



LMT SoM library API

1.0.3

Generated by Doxygen 1.13.2

1 File Index	1
1.1 File List	1
2 File Documentation	2
2.1 lmt_coap_manager.h File Reference	2
2.1.1 Macro Definition Documentation	2
2.1.1.1 APP_COAP_MAX_MSG_LEN	2
2.1.2 Function Documentation	2
2.1.2.1 getNetworkQuality()	2
2.1.2.2 modemShutdown()	3
2.1.2.3 sendEventCmdRes()	3
2.1.2.4 triggerDataPacking()	3
2.1.2.5 triggerMailer()	3
2.2 lmt_common.h File Reference	4
2.2.1 Macro Definition Documentation	4
2.2.1.1 FIRST_USER_STATUS_BIT	4
2.2.1.2 LAST_USER_STATUS_BIT	4
2.2.2 Function Documentation	4
2.2.2.1 checkBootOkMask()	4
2.2.2.2 criticalError()	5
2.2.2.3 resetStatusBit()	5
2.2.2.4 setBootOkBit()	5
2.2.2.5 setUserBootOkMask()	5
2.3 lmt_proto_handler.h File Reference	5
2.3.1 Macro Definition Documentation	6
2.3.1.1 I_TAPE	6
2.3.1.2 MAX_ACTION_PARAMETERS_SIZE	6
2.3.1.3 MAX_COLUMNS_COUNT	6
2.3.1.4 MAX_PERIODS_COUNT	7
2.3.1.5 MAX_TAPE_COUNT	7
2.3.1.6 MAX_TRACKS_COUNT	7
2.3.2 Function Documentation	7
2.3.2.1 addColumnToTape()	7
2.3.2.2 decodeMessage()	7
2.3.2.3 dumpMemory()	7
2.3.2.4 encodeMessage()	8
2.3.2.5 getEncodedMsgBuffer()	8
2.3.2.6 getEncodedMsgLen()	8
2.3.2.7 getLastPeriod()	8
2.3.2.8 getTapeRecordsCount()	9
2.3.2.9 isDataChanged()	9
2.3.2.10 isUdpPacketFull()	9

2.3.2.11 restartMeasurements()	9
2.3.2.12 rewindTape()	9
2.3.2.13 updatePeriod()	10
2.4 lmt_sdk_api.h File Reference	10
2.4.1 Function Documentation	10
2.4.1.1 lmtInit()	10
2.4.1.2 loop()	10
2.4.1.3 setup()	11
2.5 lmt_settings.h File Reference	11
2.5.1 Enumeration Type Documentation	12
2.5.1.1 logLevel	12
2.5.2 Function Documentation	12
2.5.2.1 getFileUIRetries()	12
2.5.2.2 getLogFileMaxSize()	13
2.5.2.3 getLogLevel()	13
2.5.2.4 getLogRotationFrequency()	13
2.5.2.5 getMaxResendAttempts()	13
2.5.2.6 getMaxResendTimeout()	14
2.5.2.7 getNoPsmUplinkTimeout()	14
2.5.2.8 getNumOfLogFiles()	14
2.5.2.9 getResendPacketInitialTimeout()	14
2.5.2.10 getResponseWaitTimeout()	15
2.5.2.11 getUplinkTimeout()	15
2.5.2.12 setFileUIRetries()	15
2.5.2.13 setLogFileMaxSize()	15
2.5.2.14 setLogLevel()	15
2.5.2.15 setLogRotationFrequency()	16
2.5.2.16 setMaxResendAttempts()	16
2.5.2.17 setMaxResendTimeout()	16
2.5.2.18 setNoPsmUplinkTimeout()	17
2.5.2.19 setNumOfLogFiles()	17
2.5.2.20 setResendPacketInitialTimeout()	17
2.5.2.21 setResponseWaitTimeout()	18
2.5.2.22 setUplinkTimeout()	18
2.6 lmt_som_event_emitter.h File Reference	18
2.6.1 Enumeration Type Documentation	19
2.6.1.1 som_event_t	19
2.6.2 Function Documentation	19
2.6.2.1 somEventHandler()	19
2.7 lmt_storage_manager.h File Reference	20
2.7.1 Function Documentation	20
2.7.1.1 checkFwVersion()	20

2.7.1.2 eraseFlash()	21
2.7.1.3 logError()	21
2.7.1.4 logInfo()	21
2.7.1.5 logInfoFormatted()	21
2.7.1.6 logStringHex()	22
2.7.1.7 logWarning()	22
2.7.1.8 saveSettings()	22
2.7.1.9 settingsFileRead()	22

Index	23
--------------	-----------

Chapter 1

File Index

1.1 File List

Here is a list of all files with brief descriptions:

lmt_coap_manager.h	2
lmt_common.h	4
lmt_proto_handler.h	5
lmt_sdk_api.h	10
lmt_settings.h	11
lmt_som_event_emitter.h	18
lmt_storage_manager.h	20

Chapter 2

File Documentation

2.1 lmt_coap_manager.h File Reference

Macros

- `#define APP_COAP_MAX_MSG_LEN 1280`

Functions

- `void getNetworkQuality (int32_t *rsrp, int32_t *rsrq, int32_t *snr)`
Function gets the cached signal quality from the modem.
- `void triggerDataPacking (bool set_include_radio_parms)`
Enable packer thread to create a CoAP message.
- `void triggerMailer (bool trigger_radio_data_packing)`
Enable mailer thread; first data packet containing radio data can be created.
- `void modemShutdown (void)`
Function to close the socket and power off the modem.
- `void sendEventCmdRes (int res)`
Set and send terminal command result.

2.1.1 Macro Definition Documentation

2.1.1.1 APP_COAP_MAX_MSG_LEN

```
#define APP_COAP_MAX_MSG_LEN 1280
```

2.1.2 Function Documentation

2.1.2.1 getNetworkQuality()

```
void getNetworkQuality (  
    int32_t * rsrp,  
    int32_t * rsrq,  
    int32_t * snr)
```

Function gets the cached signal quality from the modem.

Parameters

<i>rsrp</i>	Pointer to store the RSRP value
<i>rsrq</i>	Pointer to store the RSRQ value
<i>snr</i>	Pointer to store the SNR value

2.1.2.2 modemShutdown()

```
void modemShutdown (  
    void )
```

Function to close the socket and power off the modem.

2.1.2.3 sendEventCmdRes()

```
void sendEventCmdRes (  
    int res)
```

Set and send terminal command result.

Parameters

<i>res</i>	Result of the command, 0 if successful, negative value otherwise
------------	--

2.1.2.4 triggerDataPacking()

```
void triggerDataPacking (  
    bool set_include_radio_parms)
```

Enable packer thread to create a CoAP message.

Parameters

<i>set_include_radio_parms</i>	when set, radio data will be added
--------------------------------	------------------------------------

2.1.2.5 triggerMailer()

```
void triggerMailer (  
    bool trigger_radio_pata_packing)
```

Enable mailer thread; first data packet containing radio data can be created.

Parameters

<i>trigger_radio_pata_packing</i>	when set, radio data packet will be added
-----------------------------------	---

2.2 Int_common.h File Reference

Macros

- `#define FIRST_USER_STATUS_BIT 16`
- `#define LAST_USER_STATUS_BIT 31`

Functions

- `void criticalError (void)`
Calls assert and triggers system reset in case of critical error.
- `bool checkBootOkMask (uint32_t mask)`
Returns the flag if the given mask is set in BootOK status.
- `void setUserBootOkMask (uint32_t mask)`
Set user app boot ok mask.
- `void setBootOkBit (uint32_t bit)`
Set appropriate status bit.
- `void resetStatusBit (uint32_t bit)`
Reset appropriate status bit.

2.2.1 Macro Definition Documentation

2.2.1.1 FIRST_USER_STATUS_BIT

```
#define FIRST_USER_STATUS_BIT 16
```

2.2.1.2 LAST_USER_STATUS_BIT

```
#define LAST_USER_STATUS_BIT 31
```

2.2.2 Function Documentation

2.2.2.1 checkBootOkMask()

```
bool checkBootOkMask (
    uint32_t mask)
```

Returns the flag if the given mask is set in BootOK status.

Parameters

<i>mask</i>	The mask of the interested flags
-------------	----------------------------------

Returns

true if all flags are set, false otherwise

2.2.2.2 criticalError()

```
void criticalError (
    void )
```

Calls assert and triggers system reset in case of critical error.

2.2.2.3 resetStatusBit()

```
void resetStatusBit (
    uint32_t bit)
```

Reset appropriate status bit.

Parameters

<i>bit</i>	The number of bit that has to be reset
------------	--

2.2.2.4 setBootOkBit()

```
void setBootOkBit (
    uint32_t bit)
```

Set appropriate status bit.

Parameters

<i>bit</i>	The number of bit that has to be set
------------	--------------------------------------

2.2.2.5 setUserBootOkMask()

```
void setUserBootOkMask (
    uint32_t mask)
```

Set user app boot ok mask.

Parameters

<i>mask</i>	The mask of the interested flags
-------------	----------------------------------

2.3 lmt_proto_handler.h File Reference

Macros

- #define MAX_TRACKS_COUNT 12
- #define MAX_PERIODS_COUNT 3
- #define MAX_COLUMNS_COUNT 50
- #define MAX_TAPE_COUNT 1
- #define MAX_ACTION_PARAMETERS_SIZE 256
- #define I_TAPE 0

Functions

- void dumpMemory (uint8_t *ptr, uint16_t size)
Dumps in HEX the memory region at the given address of the given size.
- bool isDataChanged (void)
Returns flag signalling that data is updated in comparison to the last encoded message.
- void updatePeriod (uint8_t i_tape, uint32_t value)
Function to add a new period to the Periods array of a given Tape. During operation first add the period, then measurement. The duplicate value entries will be not added. The entries without measurements will be overwritten. On overflow it will reset the Periods array. Corrects also the Columns_count if Columns_count >= MAX_COLUMNS_COUNT.
- uint32_t getLastPeriod (uint8_t i_tape)
Returns the last defined period of the Tape instance given by pointer.
- int addColumnToTape (uint8_t i_tape, uint32_t period, int32_t *p_measurements)
Adds a measurements column and its period to the given Tape.
- pb_size_t getTapeRecordsCount (uint8_t i_tape)
Returns actual column count in Tape.
- void rewindTape (uint8_t i_tape)
Resets the Tape keeping the last period entry.
- void restartMeasurements (void)
Clear sensor measurements keeping the last measurement periods.
- bool encodeMessage (void)
Encodes uplink message.
- uint16_t getEncodedMsgLen (void)
Get encoded message length.
- const uint8_t * getEncodedMsgBuffer (void)
Get pointer to encoded message buffer.
- bool isUdpPacketFull (void)
Checks if the UDP packet is full.
- bool decodeMessage (const uint8_t *p_buffer)
Decodes downlink message.

2.3.1 Macro Definition Documentation

2.3.1.1 I_TAPE

```
#define I_TAPE 0
```

2.3.1.2 MAX_ACTION_PARAMETERS_SIZE

```
#define MAX_ACTION_PARAMETERS_SIZE 256
```

2.3.1.3 MAX_COLUMNS_COUNT

```
#define MAX_COLUMNS_COUNT 50
```

2.3.1.4 MAX_PERIODS_COUNT

```
#define MAX_PERIODS_COUNT 3
```

2.3.1.5 MAX_TAPE_COUNT

```
#define MAX_TAPE_COUNT 1
```

2.3.1.6 MAX_TRACKS_COUNT

```
#define MAX_TRACKS_COUNT 12
```

2.3.2 Function Documentation

2.3.2.1 addColumnToTape()

```
int addColumnToTape (  
    uint8_t i_tape,  
    uint32_t period,  
    int32_t * p_measurements)
```

Adds a measurements column and its period to the given Tape.

Parameters

<i>i_tape</i>	the Tape index
<i>period</i>	the period value
<i>p_measurements</i>	pointer to the measurements array in count of MAX_TRACKS_COUNT

Returns

number of remaining empty columns, -EINVAL if *i_tape* is out of range

2.3.2.2 decodeMessage()

```
bool decodeMessage (  
    const uint8_t * p_buffer)
```

Decodes downlink message.

Parameters

<i>p_buffer</i>	pointer to incoming message
-----------------	-----------------------------

Returns

success flag

2.3.2.3 dumpMemory()

```
void dumpMemory (  
    uint8_t * ptr,  
    uint16_t size)
```

Dumps in HEX the memory region at the given address of the given size.

Parameters

<i>*ptr</i>	The starting memory address
<i>size</i>	The size of the memory dump

2.3.2.4 encodeMessage()

```
bool encodeMessage (  
    void )
```

Encodes uplink message.

Returns

success flag

2.3.2.5 getEncodedMsgBuffer()

```
const uint8_t * getEncodedMsgBuffer (  
    void )
```

Get pointer to encoded message buffer.

Returns

Constant pointer to encoded message buffer

2.3.2.6 getEncodedMsgLen()

```
uint16_t getEncodedMsgLen (  
    void )
```

Get encoded message length.

Returns

Current encoded message length

2.3.2.7 getLastPeriod()

```
uint32_t getLastPeriod (  
    uint8_t i_tape)
```

Returns the last defined period of the Tape instance given by pointer.

Parameters

<i>i_tape</i>	the Tape index
---------------	----------------

Returns

the last defined period or 0 if period is not set

2.3.2.8 getTapeRecordsCount()

```
pb_size_t getTapeRecordsCount (
    uint8_t i_tape)
```

Returns actual column count in Tape.

Parameters

<i>i_tape</i>	the Tape index
---------------	----------------

Returns

Column count in Tape

2.3.2.9 isDataChanged()

```
bool isDataChanged (
    void )
```

Returns flag signalling that data is updated in comparison to the last encoded message.

Returns

A flag signalling that data had been changed

2.3.2.10 isUdpPacketFull()

```
bool isUdpPacketFull (
    void )
```

Checks if the UDP packet is full.

Returns

true if full, false otherwise

2.3.2.11 restartMeasurements()

```
void restartMeasurements (
    void )
```

Clear sensor measurements keeping the last measurement periods.

2.3.2.12 rewindTape()

```
void rewindTape (
    uint8_t i_tape)
```

Resets the Tape keeping the last period entry.

Parameters

<i>i_tape</i>	the Tape index
---------------	----------------

2.3.2.13 updatePeriod()

```
void updatePeriod (
    uint8_t i_tape,
    uint32_t value)
```

Function to add a new period to the Periods array of a given Tape. During operation first add the period, then measurement. The duplicate value entries will be not added. The entries withouth measurements will be overwritten. On overflow it will reset the Periods array. Corrects also the Columns_count if Columns_count >= MAX_COLUMNS_COUNT.

Parameters

<i>i_tape</i>	the Tape index
<i>value</i>	the period value

2.4 lmt_sdk_api.h File Reference**Functions**

- void lmtInit (void)
Initializes the SDK.
- void setup (void)
User application setup code.
- void loop (void)
User application main control loop.

2.4.1 Function Documentation**2.4.1.1 lmtInit()**

```
void lmtInit (
    void )
```

Initializes the SDK.

2.4.1.2 loop()

```
void loop (
    void )
```

User application main control loop.

2.4.1.3 setup()

```
void setup (  
    void )
```

User application setup code.

2.5 Imt_settings.h File Reference

Enumerations

- enum logLevel { LOG_ERRORS = 0 , LOG_WARNINGS , LOG_INFORMATIVE }
Log level options for system logging.

Functions

- int setLogFileMaxSize (int32_t size)
Set the maximum size of the log file.
- int setUplinkTimeout (uint16_t timeout)
Set the device uplink timeout.
- int setNoPsmUplinkTimeout (uint16_t timeout)
Set the device uplink timeout when PSM not available.
- int setResendPacketInitialTimeout (uint8_t timeout)
Set the initial timeout for resending packets.
- int setMaxResendTimeout (uint8_t timeout)
Set the maximum timeout for resending packets.
- int setMaxResendAttempts (uint8_t attempts)
Set the maximum number of resend attempts.
- int setLogRotationFrequency (uint8_t frequency)
Set the frequency of log rotation checks.
- int setResponseWaitTimeout (uint8_t timeout)
Set the CoAP response wait timeout.
- int setFileUIRetries (uint8_t retries)
Set the number of retries for file uploads.
- int setNumOfLogFiles (uint8_t file_count)
Set the maximum number of log files on flash before starting file rotation.
- int setLogLevel (logLevel level)
Set the log level for the application.
- int32_t getLogFileMaxSize (void)
Get the maximum size of the log file.
- uint16_t setUplinkTimeout (void)
Get the device uplink timeout.
- uint16_t getNoPsmUplinkTimeout (void)
Get the device uplink timeout when no PSM available.
- uint8_t getResendPacketInitialTimeout (void)
Get the initial timeout for resending packets.
- uint8_t getMaxResendTimeout (void)
Get the maximum timeout for resending packets.
- uint8_t getMaxResendAttempts (void)

- Get the maximum number of resend attempts.*
 - uint8_t getLogRotationFrequency (void)
- Get the frequency of log rotation checks.*
 - uint8_t getResponseWaitTimeout (void)
- Get the CoAP response wait timeout.*
 - uint8_t getFileUIRetries (void)
- Get the number of retries for file uploads.*
 - uint8_t getNumOfLogFiles (void)
- Get the maximum number of log file count in flash.*
 - logLevel getLogLevel (void)
- Get the log level for the application.*

2.5.1 Enumeration Type Documentation

2.5.1.1 logLevel

```
enum logLevel
```

Log level options for system logging.

This enum defines the available log levels for controlling the verbosity of saved system logs.

Enumerator

LOG_ERRORS	
LOG_WARNINGS	
LOG_INFORMATIVE	

```
00013 {
00014     LOG_ERRORS = 0,
00015     LOG_WARNINGS,
00016     LOG_INFORMATIVE
00017 } logLevel;
```

2.5.2 Function Documentation

2.5.2.1 getFileUIRetries()

```
uint8_t getFileUlRetries (
    void )
```

Get the number of retries for file uploads.

Returns

Number of retries for file uploads.

2.5.2.2 getLogFileMaxSize()

```
int32_t getLogFileMaxSize (  
    void )
```

Get the maximum size of the log file.

Returns

Maximum size of the log file in bytes.

2.5.2.3 getLogLevel()

```
logLevel getLogLevel (  
    void )
```

Get the log level for the application.

Returns

Log level for the application.

2.5.2.4 getLogRotationFrequency()

```
uint8_t getLogRotationFrequency (  
    void )
```

Get the frequency of log rotation checks.

Returns

Frequency of log rotation checks in wakeup cycles.

2.5.2.5 getMaxResendAttempts()

```
uint8_t getMaxResendAttempts (  
    void )
```

Get the maximum number of resend attempts.

Returns

Maximum number of resend attempts before the socket is closed.

2.5.2.6 getMaxResendTimeout()

```
uint8_t getMaxResendTimeout (
    void )
```

Get the maximum timeout for resending packets.

Returns

Maximum timeout for resending packets in hours.

2.5.2.7 getNoPsmUplinkTimeout()

```
uint16_t getNoPsmUplinkTimeout (
    void )
```

Get the device uplink timeout when no PSM available.

Returns

Device uplink timeout when PSM not available in hours.

2.5.2.8 getNumOfLogFiles()

```
uint8_t getNumOfLogFiles (
    void )
```

Get the maximum number of log file count in flash.

Returns

Number of maximum count of log files stored in flash.

2.5.2.9 getResendPacketInitialTimeout()

```
uint8_t getResendPacketInitialTimeout (
    void )
```

Get the initial timeout for resending packets.

Returns

Initial timeout for resending packets in minutes.

2.5.2.10 `getResponseWaitTimeout()`

```
uint8_t getResponseWaitTimeout (
    void )
```

Get the CoAP response wait timeout.

Returns

Timeout for waiting for CoAP response in seconds.

2.5.2.11 `getUplinkTimeout()`

```
uint16_t getUplinkTimeout (
    void )
```

Get the device uplink timeout.

Returns

Device uplink timeout in minutes.

2.5.2.12 `setFileUIRetries()`

```
int setFileUIRetries (
    uint8_t retries)
```

Set the number of retries for file uploads.

Parameters

<i>retries</i>	Number of retries for file uploads. $1 \leq \text{retries} \leq 10$.
----------------	---

Returns

0 on success, -EINVAL otherwise.

2.5.2.13 `setLogFileMaxSize()`

```
int setLogFileMaxSize (
    int32_t size)
```

Set the maximum size of the log file.

Parameters

<i>size</i>	Maximum size of the log file in bytes. $1024 \text{ (KB)} \leq \text{size} \leq 1048576 \text{ (1MB)}$.
-------------	--

Returns

0 on success, -EINVAL otherwise.

2.5.2.14 `setLogLevel()`

```
int setLogLevel (
    logLevel level)
```

Set the log level for the application.

Parameters

<i>level</i>	Log level for the application. LOG_ERRORS, LOG_WARNINGS, LOG_INFORMATIVE.
--------------	---

Returns

0 on success, -EINVAL otherwise.

2.5.2.15 setLogRotationFrequency()

```
int setLogRotationFrequency (  
    uint8_t frequency)
```

Set the frequency of log rotation checks.

Parameters

<i>frequency</i>	Frequency of log rotation checks in wakeup cycles. $1 \leq \text{frequency} \leq 50$.
------------------	--

Returns

0 on success, -EINVAL otherwise.

2.5.2.16 setMaxResendAttempts()

```
int setMaxResendAttempts (  
    uint8_t attempts)
```

Set the maximum number of resend attempts.

Parameters

<i>attempts</i>	Maximum number of resend attempts before the socket is closed. $1 \leq \text{attempts} \leq 10$.
-----------------	---

Returns

0 on success, -EINVAL otherwise.

2.5.2.17 setMaxResendTimeout()

```
int setMaxResendTimeout (  
    uint8_t timeout)
```

Set the maximum timeout for resending packets.

Parameters

<i>timeout</i>	Maximum timeout for resending packets in hours. 1 (h) <= timeout <= 24 (1d).
----------------	--

Returns

0 on success, -EINVAL otherwise.

2.5.2.18 setNoPsmUplinkTimeout()

```
int setNoPsmUplinkTimeout (  
    uint16_t timeout)
```

Set the device uplink timeout when PSM not available.

Parameters

<i>timeout</i>	Device uplink timeout in hours 1 <= timeout <= 24 (hours).
----------------	--

Returns

0 on success, -EINVAL otherwise.

2.5.2.19 setNumOfLogFiles()

```
int setNumOfLogFiles (  
    uint8_t file_count)
```

Set the maximum number of log files on flash before starting file rotation.

Parameters

<i>file_count</i>	Number of log files on flash. 1 <= file_count <= 20.
-------------------	--

Returns

0 on success, -EINVAL otherwise.

2.5.2.20 setResendPacketInitialTimeout()

```
int setResendPacketInitialTimeout (  
    uint8_t timeout)
```

Set the initial timeout for resending packets.

Parameters

<i>timeout</i>	Initial timeout for resending packets in minutes. 1 (min) <= timeout <= 60 (1h).
----------------	--

Returns

0 on success, -EINVAL otherwise.

2.5.2.21 setResponseWaitTimeout()

```
int setResponseWaitTimeout (
    uint8_t timeout)
```

Set the CoAP response wait timeout.

Parameters

<i>timeout</i>	Timeout for waiting for CoAP response in seconds. 1 <= timeout <= 60.
----------------	---

Returns

0 on success, -EINVAL otherwise.

2.5.2.22 setUplinkTimeout()

```
int setUplinkTimeout (
    uint16_t timeout)
```

Set the device uplink timeout.

Parameters

<i>timeout</i>	Device uplink timeout in minutes. 5 (min) <= timeout <= 1440 (24h).
----------------	---

Returns

0 on success, -EINVAL otherwise.

2.6 lmt_som_event_emitter.h File Reference**Enumerations**

- enum som_event_t {
 EVENT_MAILER_INIT_OK , EVENT_PACKER_INIT_OK , EVENT_LOGGER_INIT_OK ,
 EVENT_DEVICE_INIT_OK , EVENT_UL_START , EVENT_UL_OK ,
 EVENT_UL_FAIL , EVENT_UL_RETRY , EVENT_UL_QUEUE_FULL ,
 EVENT_LOG_ERROR , EVENT_LOG_WARNING , EVENT_LOG_INFO ,
 EVENT_TERMINAL_CMD }

Events emitted by the LMT SOM library.

Functions

- void somEventHandler (som_event_t event, void *data, uint8_t data_len)

Centralized event handler for the library. This is a weak function that can be overridden by the user application.

2.6.1 Enumeration Type Documentation

2.6.1.1 som_event_t

```
enum som_event_t
```

Events emitted by the LMT SOM library.

This enumeration defines all possible events that can be emitted by the library to notify the application about various system and application-level occurrences, such as initialization, uplink status, logging, and terminal commands.

Enumerator

EVENT_MAILER_INIT_OK	Mailer module initialized successfully.
EVENT_PACKER_INIT_OK	Packer module initialized successfully.
EVENT_LOGGER_INIT_OK	Logger module initialized successfully.
EVENT_DEVICE_INIT_OK	Device initialization completed successfully.
EVENT_UL_START	Uplink process started.
EVENT_UL_OK	Uplink completed successfully.
EVENT_UL_FAIL	Uplink failed.
EVENT_UL_RETRY	Uplink retry is being attempted.
EVENT_UL_QUEUE_FULL	Uplink queue is full.
EVENT_LOG_ERROR	An error log event occurred.
EVENT_LOG_WARNING	A warning log event occurred.
EVENT_LOG_INFO	An informational log event occurred.
EVENT_TERMINAL_CMD	A terminal command event occurred.

```
00014 {
00015     EVENT_MAILER_INIT_OK,
00016     EVENT_PACKER_INIT_OK,
00017     EVENT_LOGGER_INIT_OK,
00018     EVENT_DEVICE_INIT_OK,
00019     EVENT_UL_START,
00020     EVENT_UL_OK,
00021     EVENT_UL_FAIL,
00022     EVENT_UL_RETRY,
00023     EVENT_UL_QUEUE_FULL,
00024     EVENT_LOG_ERROR,
00025     EVENT_LOG_WARNING,
00026     EVENT_LOG_INFO,
00027     EVENT_TERMINAL_CMD
00028 } som_event_t;
```

2.6.2 Function Documentation

2.6.2.1 somEventHandler()

```
void somEventHandler (
    som_event_t event,
    void * data,
    uint8_t data_len)
```

Centralized event handler for the library. This is a weak function that can be overridden by the user application.

Parameters

<i>event</i>	The event type.
<i>data</i>	Optional data associated with the event (can be NULL).
<i>data_len</i>	Length of the data (if applicable).

2.7 Imt_storage_manager.h File Reference

Functions

- `int logError (char *text, int code)`
Writes error to app.log file.
- `int logWarning (char *text)`
Writes warning to app.log file.
- `void logStringHex (uint8_t *payload, uint8_t message_length)`
Logs the given buffer as hex string.
- `int logInfoFormatted (char *text,...)`
Variadic function for writing formatted info to app.log file.
- `int logInfo (char *text)`
Function for writing informative string to app.log file. Creating two separate functions because most of the time we don't need to format the string.
- `int saveSettings (void)`
Writes current device settings in JSON format to settings.txt file.
- `void settingsFileRead (void)`
Schedules settings.txt read.
- `int checkFwVersion (void)`
Reads and compares FW version in primary image and in NVS. In case the primary image is newer, its saved in NVS and setEventFwUpgraded() is called.
- `int eraseFlash (void)`
Wrapper function for filesystem flashErase()

2.7.1 Function Documentation

2.7.1.1 checkFwVersion()

```
int checkFwVersion (
    void )
```

Reads and compares FW version in primary image and in NVS. In case the primary image is newer, its saved in NVS and setEventFwUpgraded() is called.

Returns

1 if primary image version greater, 0 if the same, negative value on fail

2.7.1.2 eraseFlash()

```
int eraseFlash (
    void )
```

Wrapper function for filesystem flashErase()

Returns

0 on success, negative error code on fail

2.7.1.3 logError()

```
int logError (
    char * text,
    int code)
```

Writes error to app.log file.

Parameters

<i>text</i>	Error string.
<i>code</i>	Error code.

Returns

0 on success, negative value on fail

2.7.1.4 logInfo()

```
int logInfo (
    char * text)
```

Function for writing informative string to app.log file. Creating two separate functions because most of the time we don't need to format the string.

Parameters

<i>text</i>	info message string.
-------------	----------------------

Returns

0 on success, negative value on fail

2.7.1.5 logInfoFormatted()

```
int logInfoFormatted (
    char * text,
    ...)
```

Variadic function for writing formatted info to app.log file.

Parameters

<i>text</i>	info message string.
-------------	----------------------

Returns

0 on success, negative value on fail

2.7.1.6 logStringHex()

```
void logStringHex (
    uint8_t * payload,
    uint8_t message_length)
```

Logs the given buffer as hex string.

Parameters

<i>payload</i>	pointer to the buffer
<i>message_length</i>	buffer size

2.7.1.7 logWarning()

```
int logWarning (
    char * text)
```

Writes warning to app.log file.

Parameters

<i>text</i>	Warning string.
-------------	-----------------

Returns

0 on success, negative value on fail

2.7.1.8 saveSettings()

```
int saveSettings (
    void )
```

Writes current device settings in JSON format to settings.txt file.

Returns

0 on success, negative value on fail

2.7.1.9 settingsFileRead()

```
void settingsFileRead (
    void )
```

Schedules settings.txt read.

Index

- addColumnToTape
 - lmt_proto_handler.h, 7
- APP_COAP_MAX_MSG_LEN
 - lmt_coap_manager.h, 2
- checkBootOkMask
 - lmt_common.h, 4
- checkFwVersion
 - lmt_storage_manager.h, 20
- criticalError
 - lmt_common.h, 4
- decodeMessage
 - lmt_proto_handler.h, 7
- dumpMemory
 - lmt_proto_handler.h, 7
- encodeMessage
 - lmt_proto_handler.h, 8
- eraseFlash
 - lmt_storage_manager.h, 20
- EVENT_DEVICE_INIT_OK
 - lmt_som_event_emitter.h, 19
- EVENT_LOG_ERROR
 - lmt_som_event_emitter.h, 19
- EVENT_LOG_INFO
 - lmt_som_event_emitter.h, 19
- EVENT_LOG_WARNING
 - lmt_som_event_emitter.h, 19
- EVENT_LOGGER_INIT_OK
 - lmt_som_event_emitter.h, 19
- EVENT_MAILER_INIT_OK
 - lmt_som_event_emitter.h, 19
- EVENT_PACKER_INIT_OK
 - lmt_som_event_emitter.h, 19
- EVENT_TERMINAL_CMD
 - lmt_som_event_emitter.h, 19
- EVENT_UL_FAIL
 - lmt_som_event_emitter.h, 19
- EVENT_UL_OK
 - lmt_som_event_emitter.h, 19
- EVENT_UL_QUEUE_FULL
 - lmt_som_event_emitter.h, 19
- EVENT_UL_RETRY
 - lmt_som_event_emitter.h, 19
- EVENT_UL_START
 - lmt_som_event_emitter.h, 19
- FIRST_USER_STATUS_BIT
 - lmt_common.h, 4
- getEncodedMsgBuffer
 - lmt_proto_handler.h, 8
- getEncodedMsgLen
 - lmt_proto_handler.h, 8
- getFileUIRetries
 - lmt_settings.h, 12
- getLastPeriod
 - lmt_proto_handler.h, 8
- getLogFileMaxSize
 - lmt_settings.h, 12
- getLogLevel
 - lmt_settings.h, 13
- getLogRotationFrequency
 - lmt_settings.h, 13
- getMaxResendAttempts
 - lmt_settings.h, 13
- getMaxResendTimeout
 - lmt_settings.h, 13
- getNetworkQuality
 - lmt_coap_manager.h, 2
- getNoPsmUplinkTimeout
 - lmt_settings.h, 14
- getNumOfLogFiles
 - lmt_settings.h, 14
- getResendPacketInitialTimeout
 - lmt_settings.h, 14
- getResponseWaitTimeout
 - lmt_settings.h, 14
- getTapeRecordsCount
 - lmt_proto_handler.h, 9
- getUplinkTimeout
 - lmt_settings.h, 15
- I_TAPE
 - lmt_proto_handler.h, 6
- isDataChanged
 - lmt_proto_handler.h, 9
- isUdpPacketFull
 - lmt_proto_handler.h, 9
- LAST_USER_STATUS_BIT
 - lmt_common.h, 4
- lmt_coap_manager.h, 2
 - APP_COAP_MAX_MSG_LEN, 2
 - getNetworkQuality, 2
 - modemShutdown, 3
 - sendEventCmdRes, 3
 - triggerDataPacking, 3
 - triggerMailer, 3
- lmt_common.h, 4

- checkBootOkMask, 4
- criticalError, 4
- FIRST_USER_STATUS_BIT, 4
- LAST_USER_STATUS_BIT, 4
- resetStatusBit, 5
- setBootOkBit, 5
- setUserBootOkMask, 5
- lmt_proto_handler.h, 5
 - addColumnToTape, 7
 - decodeMessage, 7
 - dumpMemory, 7
 - encodeMessage, 8
 - getEncodedMsgBuffer, 8
 - getEncodedMsgLen, 8
 - getLastPeriod, 8
 - getTapeRecordsCount, 9
 - I_TAPE, 6
 - isDataChanged, 9
 - isUdpPacketFull, 9
 - MAX_ACTION_PARAMETERS_SIZE, 6
 - MAX_COLUMNS_COUNT, 6
 - MAX_PERIODS_COUNT, 6
 - MAX_TAPE_COUNT, 7
 - MAX_TRACKS_COUNT, 7
 - restartMeasurements, 9
 - rewindTape, 9
 - updatePeriod, 10
- lmt_sdk_api.h, 10
 - lmtInit, 10
 - loop, 10
 - setup, 10
- lmt_settings.h, 11
 - getFileUIRetries, 12
 - getLogFileMaxSize, 12
 - getLogLevel, 13
 - getLogRotationFrequency, 13
 - getMaxResendAttempts, 13
 - getMaxResendTimeout, 13
 - getNoPsmUplinkTimeout, 14
 - getNumOfLogFiles, 14
 - getResendPacketInitialTimeout, 14
 - getResponseWaitTimeout, 14
 - getUplinkTimeout, 15
 - LOG_ERRORS, 12
 - LOG_INFORMATIVE, 12
 - LOG_WARNINGS, 12
 - logLevel, 12
 - setFileUIRetries, 15
 - setLogFileMaxSize, 15
 - setLogLevel, 15
 - setLogRotationFrequency, 16
 - setMaxResendAttempts, 16
 - setMaxResendTimeout, 16
 - setNoPsmUplinkTimeout, 17
 - setNumOfLogFiles, 17
 - setResendPacketInitialTimeout, 17
 - setResponseWaitTimeout, 18
 - setUplinkTimeout, 18
- lmt_som_event_emitter.h, 18
 - EVENT_DEVICE_INIT_OK, 19
 - EVENT_LOG_ERROR, 19
 - EVENT_LOG_INFO, 19
 - EVENT_LOG_WARNING, 19
 - EVENT_LOGGER_INIT_OK, 19
 - EVENT_MAILER_INIT_OK, 19
 - EVENT_PACKER_INIT_OK, 19
 - EVENT_TERMINAL_CMD, 19
 - EVENT_UL_FAIL, 19
 - EVENT_UL_OK, 19
 - EVENT_UL_QUEUE_FULL, 19
 - EVENT_UL_RETRY, 19
 - EVENT_UL_START, 19
 - som_event_t, 19
 - somEventHandler, 19
- lmt_storage_manager.h, 20
 - checkFwVersion, 20
 - eraseFlash, 20
 - logError, 21
 - logInfo, 21
 - logInfoFormatted, 21
 - logStringHex, 22
 - logWarning, 22
 - saveSettings, 22
 - settingsFileRead, 22
- lmtInit
 - lmt_sdk_api.h, 10
- LOG_ERRORS
 - lmt_settings.h, 12
- LOG_INFORMATIVE
 - lmt_settings.h, 12
- LOG_WARNINGS
 - lmt_settings.h, 12
- logError
 - lmt_storage_manager.h, 21
- logInfo
 - lmt_storage_manager.h, 21
- logInfoFormatted
 - lmt_storage_manager.h, 21
- logLevel
 - lmt_settings.h, 12
- logStringHex
 - lmt_storage_manager.h, 22
- logWarning
 - lmt_storage_manager.h, 22
- loop
 - lmt_sdk_api.h, 10
- MAX_ACTION_PARAMETERS_SIZE
 - lmt_proto_handler.h, 6
- MAX_COLUMNS_COUNT
 - lmt_proto_handler.h, 6
- MAX_PERIODS_COUNT
 - lmt_proto_handler.h, 6
- MAX_TAPE_COUNT
 - lmt_proto_handler.h, 7
- MAX_TRACKS_COUNT
 - lmt_proto_handler.h, 7

modemShutdown
 lmt_coap_manager.h, 3

resetStatusBit
 lmt_common.h, 5

restartMeasurements
 lmt_proto_handler.h, 9

rewindTape
 lmt_proto_handler.h, 9

saveSettings
 lmt_storage_manager.h, 22

sendEventCmdRes
 lmt_coap_manager.h, 3

setBootOkBit
 lmt_common.h, 5

setFileUIRetries
 lmt_settings.h, 15

setLogFileMaxSize
 lmt_settings.h, 15

setLogLevel
 lmt_settings.h, 15

setLogRotationFrequency
 lmt_settings.h, 16

setMaxResendAttempts
 lmt_settings.h, 16

setMaxResendTimeout
 lmt_settings.h, 16

setNoPsmUplinkTimeout
 lmt_settings.h, 17

setNumOfLogFiles
 lmt_settings.h, 17

setResendPacketInitialTimeout
 lmt_settings.h, 17

setResponseWaitTimeout
 lmt_settings.h, 18

settingsFileRead
 lmt_storage_manager.h, 22

setup
 lmt_sdk_api.h, 10

setUplinkTimeout
 lmt_settings.h, 18

setUserBootOkMask
 lmt_common.h, 5

som_event_t
 lmt_som_event_emitter.h, 19

somEventHandler
 lmt_som_event_emitter.h, 19

triggerDataPacking
 lmt_coap_manager.h, 3

triggerMailer
 lmt_coap_manager.h, 3

updatePeriod
 lmt_proto_handler.h, 10