndirectTx : 1	Indirect transmission.
uint8_t	_reserved : 6	RFU
}	options	

2.3.3 PrvLoRaMcpsReq_t

This structure type contains information on Private LoRa MCPS-Request.

Table 26. PrvLoRaMcpsReq_t

Member type	Member	Description
uint8_t	dstMacAddr[8]	Destination MAC address.
uint8_t	*p_txData	Pointer to the buffer of frame payload to send.
uint8_t	txDataSize	Size of frame payload to send.
uint16_t	txHandle	Handle value.
uint8_t	txOption	Transmit options. See 2.3.2.

2.3.4 PrvLoRaMcpsCfm_t

This structure type contains information on Private LoRa MCPS-Confirm.

Table 27. PrvLoRaMcpsCfm_t

Member type	Member	Description
PrvLoRaEventInfoStatus_t	eventStatus	Status of the operation. See 2.2.3.
uint16_t	txHandle	Handle value set for PrvLoRaMcpsReq_t. See 2.3.3.

2.3.5 PrvLoRaMcpsInd_t

This structure type contains information on Private LoRa MCPS-Indication.

Table 28. PrvLoRaMcpsInd_t

Member type	Member	Description
PrvLoRaEventInfoStatus_t	eventStatus	Status of the operation. See 2.2.3.
uint8_t	*p_srcMacAddr	Pointer to the buffer of source MAC address (8Byte).
uint8_t	*p_rxData	Pointer to the received data stream.
uint8_t	rxDataSize	Size of the received data stream.
int16_t	rssi	RSSI of the received packet.
int8_t	snr	SNR of the received packet.
bool	isAck	True if source device set ACK in the frame.
bool	isSecurity	True if the received frame is encrypted.

2.3.6 PrvLoRaMlmeReq_t

This structure type contains information on Private LoRa MLME-Request.

Table 29. PrvLoRaMlmeReq_t

Member type	Member	Description
PrvLoRaMlme_t	mlmeType	Type of Private LoRa management services. See 2.2.5.
union {		
PrvLoRaMlmeKeyReq_t	keyReq	PRVLORA_MLME_KEY.request parameters. See 2.3.7.
PrvLoRaMlmeDevInfoReq_t	devInfoReq	PRVLORA_MLME_DEVINFO.request parameters. See 0.
PrvLoRaMlmeTxCycleReq_t	txCycleReq	PRVLORA_MLME_TXCYCLE.request parameters. See 2.3.9.
}	req	

2.3.7 PrvLoRaMlmeKeyReq_t

This structure type contains information on Private LoRa MLME-Request to request session key to the destination.

Table 30. PrvLoRaMlmeKeyReq_t

Member type	Member	Description
uint8_t	dstMacAddr[8]	MAC address of the destination.
PrvLoRaTxOptions_t	txOptions	Transmit options. See 2.3.2. IndirectTx is available, others are ignored.

2.3.8 PrvLoRaMlmeDevInfoReq_t

This structure type contains information on Private LoRa MLME-Request to request device information to the destination.

Table 31. PrvLoRaMlmeDevInfoReq_t

Member type	Member	Description
uint8_t	dstMacAddr[8]	MAC address of the destination.
PrvLoRaTxOptions_t	txOptions	Transmit options. See 2.3.2. SecEnable and IndirectTx are available, others are ignored.

2.3.9 PrvLoRaMlmeTxCycleReq_t

This structure type contains information on Private LoRa MLME-Request to request cyclic transmission to the destination.

Table 32. PrvLoRaMlmeTxCycleReq_t

Member type	Member	Description
uint8_t	dstMacAddr[8]	MAC address of the destination.
PrvLoRaTxOptions_t	txOptions	Transmit options. See 2.3.2. SecEnable and IndirectTx are available, others are ignored.
uint32_t	txCycleTime	Periodicity in sec. (0, 10 – 0x0001FFFF)

2.3.10 PrvLoRaMlmeCfm_t

This structure type contains information on Private LoRa MLME-Confirm primitive.

Table 33. PrvLoRaMlmeCfm_t

Member type	Member	Description
PrvLoRaMlme_t	mlmeType	Type of Private LoRa management services. See 2.2.5.
union {		
PrvLoRaMlmeKeyCfm_t	keyCfm	PRVLORA_MLME_KEY.confirm parameters. See 2.3.11.
PrvLoRaMlmeDevInfoCfm_t	devInfoCfm	PRVLORA_MLME_DEVINFO.confirm parameters. See 2.3.12.
PrvLoRaMlmeTxCycleCfm_t	txCycleCfm	PRVLORA_MLME_TXCYCLE.confirm parameters. See 2.3.13.
}	cfm	

2.3.11 LoRaMlmeKeyCfm_t

This structure type contains information on Private LoRa MLME-Confirm to request session key to the destination.

Table 34. PrvLoRaMlmeKeyCfm_t

Member type	Member	Description
PrvLoRaEventInfoStatus_t	status	Status of the operation. See 2.2.3.
uint8_t	dstMacAddr[8]	MAC address of the destination.

2.3.12 PrvLoRaMlmeDevInfoCfm_t

This structure type contains information on Private LoRa MLME-Confirm to request device information to the destination.

Table 35 PrvLoRaMlmeDevInfoCfm_t

Member type	Member	Description
PrvLoRaEventInfoStatus_t	status	Status of the operation. See 2.2.3.
uint8_t	snr	S/N ratio.
uint8_t	txPower	Transmission power.
uint32_t	txCycleTime	Periodicity of cyclic transmission.

2.3.13 PrvLoRaMlmeTxCycleCfm_t

This structure type contains information on Private LoRa MLME-Confirm to request periodical transmission to the destination.

Table 36 PrvLoRaMlmeTxCycleCfm_t

Member type	Member	Description
PrvLoRaEventInfoStatus_t	status	Status of the operation. See 2.2.3.

2.3.14 PrvLoRaMlmeInd_t

This structure type contains information on Private LoRa MLME-Indication primitive.

Table 37. PrvLoRaMlmeInd_t

Member type	Member	Description
PrvLoRaMlme_t	mlmeType	Type of Private LoRa management services. See 2.2.5.
union {		
PrvLoRaMlmeKeyInd_t	keyInd	PRVLORA_MLME_KEY.indiation parameters. See 2.3.15.
PrvLoRaMlmeTxCycleInd_t	txCycleInd	PRVLORA_MLME_TXCYCLE.indication parameters. See 2.3.16.
}	ind	

2.3.15 PrvLoRaMlmeKeyInd_t

This structure type contains information on Private LoRa MLME-Indication when a remote device requests the exchange of session key.

Table 38 PrvLoRaMlmeKeyInd_t

Member type	Member	Description
uint8_t	srcMacAddr[8]	MAC address of the remote device.

2.3.16 PrvLoRaMlmeTxCycleInd_t

This structure type contains information on Private LoRa MLME-Indication when a remote device requests cyclic transmission.

Table 39 PrvLoRaMlmeTxCycleInd_t

Member type	Member	Description
uint8_t	srcMacAddr[8]	MAC address of the remote device.
uint32_t	txCycleTime	Periodicity in sec.
bool	isSecurity	True if the received `TxCycleReq` command is encrypted.

2.4 Private LoRa APIs

This section contains the following functions.

Table 40. Private LoRa APIs

function	Description
PrivateLoRaInitialization	Initialize Private LoRa.
PrivateLoRaStart	Start Private LoRa.
PrivateLoRaStop	Stop Private LoRa.
PrivateLoRaProcess	Process the interruption.
PrivateLoRaGetRequest	Request Private LoRa Information Base service to get attribute of the Private LoRa layer.
PrivateLoRaSetRequest	Request Private LoRa Information Base service to set attribute of the Private LoRa layer.
PrivateLoRaRegisterRemoteDevice	Register the set of remote device and security information.
PrivateLoRaUnregisterRemoteDevice	Unregister the set of remote device and security information.
PrivateLoRaMcpsRequest	Request the Mac Layer Management Entity to handle the management service.
PrivateLoRaMlmeRequest	Request the Mac Common Part Sublayer to handle the data services

2.4.1 PrivateLoRaInitialization

PrvLoRaStatus_t PrivateLoRaInitialization(PrvLoRaPrimitives_t *p_primitives, PrvLoRaRegion_t region)		
This function initializes Private LoRa layer. Event handler functions to be set in ‘p_primitives’ are mandatory and user must implement them.		
Parameters		
	[IN] p_primitives	Pointer to a structure defining the Private LoRa event handler functions. Must set all handler functions. See 0.
	[IN] region	The region to start. See 2.2.1.
Return		
	PRVLORA_STATUS_OK	Initialization finished successfully.
	PRVLORA_STATUS_ERROR	Initialization failure.
	PRVLORA_STATUS_PARAMETER_INVALID	Requested parameter is invalid.
	PRVLORA_STATUS_NOT_SUPPORTED	Requested region is not supported.
	PRVLORA_STATUS_RADIO_ERROR	Radio driver initialization failure.

2.4.2 PrivateLoRaStart

PrvLoRaStatus_t PrivateLoRaStart(void)	
This function starts Private LoRa MAC process.	
Parameters	
-	
Return	
PRVLORA_STATUS_OK	Private LoRa was started successfully.
PRVLORA_STATUS_ERROR	Private LoRa could not be started or already started.
PRVLORA_STATUS_RADIO_PARAMETER_INVALID	Private LoRa could not be started because radio parameter (set of data rate and channel) is invalid.

2.4.3 PrivateLoRaStop

PrvLoRaStatus_t PrivateLoRaStop(void)	
This function stops Private LoRa MAC process.	
Parameters	
-	
Return	
PRVLORA_STATUS_OK	Private LoRa was stopped successfully.
PRVLORA_STATUS_ERROR	Private LoRa is not initialized.
PRVLORA_STATUS_BUSY	Private LoRa is busy and could not be stopped.

2.4.4 PrivateLoRaProcess

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

2.4.5 PrivateLoRaGetRequest

PrvLoRaStatus_t PrivateLoRaGetRequest(PrvLoRaIb_t ibId, PrvLoRaIbRequest_t *p_ibGet)		
This function is the Private LoRa information base service to get attributes of the Private LoRa layer.		
Parameters		
[IN] ibId	Private LoRa MAC attribute type to get. See 2.2.4.	
[OUT] p_ibGet	Parameters got from Private LoRa. See 2.3.1.	
Return		
PRVLORA_STATUS_OK	The request is finished successfully.	
PRVLORA_STATUS_ERROR	Private LoRa is not initialized.	
PRVLORA_STATUS_PARAMETER_INVALID	Requested parameter is invalid.	
PRVLORA_STATUS_SERVICE_UNKNOWN	Requested information base is unknown.	

2.4.6 PrivateLoRaSetRequest

PrvLoRaStatus_t PrivateLoRaSetRequest(PrvLoRaIb_t ibId, PrvLoRaIbRequest_t *p_ibSet)		
This function is the PrivateLoRa information base service to set attributes of the Private LoRa layer. Information base cannot be changed when PrivateLoRa service is running. Information base parameters set by this function are initialized to their default values upon calling `PrivateLoRaInitialization()`.		
Parameters		
[IN] ibId	Private LoRa MAC attribute type to set. See 2.2.4.	
[IN] p_ibSet	Parameters got from Private LoRa. See 2.3.1.	
Return		
PRVLORA_STATUS_OK	The request is finished successfully.	
PRVLORA_STATUS_ERROR	Private LoRa is not initialized.	
PRVLORA_STATUS_BUSY	Private LoRa is busy. Another service is running.	
PRVLORA_STATUS_PARAMETER_INVALID	Requested parameter is invalid.	
PRVLORA_STATUS_SERVICE_UNKNOWN	Requested information base is unknown.	
PRVLORA_STATUS_IB_ATTRIBUTE_INVALID	Requested information base is read only.	

2.4.7 PrivateLoRaRegisterRemoteDevice

PrvLoRaStatus_t PrivateLoRaRegisterRemoteDevice(uint8_t *p_remoteMacAddr, uint8_t *p_psk, uint8_t *p_sessionKey, uint32_t initFrameCounterTx, uint32_t initFrameCounterRx)	
This function is to register the set of remote device and security information. Registration is required to send and receive frames with the remote device. If secured frame never be used, set dummy data to PSK (`*p_psk`) and frame counter (`initFrameCounterTx` and `initFrameCounterRx`).	
Parameters	
[IN] p_remoteMacAddr	MAC address of the remote device to register.
[IN] p_psk	Set 128bit (16Byte) PSK. NULL cannot be set. If secured frame never be used, set dummy data.
[IN] p_sessionKey	Set 128bit (16Byte) session key if it is known. Otherwise set NULL.
[IN] initFrameCounterTx	Initial frame counter for Tx. If secured frame never be used, set dummy value.
[IN] initFrameCounterRx	Initial frame counter for Rx. Note that set 0xFFFFFFFF to receive the frame whose frame counter is zero. If secured frame never be used, set dummy value.
Return	
PRVLORA_STATUS_OK	The request is finished successfully.
PRVLORA_STATUS_ERROR	Private LoRa is not initialized, or no free entry to register new device information.
PRVLORA_STATUS_BUSY	Private LoRa is busy. Another service is running.
PRVLORA_STATUS_PARAMETER_INVALID	Requested parameter is invalid.

2.4.8 PrivateLoRaUnregisterRemoteDevice

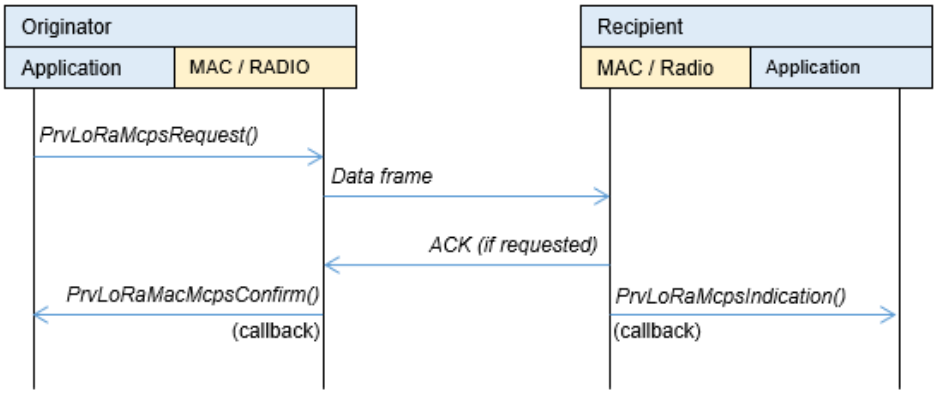
PrvLoRaStatus_t PrivateLoRaUnregisterRemoteDevice(uint8_t *p_remoteMacAddr)	
This function is to unregister the set of remote device and security information. Note that if indirect transmission requests to the specified remote device is remained, these are canceled.	
Parameters	
[IN] p_remoteMacAddr	MAC address of the remote device to unregister. All remote device information will be unregistered if it is NULL.
Return	
PRVLORA_STATUS_OK	The request is finished successfully.
PRVLORA_STATUS_ERROR	Private LoRa is not initialized.
PRVLORA_STATUS_BUSY	Private LoRa is busy. Another service is running.
PRVLORA_STATUS_PARAMETER_INVALID	Requested parameter is invalid.
PRVLORA_STATUS_NO_ REMOTE_DEVICE_ENTRY	Specified remote device is not registered.

2.4.9 PrivateLoRaMcpsRequest

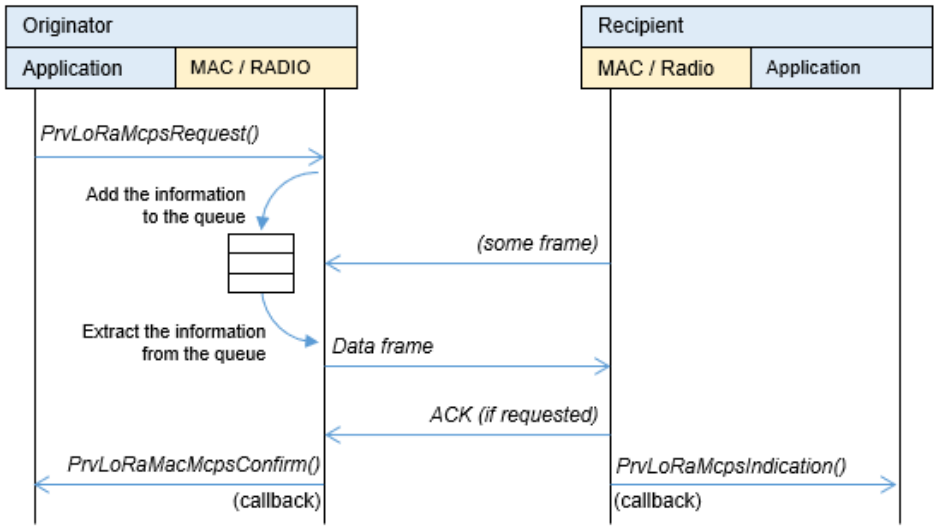
PrvLoRaStatus_t PrivateLoRaMcpsRequest(PrvLoRaMcpsReq_t *p_mcpsReq)	
This function requests Private LoRa MCPS-Request (send data message). When the process of the transmission and the reception finish, the callback function `PrvLoRaMacMcpsConfirm()` in the `PrvLoRaPrimitives_t` will be called.	
Parameters	
[IN] p_mcpsReq	Pointer to a structure of MCPS-Request. See 2.3.3.
Return	
PRVLORA_STATUS_OK	The request is finished successfully.
PRVLORA_STATUS_ERROR	Private LoRa is not initialized.
PRVLORA_STATUS_INACTIVE	Private LoRa is not started.
PRVLORA_STATUS_BUSY	Private LoRa is busy. Another service is running.
PRVLORA_STATUS_PARAMETER_INVALID	Requested parameter is invalid.
PRVLORA_STATUS_NO_NETWORK	Not joined to the network.
PRVLORA_STATUS_NO_ REMOTE_DEVICE_ENTRY	The destination device is not registered. See 2.4.7.

A flow diagram of MCPS sequence is shown below for reference.

- Direct transmission



- Indirect transmission

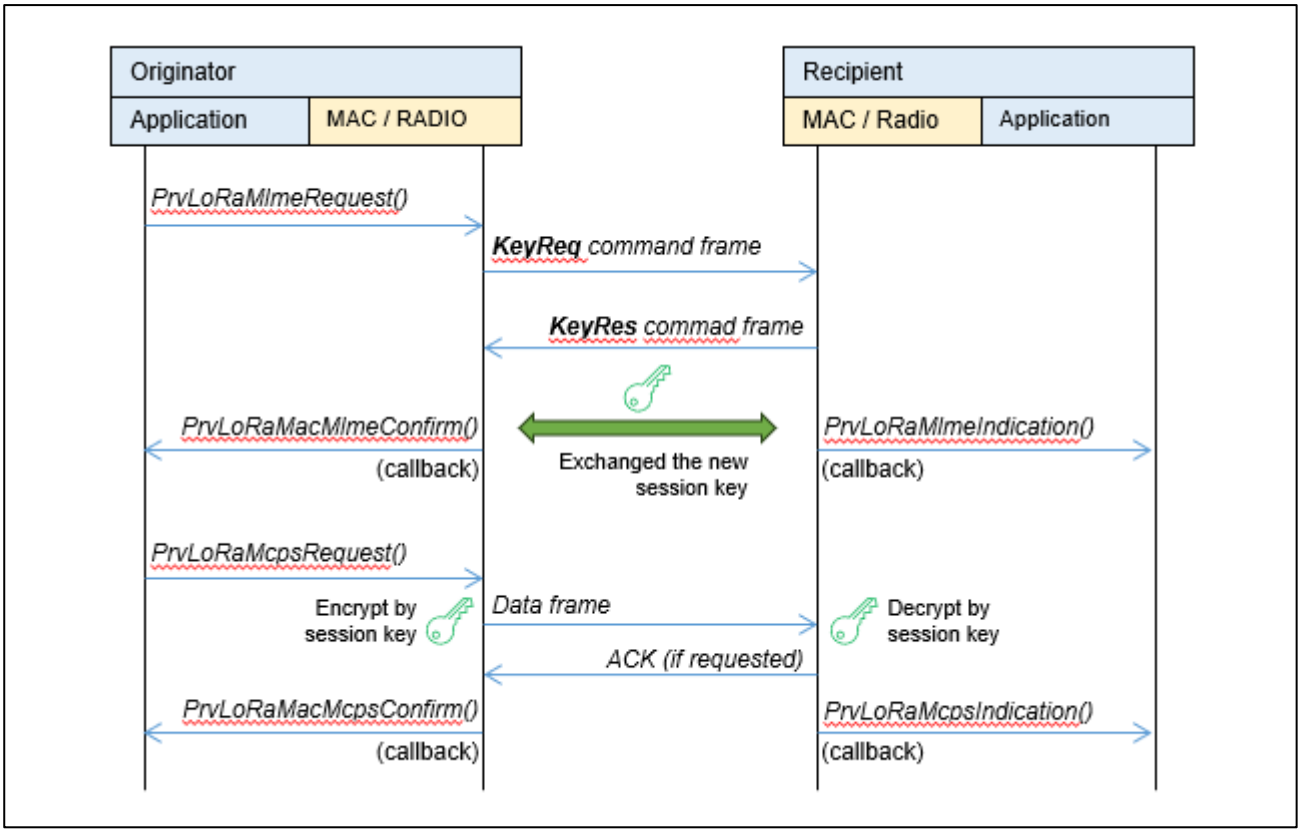


2.4.10 PrivateLoRaMlmeRequest

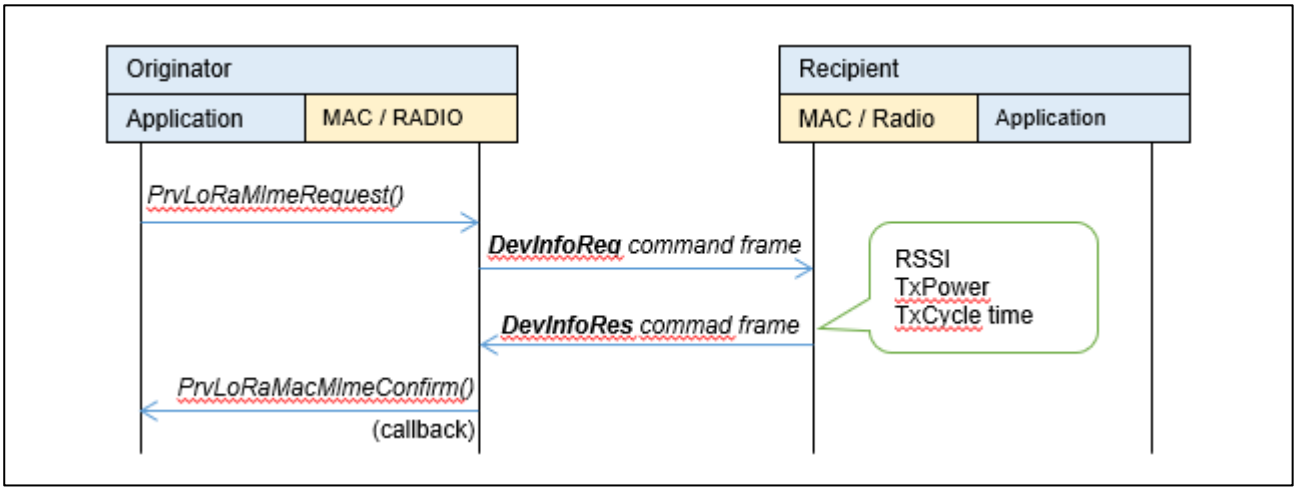
PrvLoRaStatus_t PrivateLoRaMlmeRequest(PrvLoRaMlmeReq_t *p_mlmeReq)	
This function requests Private LoRa MLME-Request (send MAC command). When the process of the transmission and the reception finish, the callback function `PrvLoRaMacMlmeConfirm()` in the `PrvLoRaPrimitives_t` will be called.	
Parameters	
[IN] p_mlmeReq	Pointer to a structure of MLME-Request. See 2.3.6.
Return	
PRVLORA_STATUS_OK	The request is finished successfully.
PRVLORA_STATUS_ERROR	Private LoRa is not initialized.
PRVLORA_STATUS_INACTIVE	Private LoRa is not started.
PRVLORA_STATUS_BUSY	Private LoRa is busy. Another service is running.
PRVLORA_STATUS_PARAMETER_INVALID	Requested parameter is invalid.
PRVLORA_STATUS_NO_NETWORK	Not joined to the network.
PRVLORA_STATUS_NO_ REMOTE_DEVICE_ENTRY	The destination device is not registered. See 2.4.7.

A flow diagram of MLME sequence is shown below for reference.

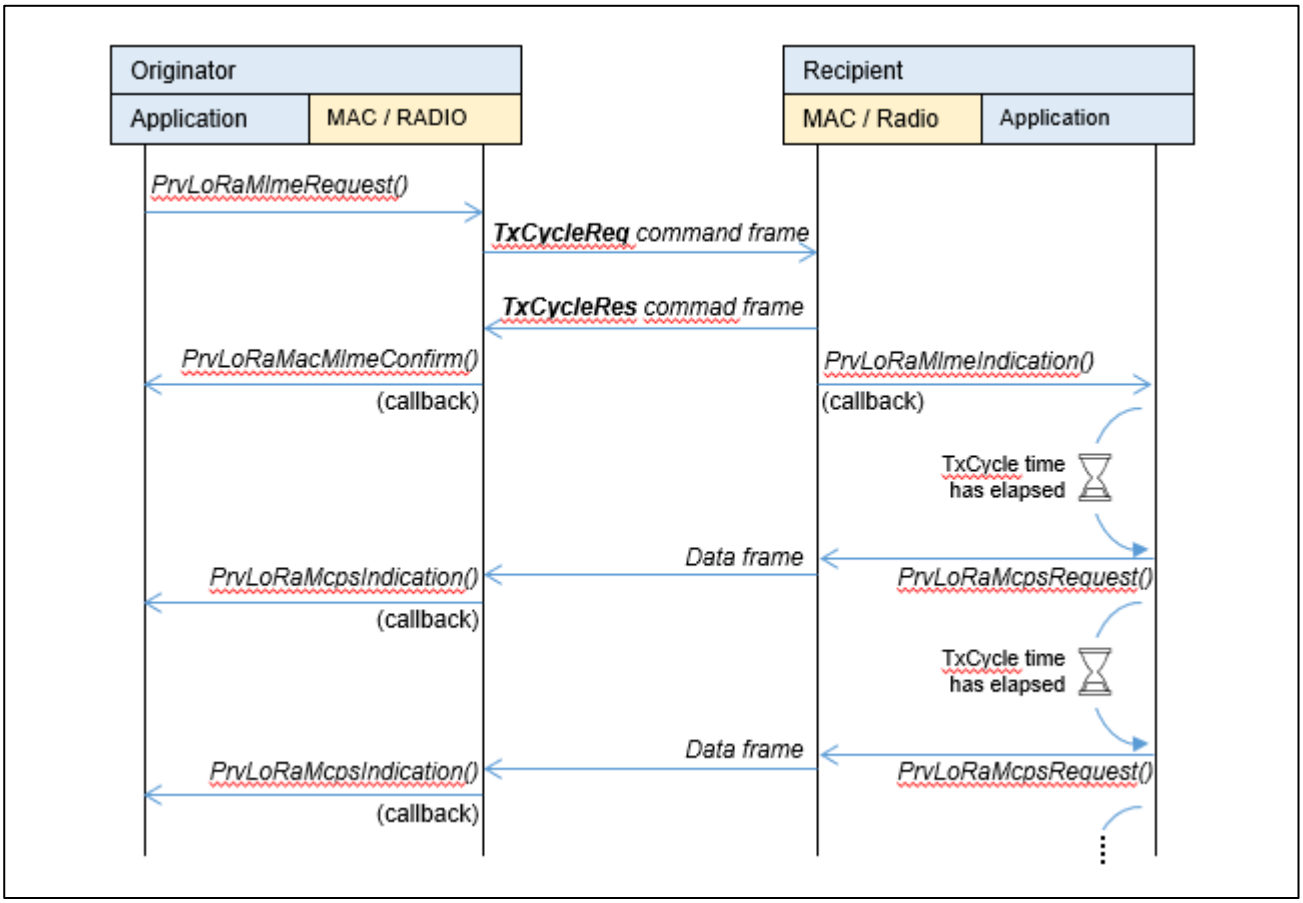
- PRVLORA_MLME_KEY



- PRVLORA_MLME_DEVINFO



- PRVLORA_MLME_TXCYCLE



2.5 Private LoRa Primitive Callback Handler (LoRaMacPrimitives_t)

This type is a structure containing Private LoRa events handler functions used to notify upper layers Private LoRa primitive events. This structure includes the following member functions. These members must be set before calling `PrivateLoRaInitialization()` (see 2.4.1).

Table 41. PrvLoRaPrimitives_t

Member	Description
void (*PrvLoRaMacMcpsConfirm)(PrvLoRaMcpsCfm_t *p_mcpsCfm)	Notify `PrivateLoRaMcpsRequest()` processed (see 2.4.9).
void (*PrvLoRaMacMcpsIndication)(PrvLoRaMcpsInd_t *p_mcpsInd)	Notify the payload received.
Void (*PrvLoRaMacMlmeConfirm)(PrvLoRaMlmeCfm_t *p_MlmeCfm)	Notify `PrivateLoRaMlmeRequest()` was accepted (see 2.4.10).
void (*PrvLoRaMacMlmeIndication)(PrvLoRaMlmeInd_t *p_MlmeInd)	Notify the requested message received.
Void (*PrvLoRaMacNotification)(PrvLoRaNotification_t *p_notify)	Notify the Private LoRa event.

2.5.1 PrvLoRaMacMcpsConfirm

void (*PrvLoRaMacMcpsConfirm)(PrvLoRaMcpsCfm_t *p_mcpsCfm)	
This function notify that `PrivateLoRaMcpsRequest()` processed.	
Parameters	
[IN] p_mcpsCfm	Pointer to a structure of MCPS-Confirm. See 2.3.4.
Return	
-	

2.5.2 PrvLoRaMacMcpsIndication

void (*PrvLoRaMacMcpsIndication)(PrvLoRaMcpsInd_t *p_mcpsInd)	
This function notifies the received payload.	
Parameters	
[IN] p_mcpsInd	Pointer to a structure of MCPS-Indication. See 2.3.5.
Return	
-	

2.5.3 PrvLoRaMacMlmeConfirm

void (*PrvLoRaMacMlmeConfirm)(PrvLoRaMlmeCfm_t *p_MlmeCfm)	
This function notifies `PrivateLoRaMlmeRequest()` was completed	
Parameters	
[IN] p_mlmeCfm	Pointer to a structure of MLME-Confirm. See 2.3.9.
Return	
-	

2.5.4 PrvLoRaMacMlmeIndication

void (*PrvLoRaMacMlmeIndication)(PrvLoRaMlmeInd_t *p_MlmeInd)	
This function notifies Private LoRa MLME-Indication primitive	
Parameters	
[IN] p_mlmeInd	Pointer to a structure of MLME-Indication. See 2.3.13.
Return	
-	

2.5.5 PrvLoRaMacNotification

void (*PrvLoRaMacMlmeIndication)(PrvLoRaMlmeInd_t *p_MlmeInd)	
This function notifies Private LoRa MLME-Indication primitive	
Parameters	
[IN] p_mlmeInd	Pointer to a structure of MLME-Indication. See 2.3.13.
Return	
-	

3. Timer

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

4. Power Saving

4.1 PrivateLoRaSetLowPower

PrvLoRaStatus_t PrivateLoRaSetLowPower(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 Private LoRa is not busy. This function is not available for Private LoRa coordinator and devices which Rx is running when in idle.	
Parameters	
-	
Return	
PRVLORA_STATUS_OK	MCU could be set to the low power mode and returned to the normal mode successfully.
PRVLORA_STATUS_BUSY	MCU cannot be set to the low power mode due to the following reasons: - Private LoRa is busy. - Application disallows MCU to be set to low power mode. (See below)
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 Code (Informative)

5.1 Initialize and Start the Private LoRa

The following code-snippet shows how to use the API to initialize and start the Private LoRa.

```
#include “PrivateLoRa.h”

static void AppPrvLoRaCallbackMcpsConfirm( PrvLoRaMcpsCfm_t *p_mcpsCfm );
static void AppPrvLoRaCallbackMcpsIndication( PrvLoRaMcpsInd_t *p_mcpsInd );
static void AppPrvLoRaCallbackMlmeConfirm( PrvLoRaMlmeCfm_t *p_mlmeCfm );
static void AppPrvLoRaCallbackMlmeIndication( PrvLoRaMlmeInd_t *p_mlmeInd );
static void AppPrvLoRaCallbackMacNotification(PrvLoRaNotification_t *p_notify );

PrvLoRaPrimitives_t  appPrvLoRaPrimitives;

/* Set Private LoRa primitives */
appPrvLoRaPrimitives.PrvLoRaMacMcpsConfirm      = AppPrvLoRaCallbackMcpsConfirm;
appPrvLoRaPrimitives.PrvLoRaMacMcpsIndication   = AppPrvLoRaCallbackMcpsIndication;
appPrvLoRaPrimitives.PrvLoRaMacMlmeConfirm      = AppPrvLoRaCallbackMlmeConfirm;
appPrvLoRaPrimitives.PrvLoRaMacMlmeIndication   = AppPrvLoRaCallbackMlmeIndication;
appPrvLoRaPrimitives.PrvLoRaMacNotification     = AppPrvLoRaCallbackMacNotification;

/* Initialize Private LoRa */
PrivateLoRaInitialization( &appPrvLoRaPrimitives, PRVLORA_REGION_EU );

/* Start Private LoRa */
PrivateLoRaStart();
```

5.2 Set Private LoRa IB Parameters

The following code-snippet shows how to use the API to set the parameters; `MAC address`, `channel ID`, and `data rate ID`.

```
Static uint8_t MacAddr[ 8 ] = { 0x00, 0x11, 0x22, 0x33, 0x44, 0x55, 0x66 0x77 };

PrvLoRaIbRequest_t  ibSet;

/* MAC address */
memcpy( &(ibSet.macAddr[0]), MacAddr, 8 );
PrivateLoRaSetRequest( PRVLORA_IB_MACADDR, &ibSet );

/* Channel ID and data rate */
ibSet.drIndex = 2;  // Data rate ID = 2 : LoRa®: SF10 – BW125 kHz
PrivateLoRaSetRequest( PRVLORA_IB_DR, &ibSet );

ibSet.channelId = 1;  // channel ID = 1 : 863.3MHz (in case data rate ID = 2)
PrivateLoRaSetRequest( PRVLORA_IB_CHANNEL_ID, &ibSet );
```

5.3 Get Private LoRa IB Parameters

The following code-snippet shows how to use the API to get the parameters; `MAC address`, `channel ID`, and `data rate ID`.

```
Static uint8_t MacAddr[ 8 ], datarate_ID, channel_ID;

PrvLoRaIbRequest_t  ibGet;
PrvLoRaStatus_t       status;

/* MAC Address */
status = PrivateLoRaGetRequest( PRVLORA_IB_MACADDR, &ibGet );
if( status == PRVLORA_STATUS_OK )
{
      memcpy( MacAddr, &(ibGet.macAddr[0]), 8 );
}

/* Channel ID and data rate */
status = PrivateLoRaGetRequest( PRVLORA_IB_DR, &ibGet );
if( status == PRVLORA_STATUS_OK )
{
      datarate_ID = ibGet.drIndex;
}
status = PrivateLoRaGetRequest( PRVLORA_IB_CHANNEL_ID, &ibGet );
if( status == PRVLORA_STATUS_OK )
{
      channel_ID = ibGet.channelId;
}
```

5.4 Remote Device Registration for Communication

The following code-snippet shows how to use the API to communicate with remote device. Registration is required to send and receive frames with the remote device. If secured frame never be used, set dummy data to PSK and frame counter.

```
Static uint8_t DestMacAddr[ 8 ] = { 0x88, 0x99, 0xAA, 0xBB, 0xCC, 0xDD, 0xEE, 0xFF };
static uint8_t PSK[ 16 ] = { 0x2B, 0x7E, 0x15, 0x16, 0x28, 0xAE, 0xD2, 0xA6, 0xAB, 0xF7, 0x15, 0x88,
0x09, 0xCF, 0x4F, 0x3C };

uint8_t *p_sessionKey = NULL;  // set session key (16 Byte) if known.

/* Resigter remote device */
PrivateLoRaRegisterRemoteDevice( DevMacAddr, PSK, p_sessionKey, 0, 0xFFFFFFFF );
```

5.5 Send Data

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

```
Static uint8_t DestMacAddr[ 8 ] = { 0x88, 0x99, 0xAA, 0xBB, 0xCC, 0xDD, 0xEE, 0xFF };
uint8_t DataBuff[ 3 ] = { 1, 2, 3 };

PrvLoRaMcpsReq_t   mcpsReq;
PrvLoRaStatus_t    status;

memcpy( &(mcpsReq.dstMacAddr[0]), &(DestMacAddr[0]), 8 );
mcpsReq.p_txData                        = DataBuff;
mcpsReq.txDataSize                      = sizeof( DataBuff );
mcpsReq.txHandle                        = 0;    // arbitrary value
mcpsReq.txOptions.txOptValue            = 0;    // (initialize)
mcpsReq.txOptions.options.AckRequest    = 1;    // request ACK to the destination (if necessary)
mcpsReq.txOptions.options.SecEnable     = 1;    // encrypt message (if necessary)

status = PrivateLoRaMcpsRequest( &mcpsReq );
if( status == PRVLORA_STATUS_OK )
{
    // Service started successfully. Waiting for the MCPS-Confirm event (callback).
}
```

5.6 Send Command

The following code-snippet shows how to use the API to send command `KeyReq`.

```
Static uint8_t DestMacAddr[ 8 ] = { 0x88, 0x99, 0xAA, 0xBB, 0xCC, 0xDD, 0xEE, 0xFF };

PrvLoRaMlmeReq_t   mlmeReq;
PrvLoRaStatus_t    status;

mlmeReq.mlmeType = PRVLORA_MLME_KEY;
memcpy( &(mlmeReq.req.keyReq.dstMacAddr[0]), &(DestMacAddr[0]), 8 );
mlmeReq.req.keyReq.TxOption.txOptValue          = 0;
mlmeReq.req.keyReq.TxOption.options.InderectTx = 0;    // indirect transmission (set 1 if necessary)

status = PrivateLoRaMlmeRequest( &mlmeReq );
if( status == PRVLORA_STATUS_OK )
{
    // Service started successfully. Waiting for the MLME-Confirm event (callback).
}
```

Revision History

Rev.	Date	Description	
		Section	Summary
04.40	Dec.22.23	-	Initial Release
04.50	May.24.24	-	Supported RA0E1 (R7FA0E1073CFJ) as a target device.
04.60	Sep.27.24	1.2	Updated directories.
		1.5	Added related document [5].
04.70	Apr.18.25	-	Supported RA0E2 (R7FA0E2094CFM) as a target device.

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 be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. 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.
6. 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.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. 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.
9. 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.
10. 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.
11. 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.
12. 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.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. 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.5.0-1 October 2020)

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

Arm® and Cortex® are registered trademarks of Arm Limited. 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.