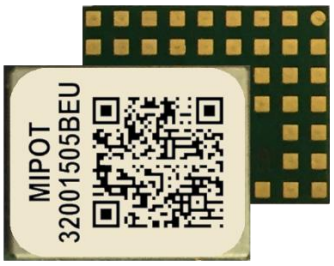
Wireless Protocol Modules MiP Series 32001505CEU

Mipot Network LoRa™ Node

Command Reference



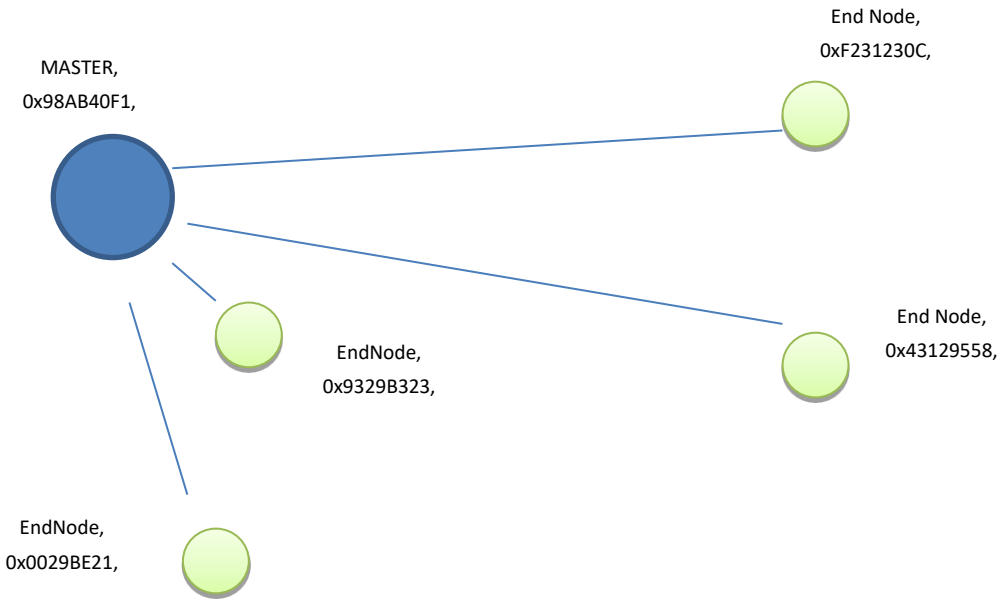
Description

This document provides list of commands that the 32001505CEU implements and the description of their use.

Contents

1. Network description

MIPOT LONG RAGE NETWORK is an **asynchronous network** based on physical layer of LoRa Technology. Network topology is a **star** where, in order to minimize power consumption, **end nodes** work in **duty cycle mode**:



All messages are cyphered with **AES128 encryption** algorithm ensuring confidential authentication and integrity during the exchanging of data payload.

2. Communication interface

The **32001505CEU** is a host based module needing an external microcontroller to configure and operate it. The communication interface between the module and the external microcontroller can be one among available SPI, I^2C or UART, depending on user application needs.

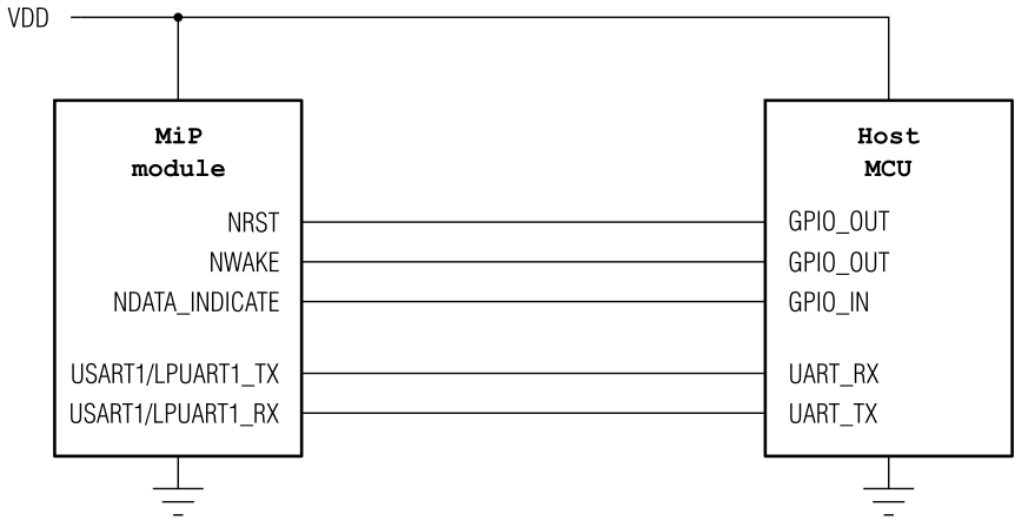
2.1. SPI/I^2C/UART interface

SPI/I^2C/UART interface allows Host both to configure the module and to exchange LoRa radio frame data messages.

One among available SPI, I^2C or UART interfaces can be chosen. There is no need to preliminary configure the interface to use. After reset the module listens for messages on each of the available ports and automatically elects as communication interface the channel where the first valid message is received.

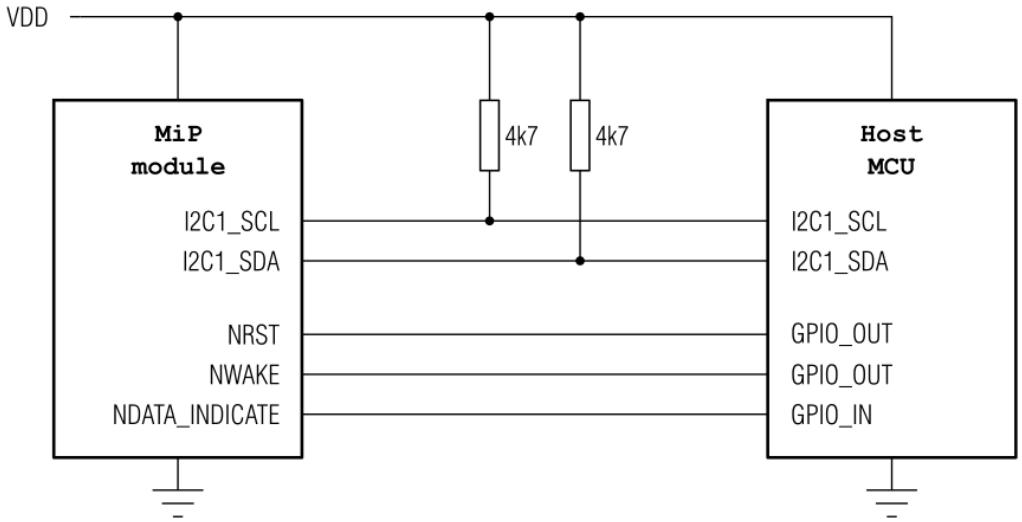
2.2. Connection with USART1 or LPUART1

The serial port uses the DTE terminology and direction



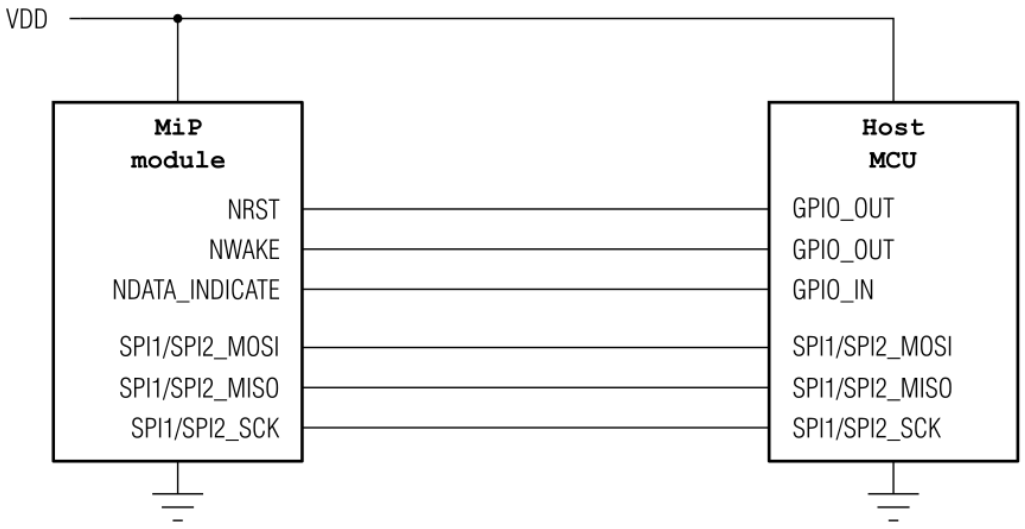
2.3. Connection with I2C1

The I2C interface requires an external pullup on the communication lines.



2.4. Connection with SPI1 or SPI2

The NWAKE pin can be used as the SPI's NSS signal.



3. Communication interface: SPI/UART

SPI/UART interface allows Host both to configure the module and to exchange LoRa radio frame data messages.

3.1. Byte Order

Multiple byte values are transmitted in little endian order with least significant byte first (LSB).

3.2. Message Structure

The structure of the messages is the following:

HEADER	CMD	LENGTH	PAYLOAD (n Bytes)	CHECKSUM

Where:

HEADER	=	0xAA
CMD	=	Command code to the module, see the following table
LENGTH	=	Payload length
Checksum	=	2's complement on one byte of the sum of all preceding bytes

Each command from the host invokes an answer from the module in the same format.

The answer to the host has the CMD field equal to host request OR 0x80.

3.3. Message Types

There are three types of messages:

Commands: sent from the host to the module to request an information or an action.

Replies: sent from the module to the host as direct reply to a command, their command code is equal to the host request OR 0x80.

Indications: messages sent from the module to the host that are sent without prior action from the host, triggered by events on the radio interface. (e.g.: a received transmission).

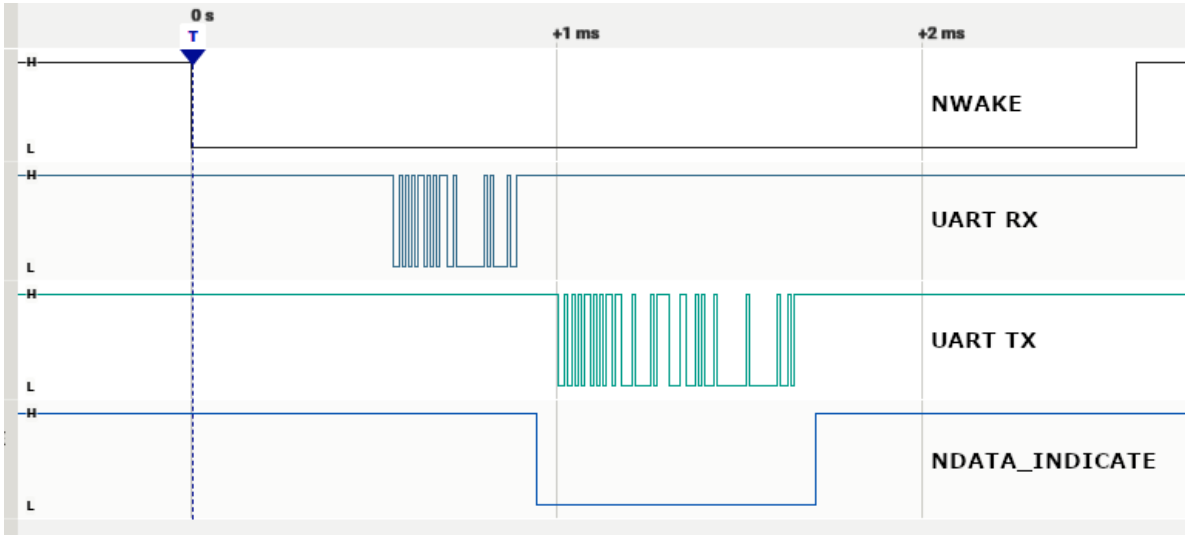
3.4. Detailed Signal Flow

The module enters sleep mode as soon as possible therefore, before initiating a UART session, the host shall wake it up by setting the NWAKE pin LOW and then setting it back HIGH at the end of the session.

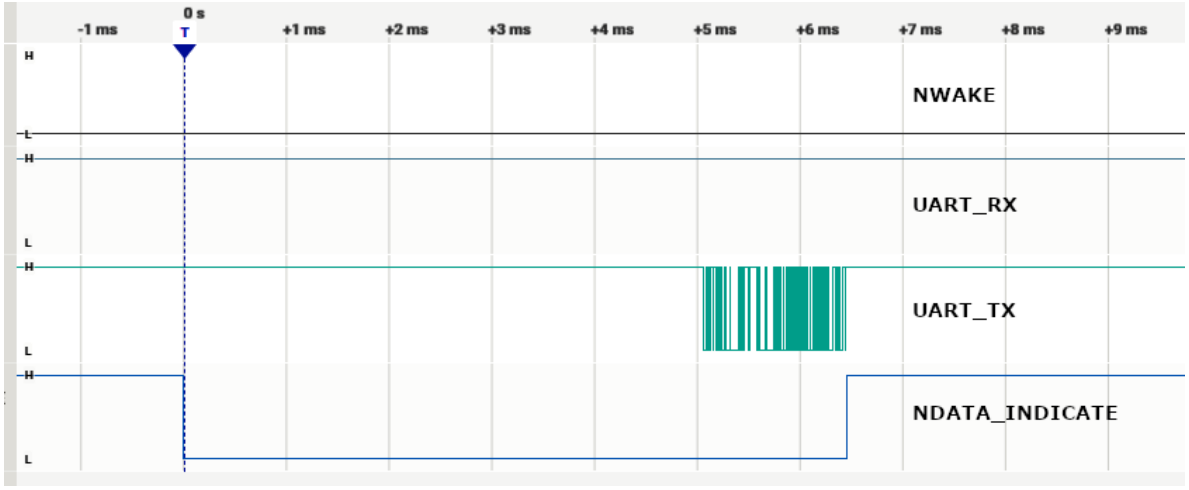
When the module receives a valid command and the checksum is correct, the module sets NDATA_INDICATE LOW, transmits the answer through UART TX pin and then sets NDATA_INDICATE HIGH.

When transferring a received radio frame to the host microcontroller, the module set NDATA_INDICATE LOW, wait for DATA_INDICATE_TIMEOUT expiration, and then send the message on UART TX pin. DATA_INDICATE_TIMEOUT represents the time (in ms) between the instant when NDATA_INDICATE pin goes LOW and the start of transmission on UART TX pin.

3.5. Example of UART command session (Host -> Module):



3.6. Example of UART RX indicate message session (DATA_INDICATE_TIMEOUT = 5ms) (Module -> Host):



4. Command Set Description

List of the implemented command:

Command (CMD)	Value	Description
RESET_CMD	0x30	Module software reset
FACTORY_RESET_CMD	0x31	Restore EEPROM to factory default values
EEPROM_WRITE_CMD	0x32	Write EEPROM parameter
EEPROM_READ_CMD	0x33	Read EEPROM parameter
GET_FW_VERSION_CMD	0x34	Get firmware version
GET_SERIALNO_CMD	0x35	Get serial number
ENABLE_PAIRING_CMD	0x40	Enable/Disable pairing command (Only for MASTER)
DEVICE_PAIRING_IND	0x41	MASTER pairing indicate (Only for MASTER)
GET_NETWORK_TABLE_SIZE_CMD	0x42	Get MASTER Network Table Size (Only for MASTER)
GET_NETWORK_TABLE_ROW_CMD	0x43	Get MASTER Network Table Row (Only for MASTER)
DEL_EN_DEVICE_CMD	0x44	Delete END NODE from MASTER Network Table (Only for MASTER)
DEL_ALL_EN_DEVICE_CMD	0x45	Delete whole Network Table (Only for MASTER)
PAIRING_REQ_CMD	0x48	Pairing request (Only for END NODE)
PAIRING_CONFIRM_IND	0x49	Indication of Activation (Only for END NODE)
GET_ACTIVATION_STATUS_CMD	0x4A	Get Activation Status Command (Only for END NODE)
TX_MSG_CMD	0x50	Transmission of radio message
TX_MSG_IND	0x52	Indication of radio message transmission
RX_MSG_IND	0x53	Indicate radio message reception
LINK_CHECK_REQ_CMD	0x56	Request for a link Check
LINK_CHECK_ANS_IND	0x57	Link Check Indication
SET_APP_KEY_CMD	0x58	Write EEPROM parameter Application encryption key
TX_SESSION_ABORT_IND	0x59	Indicate the abort of current communication session (Only for END NODE)

4.1. RESET_CMD (0x30)

This command performs a module Reset.

When a valid reset request is received, the module replies immediately to the host microcontroller.

All communication interfaces will be disabled during the reset procedure.

Host: 0xAA, 0x30, 0x00, 0x26

Reply: 0xAA, 0xB0, 0x00, 0xA6

4.2. FACTORY_RESET_CMD (0x31)

This command restores EEPROM factory default values.

Host: 0xAA, 0x31, 0x00, 0x25

Reply: 0xAA, 0xB1, 0x01, Status, checksum

Status: 0x00 = Success

A value different from 0: error

4.3. EEPROM_WRITE_CMD (0x32)

This command performs an EEPROM data write. For Addresses and Data values see “Module Configuration” section.

Host: 0xAA, 0x32, Length, Start Address, <Data>, checksum

Reply: 0xAA, 0xB2, 0x01, EEWriteStatus, checksum

EEWriteStatus:

0x00 = Success

0x01 = Invalid address

Note: Data outside allowed range will not be stored in EEPROM and the current value will not be modified. If the variable to be updated has the same value of the new one then the EEPROM will not be updated in order to minimize memory write cycles.

4.4. EEPROM_READ_CMD (0x33)

This command reads EEPROM data. For Addresses and Data values see “Module Configuration” section.

Host: 0xAA, 0x33, 0x02, Start Address, Number of bytes, checksum

Reply: 0xAA, 0xB3, Length, Status, Data, checksum

Status: 0x00 = Success, Data contains EEPROM values

0xFF = failure, Data is empty and Length is equal to 1

4.5. GET_FW_VERSION_CMD (0x34)

Get 32-bit firmware version.

Host: 0xAA, 0x34, 0x00, 0x22

Reply: 0xAA, 0xB4, 0x04, FWV0, FWV1, FWV2, FWV3, checksum

FWV0, FWV1, FWV2, FWV3: Firmware version

4.6. GET_SERIALNO_CMD (0x35)

Get unique 32-bit Serial Number.

Host: 0xAA, 0x35, 0x00, 0x21

Reply: 0xAA, 0xB5, 0x04, SN0, SN1, SN2, SN3, checksum

SN0, SN1, SN2, SN3: 32-bit Mipot Serial Number.

4.7. ENABLE_PAIRING_CMD (0x40)

This command is accepted only when the module is configured as MASTER.

Enable or disable network the pairing procedure, where END NODEs are discovered by MASTER.

Host: 0xAA, 0x40, 0x01, EN, cks

Reply: 0xAA, 0xC0, 0x00, 0x96

EN: 0x00 = disable pairing procedure

A value different from 0: enable pairing procedure

4.8. DEVICE_PAIRING_IND (0x41)

This command is accepted only when the module is configured as MASTER.

Indicates successful pairing with an END NODE. When MASTER module receives a Radio Join Request then adds the device to its Network Table and then sends this command to host.

Module: 0xAA, 0x41, Len, EN_ID, CUSTOM_PAIRING_PAYLOAD_BYTE, cks

EN_ID: is a 4-bytes array containing Serial Number of the paired device

CUSTOM_PAIRING_PAYLOAD_BYTE:

is a byte defined in EEPROM through EndNodePairingReqPayload parameter. This parameter could be useful, for instance, in host application to declare the type of an END NODE without using extra radio messages.

4.9. GET_NETWORK_TABLE_SIZE_CMD (0x42)

This command is accepted only when the module is configured as MASTER.

Get from MASTER module the size of Network Table. The last row containing a valid data has an index equal to (TABLE_SIZE – 1).

Host: 0xAA, 0x42, 0x00, 0x14

Reply: 0xAA, 0xC2, 1, ROUTING_SIZE, cks

NOTE: The maximum number of END NODES is 255.

4.10. GET_NETWORK_TABLE_ROW_CMD (0x43)

This command is accepted only when the module is configured as MASTER.

Returns a Network Table row from index.

Host: 0xAA, 0x43, 0x01, IDX, cks

Reply: 0xAA, 0xC3, 0x05, <Routing Row>, cks

IDX: is the index of Network Table row and must be between 0 to (TABLE_SIZE – 1).

<Routing Row> has the following information:

- EN_ID (4 Bytes);
- CUSTOM_PAIRING_PAYLOAD_BYTE is the byte received at pairing phase.

All multiple byte fields are LSB first.

4.11. DEL_END_DEVICE_CMD (0x44)

This command is accepted only when the module is configured as MASTER.

Deletes a Network Table row from module Serial Number.

Host: 0xAA, 0x44, 0x04, <EN_ID>, cks

Reply: 0xAA, 0xC4, 0x01, status, cks

Status: 0x00 = Success

0xFF = Failure. Device not found in MASTER Network Table

4.12. DEL_ALL_EN_DEVICE_CMD (0x45)

This command is accepted only when the module is configured as MASTER.

Deletes whole Network Table.

Host: 0xAA, 0x45, 0x00, 0x11

Reply: 0xAA, 0xC5, 0x01, status, cks

Status: 0x00 = Success

0xFF = Failure

4.13. PAIRING_REQ_CMD (0x48)

This command is accepted only when the module is configured as END NODE.

Requests a pairing to a network.

Host: 0xAA, 0x48, 0x00, 0x0E

Reply: 0xAA, 0xC8, 0x01, status, cks

Status: 0x00 = OK

0x01 = BUSY, pairing request not accepted

4.14. PAIRING_CONFIRM_IND (0x49)

This command is accepted only when the module is configured as END NODE.

Indicates the reception of a Pairing Confirm message from MASTER.

Module: 0xAA, 0x49, 0x06, status, MST_ID, IDX, cks

Status: 0x00 = success

A value different from zero means an error occurred and MST_ID and IDX are equal to 0.

MST_ID: MASTER Serial Number (4 bytes). If Status is different from success (0x00) this field is equal to 0x00000000.

IDX: MASTER table index assigned after pairing phase.

4.15. GET_ACTIVATION_STATUS_CMD (0x4A)

This command is accepted only when the module is configured as END NODE.

Gets the module activation status.

Host: 0xAA, 0x4A, 0x00, 0x0C

Reply: 0xAA, 0xCA, 0x05, status, MST_ID, cks

Status: 0x00 = Not activated

0x01 = Paired to a network

MST_ID: MASTER Serial Number (4 bytes)

4.16. TX_MSG_CMD (0x50)

This command performs the transmission of radio frames.

Host: 0xAA, 0x50, Length, Options, <DST_ID>, <MsgPayload>, cks

Reply: 0xAA, 0xD0, 0x01, Status, cks

Options: 0bxxxxxxx0 = Unconfirmed Data Transmission. The message will be transmitted UncofirmedTxNumber times (see Stack parameter table configuration).

0bxxxxxxx1 = Confirmed Data Transmission. As long as an Ack is not received, the message will be transmitted up to ConfirmedTxNumber times (see Stack parameter table configuration).

DST_ID: Destination device Serial Number (4 bytes). This field is used only when the module is configure as MASTER. When the module is configured as END NODE this field will be ignored.

0xFFFFFFFF = Broadcast Message. In this case message will be an Unconfirmed Data message type, independently from Options field.

Other values are used for unicast addressing.

MsgPayload: Data to be transmitted. **Maximum allowed payload size is 26 bytes**.

Status: 0x00 = Success

0x01 = Device busy

0x02 = Device not activated

0x03 = Payload Size Error

NOTE:

The module does not manage automatically duty cycle restrictions. Host application must handle the duty cycle requirements in order to assure compliance with the harmonized standard limits.

The following table shows time-on-air for a single frame as a function of the number of transmitted bytes for an END NODE device:

Number of bytes	Time on Air (ms)
0-10	67
11-26	88

The following table shows time-on-air for a single frame as a function of the number of transmitted bytes for MASTER device:

Number of bytes	Time on Air (ms)
0-10	1155
11-26	1175

4.17. TX_MSG_CONFIRMED_IND (0x51)

This command indicates the end of a confirmed transmission session. An Ack is received or all transmission retries have been carried out.

Module: 0xAA, 0x51, 0x07, Status, <SessionTxTime>, AckReceived, NbRetries, cks

Status: 0x00 = success

A value different from zero means that an error has occurred.

AckReceived: 0 = No Ack received

1 = Ack received

NbRetries: Number of carried out transmissions.

SessionTxTime:

4 bytes. Session transmission time expressed in ms; includes cumulative transmission time for all retries.

4.18. TX_MSG_IND (0x52)

This command indicates the end of an unconfirmed transmission session (All UncofirmedTxNumber retries have been carried out).

Module: 0xAA, 0x52, 0x05, Status, <SessionTxTime>, cks

Status: 0x00 = success

A value different from zero means that an error has occurred.

SessionTxTime:

4 bytes. Session transmission time expressed in ms; includes cumulative transmission time for all retries.

4.19. RX_MSG_IND (0x53)

This command indicates the reception of radio frames.

Module: 0xAA, 0x53, Length, Status, RssiLSB, RssiMSB, SNR, <SRC_ID>, <Payload>, cks

Status: 0x00 = success

Values different from zero are reserved.

RssiLSB/MSB: 16-bit Rssi Value expressed in dBm

SNR: 8-bit Signal-to-Noise Ratio

SRC_ID: Source device Serial Number (4 bytes)

Payload: Data Message

4.20. LINK_CHECK_REQ_CMD (0x56)

This command initiates from END NODE a link check procedure between END NODE and MASTER.

Host: 0xAA, 0x56, 0x03, Power, MessageNumber, MessageTh, cks

Reply: 0xAA, 0xD6, 0x01, status, cks

Power: Defines the power used for the link check procedure expressed in dBm. Allowed range is from 2 to 14dBm. Suggested value is 11dBm.

MessageNumber:
Defines the number of messages to be transmitted. Allowed Range is from 4 to 20. Suggested Value is 5.

MessageTh: Defines pass/fail criteria threshold as the number of received messages. If the number of received messages is >= to MessageTh test succeeded. The suggested value is 4. Minimum value is 1 and maximum Value is Message Number.

Status: 0x00 = Success
0x01 = Device Busy
0x02 = Parameters Out of range

4.21. LINK_CHECK_ANS_IND (0x57)

This command indicates the end of a link check procedure.

Module: 0xAA, 0x57, 0x02, Result, RxMessages, cks

Result: 0x00 = Link test KO
0xFF = Link test OK

RxMessages: Number of received messages.

4.22. SET_APP_KEY_CMD (0x58)

This command performs an EEPROM data write.

Host: 0xAA, 0x58, 0x10, <AppKey>, cks

Reply: 0xAA, 0xD8, 0x00, cks

AppKey: 16 byte in Little Endian Order. Needed for Application encryption customization. This key is used only when AppEnAes parameter is set to 1.

4.23. TX_SESSION_ABORT_IND (0x59)

This command is used only when the module is configured as END NODE.

This command indicates the end of current communication session due to a master confirmed type session.

Module: 0xAA, 0x59, 0x04, <SessionTxTime>, cks

SessionTxTime:

4 bytes. Session transmission time expressed in ms; includes cumulative transmission time for all retries.

5. Module Configuration

Multiple byte values are expressed in little endian order with least significant byte first (LSB).

5.1. Stack Parameters

Parameter	Description	Address	Values Range	Default	Notes
DeviceType	MASTER – END NODE Selection	0x00	0-1	1 = END NODE	0x00 = MASTER 0x01= END NODE
UnconfirmedTxNumber	Define the Number of broadcast consecutive message transmissions	0x01	1-15	3	
ConfirmedTxNumber	Define the Number of transmission retries if ACK is not received	0x02	1-15	3	
EndNodePairingReqPayload	Define the Pairing Request Payload	0x03	0 – 255	0	END NODE Pairing Request Custom Payload (1 byte)
EndNodePairingMstAddress	Define the paired MASTER address	0x04 - 0x07	0x00000000 – 0xFFFFFFFF	0x00000000	This field is meaningful only for END NODE
EndNodeMstTblIdx	Define the END NODE index assigned by MASTER after the pairing phase.	0x08	0x00-0xFF	0x00	This field is meaningful only for END NODE

5.2. Radio Physical Parameters

Parameter	Description	Address	Range	Default	Notes
Power	Power expressed in dBm	0x10	2-14	14	Power expressed in dBm
Frequency	Channel Frequency selection	0x11	0 – 2	2	0 = 868.1MHz 1 = 868.3MHz 2 = 868.5MHz
RSSI_Th	Define the RSSI threshold for channel free assessment. It is an absolute value	0x12	80-110	90	90 => RSSI_Th = -90dBm

5.3. Module Parameters

Parameter	Description	Address	Range	Default	Notes
DATA_INDICATE_TIMEOUT	Timeout in ms	0x80	1-255	5	Expressed in ms
UartBaudrate	Uart baudrate selection	0x81	0 – 4	4	0 = 9600 1 = 19200 2 = 38400 3 = 57600 4 = 115200
AppEnAes	Application AES Key Enable/Disable	0x82	0 – 1	0	0 = Disabled 1 = Enabled

5.4. Internal DATA (Read Only)

Parameter	Description	Notes
SerialNumber0	Byte 0 SN	Serialization at 32 bit
SerialNumber1	Byte 1 SN	
SerialNumber2	Byte 2 SN	
SerialNumber3	Byte 3 SN	
FwVersion0	Byte 0 FW Version	Fw Version
FwVersion1	Byte 1 FW Version	
FwVersion2	Byte 2 FW Version	
FwVersion3	Byte 3 FW Version	

5.5. Application AES KEY (Write Only)

The module implements on-board network AES encryption with an internal key (not accessible to the host microcontroller). If the Host microcontroller needs to customize the encryption at application level, it has to enable this feature by setting AppEnAes parameter to 1 then write AppKey through SET_APP_KEY_CMD.

Parameter	Description	Values Range	Default	Notes
AppKey	16 byte Application Key	0-255 for all 16 bytes	0 for all 16 bytes	Used at application level (Write Only Variable)

6. Examples

This section describes some examples for network configuration and message exchange.

The examples will consider five 32001345 modules with the following serial number:

0x11111111

0x22222222

0x33333333

0x44444444

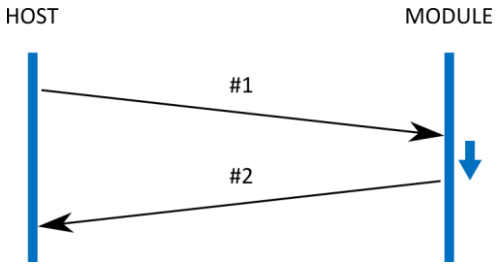
0x55555555

6.1. MASTER NODE CONFIGURATION

All modules are factory configured as End Node (EN); one module must be elected as MASTER (MST) node (the example will use Module 0x55555555 as MST). To do this the host microcontroller has to write the “DeviceType” parameter to 0.

Module 0x55555555:

Host command:	0xAA, 0x32, 0x02, 0x00, 0x00, 0x22	(#1)
Module reply:	0xAA, 0xB2, 0x01, 0x00, 0xA3	(#2)



6.2. END NODES PAIRING PHASE

Now all ENs have to be paired to MST node. First of all, MST node has to be put in pairing mode:

Module 0x55555555:

Host command:	0xAA, 0x40, 0x01, 0x01, 0x14	(#3)
Module Answer:	0xAA, 0xC0, 0x00, 0x96	(#4)

Then each EN, being in the range of MST, should send a pairing request command.

For Modules 0x11111111, 0x22222222, 0x33333333, 0x44444444:

Host command:	0xAA, 0x48, 0x00, 0x0E	(#5)
Module Answer:	0xAA, 0xC8, 0x01, 0x00, 0x8D	(#6)

When the pairing procedure succeeds, an indicate message will be transmitted by both modules on MST side and EN side:

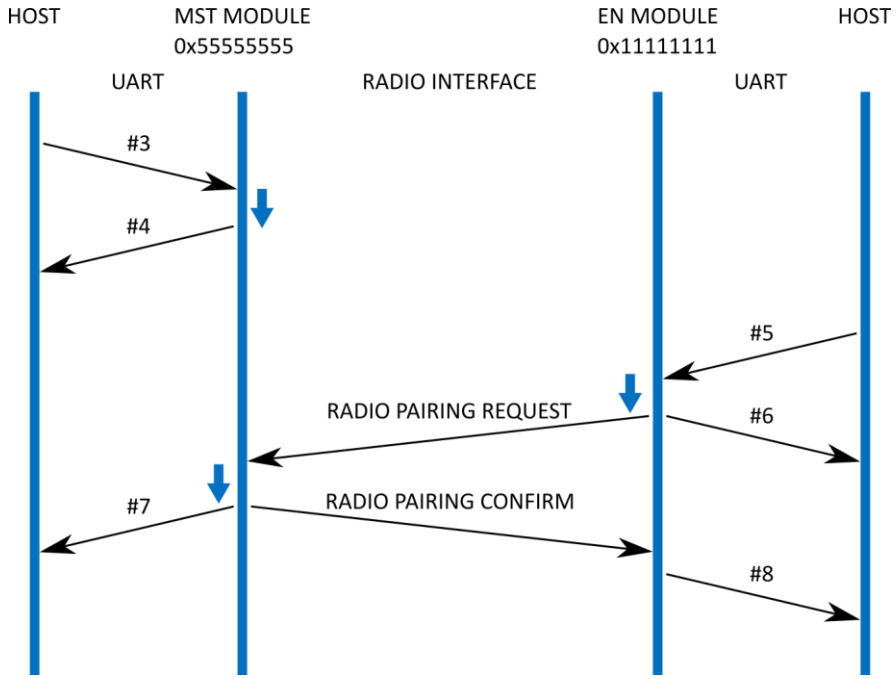
Hereafter the message exchange for EN 0x11111111 successfully paired with MST.

MST Module 0x55555555:

Module Indicate:	0xAA, 0x41, 0x05, 0x11, 0x11, 0x11, 0x11, 0x11, 0x00, 0xCC	(#7)

EN Module 0x11111111:

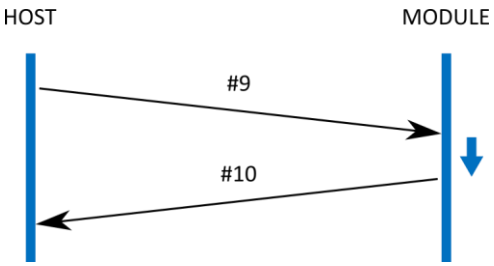
Module Indicate:	0xAA, 0x49, 0x06, 0x00, 0x55, 0x55, 0x55, 0x55, 0x00, 0xB3	(#8)



When the pairing procedure ends, the pairing mode has to be disabled on MST.

Module 0x55555555:

Host command:	0xAA, 0x40, 0x01, 0x00, 0x15	(#9)
Module Answer:	0xAA, 0xC0, 0x00, 0x96	(#10)



6.3. END NODE INSTALLATION AND LINK CHECK

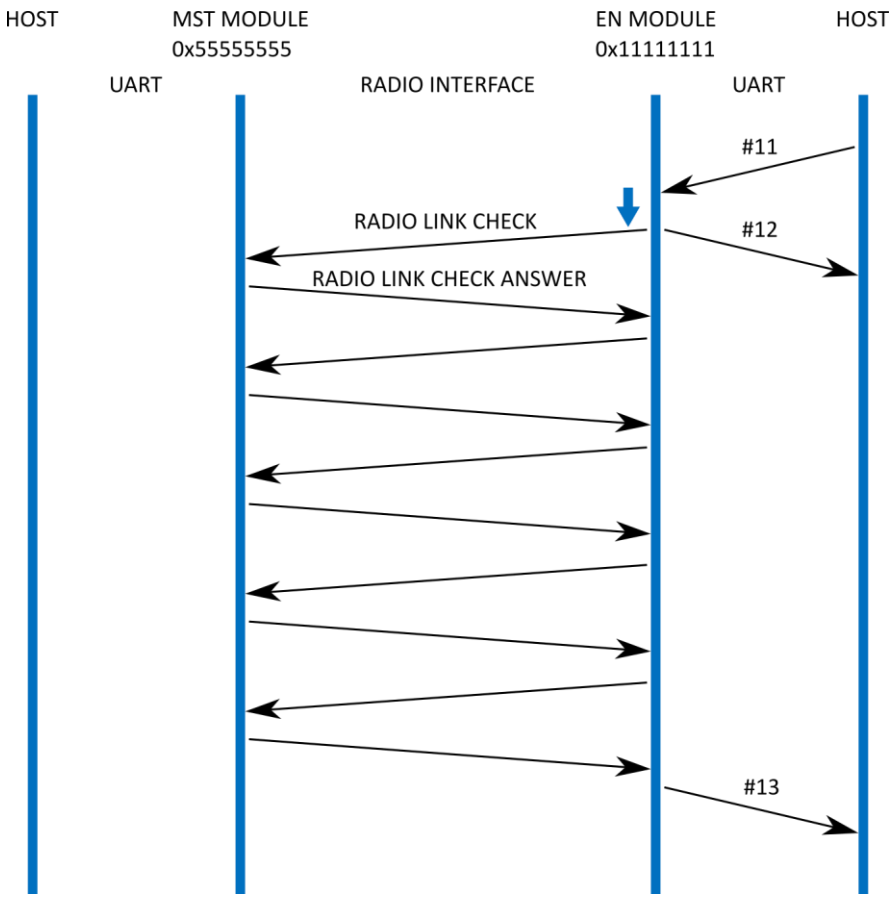
After pairing completion, all ENs can be installed in the application area. In order to check the link reliability for each EN, a **LINK_CHECK_REQ_CMD (0x56)** should be used. In the following example, five messages at 11dBm power setting (3dB less than default power) are sent. If an EN receives a number of messages >= 4 test is ok.

For Modules 0x11111111, 0x22222222, 0x33333333, 0x44444444:

Host command:	0xAA, 0x56, 0x03, 0x0B, 0x05, 0x04, 0xE9	(#11)
Module Answer:	0xAA, 0xD6, 0x01, 0x00, 0x7F	(#12)

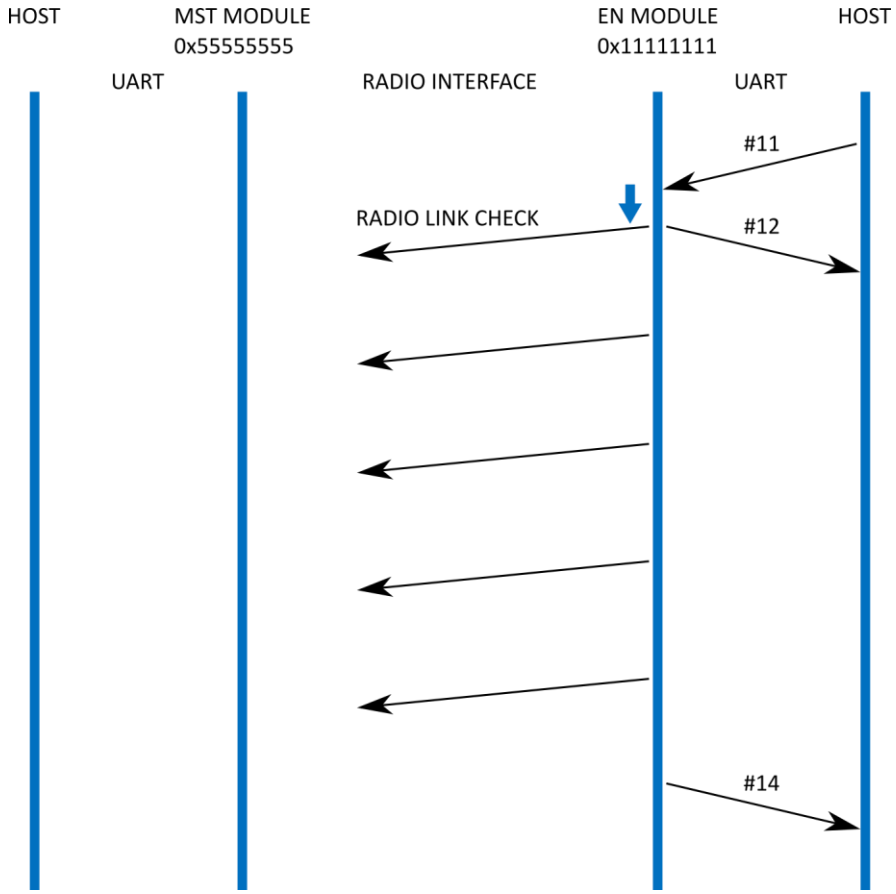
When the test is successful, Modules 0x11111111, 0x22222222, 0x33333333, 0x44444444 send an *indicate* to host:

Module Indicate:	0xAA, 0x57, 0x02, 0xFF, 0x05, 0xF9	(#13)



If the link check test does not succeed:

Module Indicate: 0xAA, 0x57, 0x02, 0x00, 0x00, 0xFD (#14)



6.4. MESSAGE UNCONFIRMED TRANSMISSION SESSION

This example shows how to send an unconfirmed message from EN (same procedure applies to MASTER node). UnconfirmedTxNumber has been set to 3 (Default value).

To send a PAYLOAD equal to {0x11, 0x22, 0x33, 0x44} from EN 0x11111111 to MST:

Host command: 0xAA, 0x50, 0x09, 0x00, 0xFF, 0xFF, 0xFF, 0xFF, **0x11, 0x22, 0x33, 0x44**, 0x57 (#15)

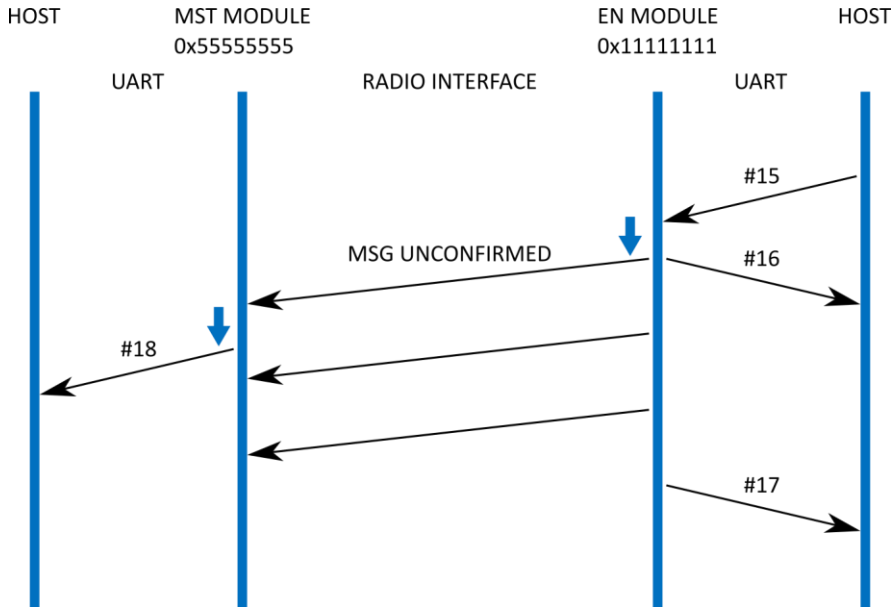
Module Answer: 0xAA, 0xD0, 0x01, 0x00, 0x85 (#16)

When the session ends, EN module sends back to the host an indication message containing the session time-on-air:

EN Module Indicate: 0xAA, 0x52, 0x05, 0x00, 0xC9, 0x00, 0x00, 0x00, 0x36 (#17)

When MST module receives a radio message, it indicates this to the Host with an indication message:

MST Module Indicate:0xAA, 0x53, 0x0C, 0x00, 0xC7, 0xFF, 0x06, 0x11, 0x11, 0x11, 0x11, **0x11, 0x22, 0x33, 0x44**, 0x3D (#18)



6.5. MESSAGE CONFIRMED TRANSMISSION SESSION

This example shows how to send a confirmed message from MST to EN. Here assume ConfirmedTxNumber equal to 3 (Default value).

To send a PAYLOAD equal to {0xAA, 0xBB, 0xCC, 0xDD, 0xEE, 0xFF} from MST to EN 0x11111111:

Host Command: 0xAA, 0x50, 0x0B, 0x01, 0x11, 0x11, 0x11, 0x11, **0xAA, 0xBB, 0xCC, 0xDD, 0xEE, 0xFF**, 0xBB (#19)

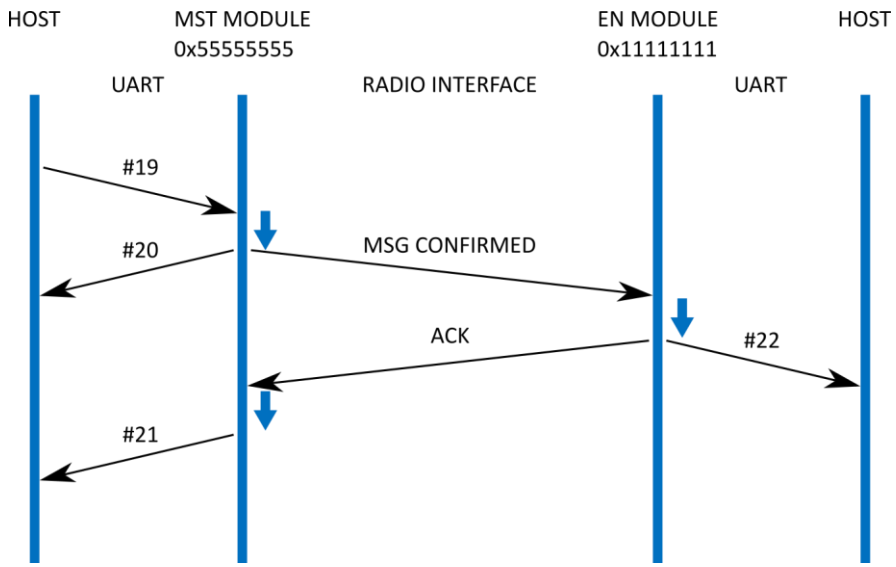
Module Answer: 0xAA, 0xD0, 0x01, 0x00, 0x85 (#20)

When the session ends, MST module sends back to the host an indication message containing the session time-on-air, a confirm of Ack reception, and the number of transmitted messages:

MST Module Indicate:0xAA, 0x51, 0x07, 0x00, 0x31, 0x04, 0x00, 0x00, 0x01, 0x01, 0xC7 (#21)

When a EN module receives a radio message, it indicates this to the Host with an indication message:

EN Module Indicate: 0xAA, 0x53, 0x0E, 0x00, 0xCB, 0xFF, 0x06, 0x55, 0x55, 0x55, 0x55, **0xAA, 0xBB, 0xCC, 0xDD, 0xEE, 0xFF,** 0xD6 (#22)



7. Glossary

SN = Serial Number

Fw = Firmware

LSB = Least significant byte

MSB = Most significant byte

Cks = Checksum

EN = END NODE

MST = MASTER

8. Revision History

Revision	Date	Description
0.1	08.09.2022	Preliminary
1.0	17.10.2022	-Add connection examples -Add timing diagrams