ST87MXX TCP/UDP-IP Application Note

Purpose and scope

This document provides details on the usage of AT commands for the embedded TCP/UDP-IP stack supported by the ST87MXX NB-IoT module.

Document status
Official

1 General information

1.1 Acronyms and terms

Table 1. Definitions of terms

Term	Definition
DNS	Domain Name System
IP	Internet Protocol
JTAG	Joint Test Action Group
MBR	Maximum Bit Rate
ME	Mobile Equipment
NCP	Network Control Protocol
PDP	Packet Data Protocol
PPP	Point-to-Point Protocol
QCI	QoS Class Identifier
QoS	Quality of Service
RAI	Release Assistance Indication
TA	Terminal Adaptor, e.g. a GSM data card (equal to DCE; Data Circuit terminating Equipment)
TCP	Transport Control Protocol
TE	Terminal Equipment, e.g. a computer (equal to DTE; Data Terminal Equipment)
TFT	Traffic Flow Template
UART	Universal Asynchronous Receiver Transmitter
UDP	User Datagram Protocol

1.2 Reference documents

The documents listed in Table 2 provide further information.

Table 2. Document references

Reference	Document
[1]	ST87MXX UM AT commands description

1.3 Revision history

Table 3. Document revision history

Date	Version	Changes
2023-12-05	V1.0	Official release
2024-03-18	V1.1	Add RAI option inside UDP use case. Add NBSENT use case
2024-06-17	V1.2	Update AT Commands. Typo corrections
2024-10-10	V1.3	Update AT Commands. Add DNS configuration and usage Update Error codes and URC lists
2025-01-08	V1.4	Update AT Commands. Update DNS service Update IPPARAMS service Remove Error and URC lists
2025-02-18	V1.5	Update AT DNS Command Add TCP/UDP binary transfer use cases
2025-04-04	V1.6	Update AT DNS Command
2025-07-01	V1.7	Remove AT command tables and add ST87MXX UM AT commands description reference
2025-08-25	V2.0	Update for release

1.4 Table of contents

1.5 List of tables

2 Introduction

This document lists several AT commands and parameters to explain TCP/UDP handling. **For the complete description of the AT commands and parameters, please refer to the AT command user manual [1]**.

3 IP configuration

3.1 Preamble

To establish an IP connection with TCP or UDP packet exchanges, the ME must first be attached (PDP context connection established) to an NB-IoT network. The platform should perform this automatically after boot.

This is notified to the Host by the message: +CGEV: ME PDN ACT 5

Note: the value **5** is the context ID used by default with the ST87MXX platform.

3.2 IP default parameters command

The **AT#IPPARAMS** command is used to read or set IP configuration default parameters.

Parameters	Description
<auto_ip>	Integer type 0: Automatic configuration of IP link not activated. 1: Automatic configuration of IP link activated. When activated, the stack will, by itself, configure the IPv4 and IPv6 addresses and parameters with the information provided by the network.
<preferred_ip_version>	Integer type 0: IPv4 is the preferred stack 1: IPv6 is the preferred stack. The preferred IP stack version will be used by default in case the parameter provided by the user cannot help to decide.
<max_ipstack_periodicity>	Integer type Max Timer in second for the IP stack periodicity. By default, it is set to 0xFFFF (no periodic wake-up is supported unless except it is managed in the Customer Application mode). Minimum value: 1, maximum value: 0xFFFF.
<auto_ip_timeout>	Integer type Timer in second between the module startup and attached PS attachment state for entering in sleep
<nbsent>	Integer type Indicates if the NB_SENT shall be sent in addition of OK / ERROR answer in case of UDP packet transmission. The NB_SENT response occurs when the packet has been received and acknowledged by the eNodeB. However, it does not guarantee that the packet has been received by the remote server. 0: No NB_SENT answer 1: NB_SENT answer
<DNSv4>	String type IPv4 address of the Domaine Name Server (DNS)
<DNSv6>	String type IPv6 address of the Domaine Name Server (DNS)

<max_ip_frame_size>	Integer type Maximum size of an IP frame (including IP header) in terms of bytes. Minimum value: 512, maximum value: 1600.
<dns_to_use>	Integer type Specify the DNS address to be used: DNS address provided by the network or DNS address saved in NVM. In case user wants to use the IP address provided by the network and no address is provided, the system will switch to the address saved in NVM. 0 : Use address stored in NVM 1 : Use address provided by the network

The ME can also use this **AT#IPPARAMS** command to retrieve the IP default configuration. When testing this command, the result is:

AT#IPPARAMS?

#IPPARAMS: 1,0,65535,60,0,8.8.8.8,0000:0000:0000:0000:0000:0000:0000:0000,1400,1
OK

The same AT command is also used to set preferences of IP connection, e.g. to configure the DNS server used to resolve URL names (see the AT#DNS command).

By default, the parameter settings are not retained after a reset. To make them permanent, the command **AT#RESET=1** must be sent so that the new setting will be saved in the NWM and kept after a reset.

3.2.1 DNS configuration

If <DNSv4>or <DNSv6> parameters are set to 0 in the AT#IPPARAMS command, the own specific DNS address of DNS is not used.

If the <dns_to_use> parameter is set to 1, the auto fill DNS server from NB-IOT network parameters is used.
After save, reset and re-attach of the modem, the DNS will be set with the DNS provided by the network itself.

The network DNS address can be checked with **AT+CGCONTRDP** command.
Here is an example of APN 5:

```
AT+CGCONTRDP=5

+CGCONTRDP:
5,5,"iotinternet.mnc010.mcc208.gprs","10.231.206.197.255.255.255.0","","172.20.2.3
9","172.20.2.10","","",,,,,,,

OK
```

The fields in bold blue letters are the primary and secondary DNS servers declared in the SFR French network.

3.2.2 IP maximum frame size

The maximum size of an IP frame can be set through the <max_ip_frame_size> parameter of the **AT#IPPARAMS** command. By default, this value is set to 1400: <max_ip_frame_size> = 1400

3.2.3 Example of IP parameters configuration

The following lines show an example for automatic configuration, IPv4, no NB_SENT URC, no user DNS set, 1400 bytes maximum frame, Auto DNS.

```
AT#IPPARAMS=1,0,65535,60,0,0.0.0.0,0000:0000:0000:0000:0000:0000:0000:0000,1400,1
OK

AT#RESET=1

OK
```

Note: After boot and by default, the IP configuration is set to automatic IP. After platform's boot, the following appears on the UART trace:

`#SIMST: 1` <----- SIM detected

`+CEREG: 2` <----- Registration in progress

`+CEREG: 0` <----- Registration stopped

`+CEREG: 2` <----- Registration in progress

`+CGEV: ME PDN ACT 5` <----- Attached to the network with ID n°5

`+CEREG: 1` <----- Registered URC

`#IPCFG: 5,0,1` <----- IP configuration set automatically by the platform

3.3 Set IPv4 or IPv6 configuration:

To start a TCP or UDP communication, the host shall first perform the IP configuration.
Then, if the automatic configuration of IP link is not activated (see chapter “3.2 IP default parameters command”), the ME shall retrieve its IP address by using:

AT+CGPADDR=5

The answer from network can be:

```
+CGPADDR: 5,"100.115.11.113","2A02:8440:5202:65D1::0029:28FD:B001"

OK
```

- first parameter (“5”) is the default context ID.
- second parameter ("100.115.11.113") is the IPv4 address.
- third parameter ("2A02:8440:5202:65D1::0029:28FD:B001") is the IPv6 address.

Depending on the network, it is possible to retrieve only one of the two addresses.

Then, the ME shall activate the configuration with command **AT#IPCFG**.

3.3.1 IPv4 configuration

The IPv4 configuration is done through the following command:

AT#IPCFG=5,0,<UE IPv4 address>

For example, with **AT#IPCFG=5,0,100.115.11.113**, the ME should receive this answer:

IP Network is Up for context 5

Once done, to test the IP connection, the ME can ping Google with following command:

AT#IPPING=5,8.8.8.8

1,0,10,1518

#IPPING: 1,1,0,1518,1518,1518

OK

3.3.2 IPv6 configuration

The IPv6 configuration is done through the following command:

AT#IPCFG=5,0,<UE IPv6 address>[,<prefix length>]

For example:

AT#IPCFG=5,0,2A02:8440:5202:65D1::0029:28FD:B001

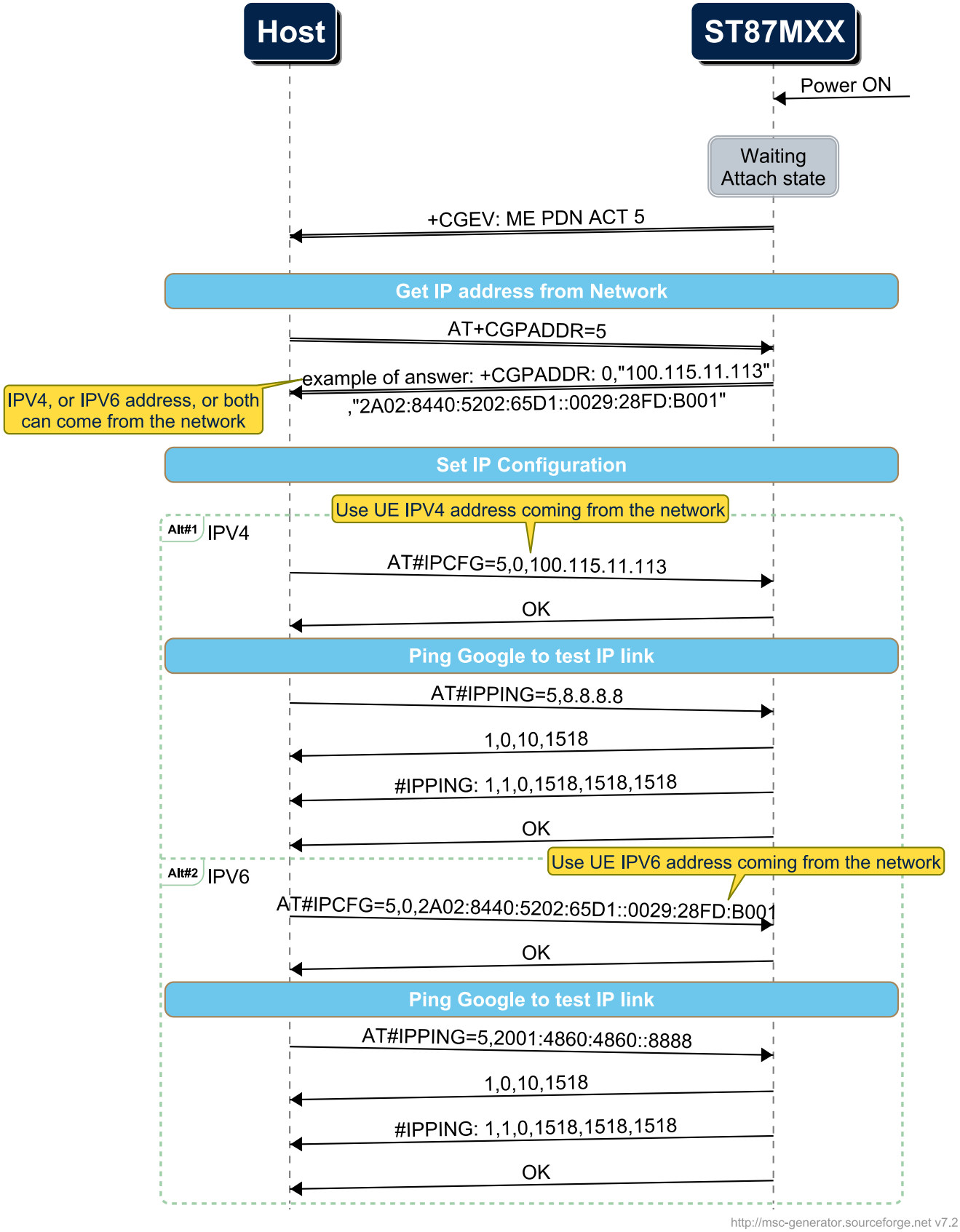
Then the ME should receive this answer:

IP Network is Up for context 5

Once done, Google can be pinged in IPv6 with following command:

AT#IPPING=5,2001:4860:4860::8888

3.4 MSC for IP configuration



4 TCP use cases

4.1 Preamble

For a TCP connection and for TCP packet exchange, the ME must configure the IP layer if the AutoIP configuration is not set by default (see chapter “3.2 IP default parameters command”). Afterward, the ME can initiate a TCP connection with a TCP server.

To run a test with TCP functionality, a TCP server must be available inside the Internet network.

4.2 Create TCP socket

To exchange TCP packets, first a socket must be created. It is then used to exchange data.

For example, the ME can create a socket with the following AT command:

AT#SOCKETCREATE=5,0,TCP,1000,1,1,1

5: is the context ID used for IP connection. It has to be used as IP configuration.
0: for IPv4, **1**: for IPv6 it should be the same chosen during IP configuration
TCP is the type of socket used to create it.
1000: Example of Local Port customer choice
1 is the timeout set in seconds to send to the server the request.
1 is the timeout set in seconds to send and receive the response from the server.
1 is enabling the URC reception when TCP data are available and can be read with **AT#IPREAD**

The answer is:

#SOCKETCREATE: 0 ------> socket is created with ID 0

OK

4.3 Connect TCP socket

After the creation of the socket, the ME can connect the TCP layer to the remote server using the following command:

AT#TCPCONNECT=5,0,<IP Remote Server address> ,<TCP Remote Server port>

5: is the context ID used for IP connection: 5 it is the one used by default by the ST87MXX platform
0: is the Socket ID received during socket creation.

The IP Remote Server address should be a IPv4 address if a IPv4 socket has been created, and it should be a IPv6 address is a IPv6 socket has been created.

4.4 Send data / Receive Data

The socket is now connected from ME to server; the data socket can be used.

To send data from UE to the TCP server, the following command can be used:

AT#IPSENDTCP=5,0,0,ST87M NBIOT TCP test

5: is the context ID used for IP connection: 5 it is the one used by default by the ST87MXX platform.
0: Is the Socket ID received during socket creation
0: Data type is ASCII

If the “URC generation bit” param was activated during the creation of the socket, the following notification URC is received when data is available:

#IPRECV: 5,0

To read the data coming from server inside the received data buffer, the following command can be used:

AT#IPREAD=5,0

5: is the context ID used for IP connection: 5 it is the one used by default by the ST87MXX platform.
0: Is the Socket ID received during socket creation

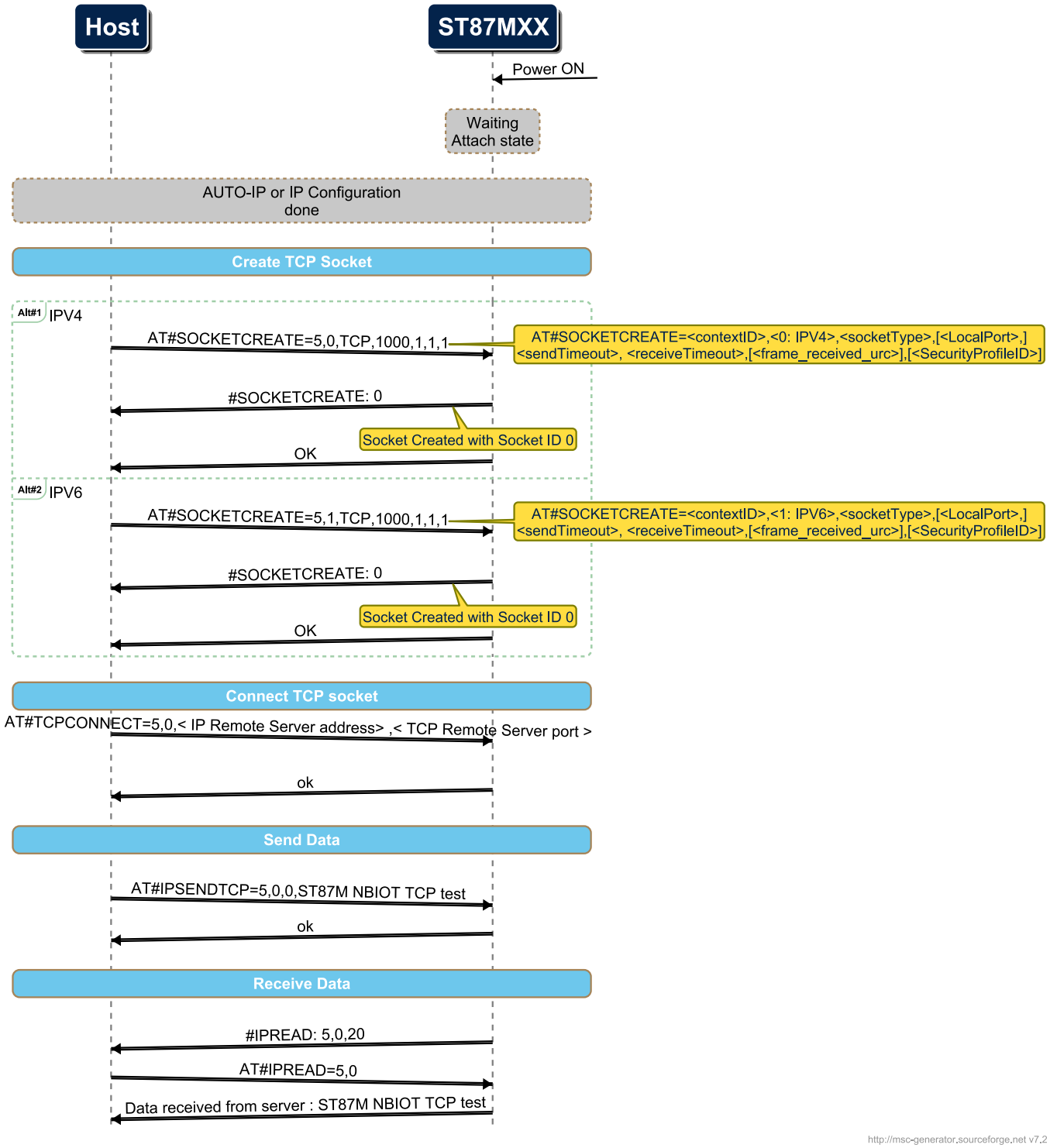
The answer is:

#IPREAD: 5,0,20 /* 20 is the size of packet received */
ST87M NBIOT TCP test

OK

Note: For this example, the answer is the same as the TCP sent packet, since a TCP test server with loopback feature is used (the received packet is sent back to the sender).

4.1 MSC for TCP use cases



5 UDP connection (send and receive one packet)

5.1 Preamble

For a UDP connection and for UDP packet exchange, the ME must configure the IP layer if the AutoIP configuration is not set by default (see chapter “3.2 IP default parameters command”). Afterward, the ME can initiate a UDP connection with a UDP server.

To run a test with UDP functionality, a UDP server must be available inside the Internet network.

5.2 Create UDP socket

To exchange UDP packets, first a socket must be created. It is then used to exchange data.

For example, the ME creates a socket with the following AT command:

AT#SOCKETCREATE=5,0,UDP,5000,1,1,1

5: is the context ID used for IP connection. It has to be used as IP configuration.
0: for IPv4, **1**: for IPv6 (it should be the same chosen during IP configuration)
UDP is the type of socket used to create it.
5000: is the local port number used by UDP server to send data
1: is the timeout set in seconds to send to the server the request.
1: is the timeout set in seconds to send and receive the response from the server.
1: is activating if customer needs an URC (#RECV) when TCP or UDP data is received from the network and can be read with **AT#IPREAD** command.

The answer is:

#SOCKETCREATE: 0 ------> socket is created with ID 0

OK

5.3 Connect UDP socket

The UDP transmission is a connectionless protocol. Establishing a connection for a UDP socket is not required.

5.4 Send data / Receive Data

5.4.1 Send data

To send data from UE to UDP server, the following AT command can be used:

AT#IPSENDUDP=5,0,<xxx.xxx.xxx.xxx>,6000,0,0,ST87M NBIOT UDP test

5: is the context ID used for IP connection
0: is the Socket number used during socket creation
<xxx.xxx.xxx.xxx>: IP Address of UDP server. It should be a IPv4 address if a IPv4 socket has been created, and it should be a IPv6 address is a IPv6 socket has been created.
6000: Port number of UDP server to receive data
0: RAI not requested
0: Data type set to ASCII
ST87M NBIOT UDP test: String to send to UDP server

If the data is coming from the server and if during the socket creation, the customer has enabled the URC reception, the host will receive this URC when incoming data is available:

The answer is:

#IPRECV: 5,0

5: is the context ID used for IP connection
0: is the socket ID used for socket creation

5.4.2 Read data

To read inside received data buffer to get data from the server, the following AT command is used:

AT#IPREAD=5,0
5: is the context ID used for IP connection
0: is the socket ID used for socket creation

The answer is:

#IPREAD: 5,0,20 ------> URC received to notify incoming read data

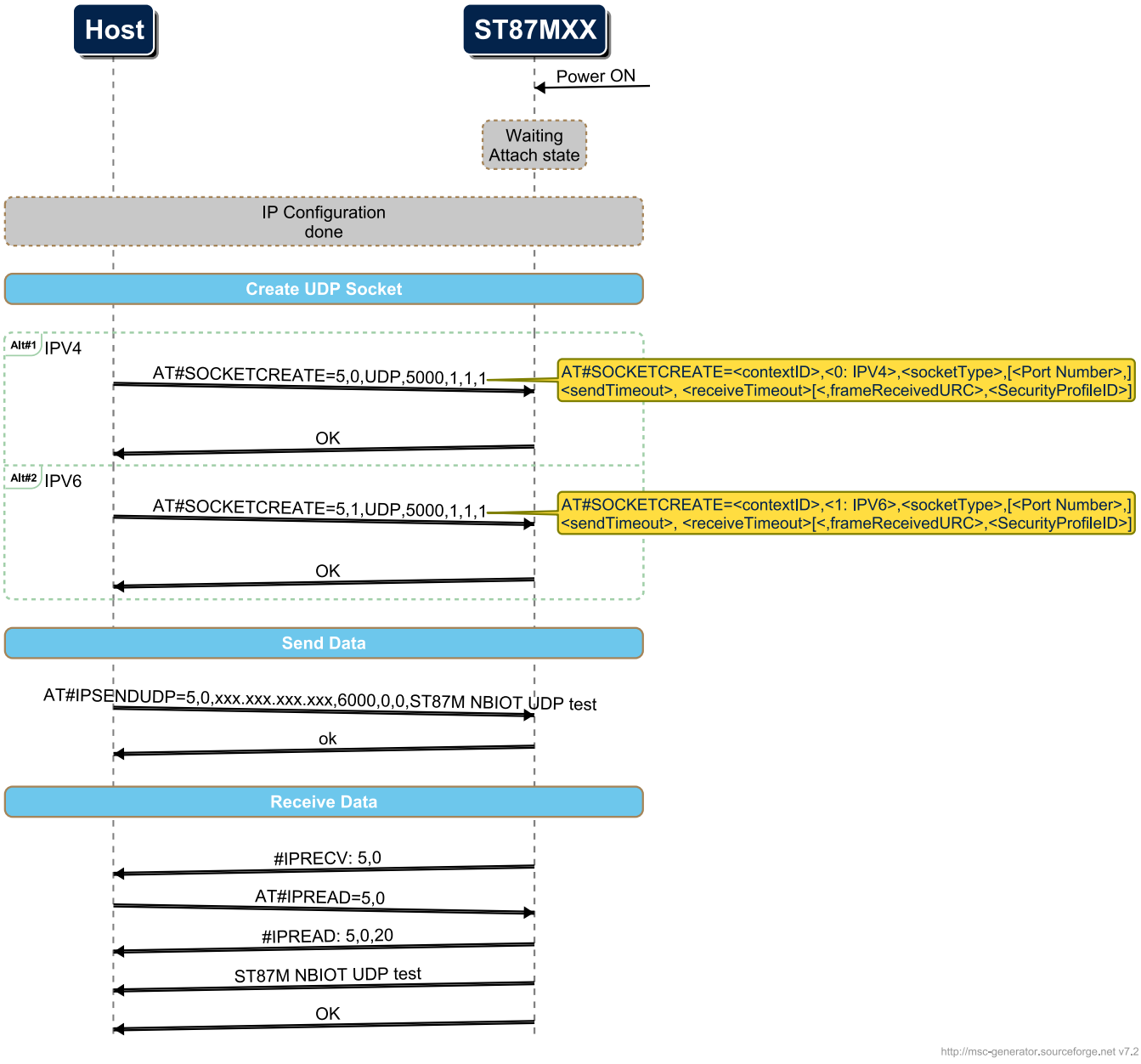
5: is the context ID used for IP connection
0: is the socket ID used for socket creation
20: the size of data received

ST87M NBIOT UDP test ------> ASCII data

OK

Note: For this example, the answer is the same as the UDP sent packet since a UDP test server with loopback feature is used (the received packet is sent back to the sender).

5.5 MSC for UDP use case (send/receive)



6 UDP connection (multiple packets with RAI)

6.1 Preamble

For UDP connection and for UDP packets exchange, the ME must configure the IP layer if the AutoIP configuration is not set by default (see chapter “3.2 IP default parameters command”). Afterward, the ME can initiate a UDP connection with a UDP server.

To run a test with UDP functionality, a UDP server must be available inside the Internet network.

6.2 Create UDP socket to send data only (URC data coming from network disabled)

To exchange UDP packets, first a socket must be created. It is then used to exchange data.

For example, the ME creates a socket with the following AT command:

AT#SOCKETCREATE=5,0,UDP,5000,1,1,0

5: is the context ID used for IP connection. It has to be used as IP configuration.
0: for IPv4, **1**: for IPv6 (it should be the same chosen during IP configuration)
UDP is the type of socket used to create it.
5000: is the local port number used by UDP server to send data
1: is the timeout set in seconds to send to the server the request.
1: is the timeout set in seconds to send and receive the response from the server.
0: is deactivating the URC (#RECV) when TCP or UDP data is received from the network

The answer is:

#SOCKETCREATE: 0 ------> socket is created with ID 0

OK

6.3 Connect UDP socket

The UDP transmission is a connectionless protocol. Establishing a connection for a UDP socket is not required.

6.4 Send multiple packets with RAI activated

The goal of this use case is to address the scenario where the customer wants to send multiple packets without receiving a response from the network and activate the RAI feature to release the connection as soon as possible to save power. The RAI feature should be enabled only on the last packet sent by application.

The following lines show a scenario where the user sends 5 packets to the UDP server with RAI enabled on the last packet.

AT#IPSENDUDP=5,0,<xxx.xxx.xxx.xxx>,6000,0,0,ST87 Packet 1
OK

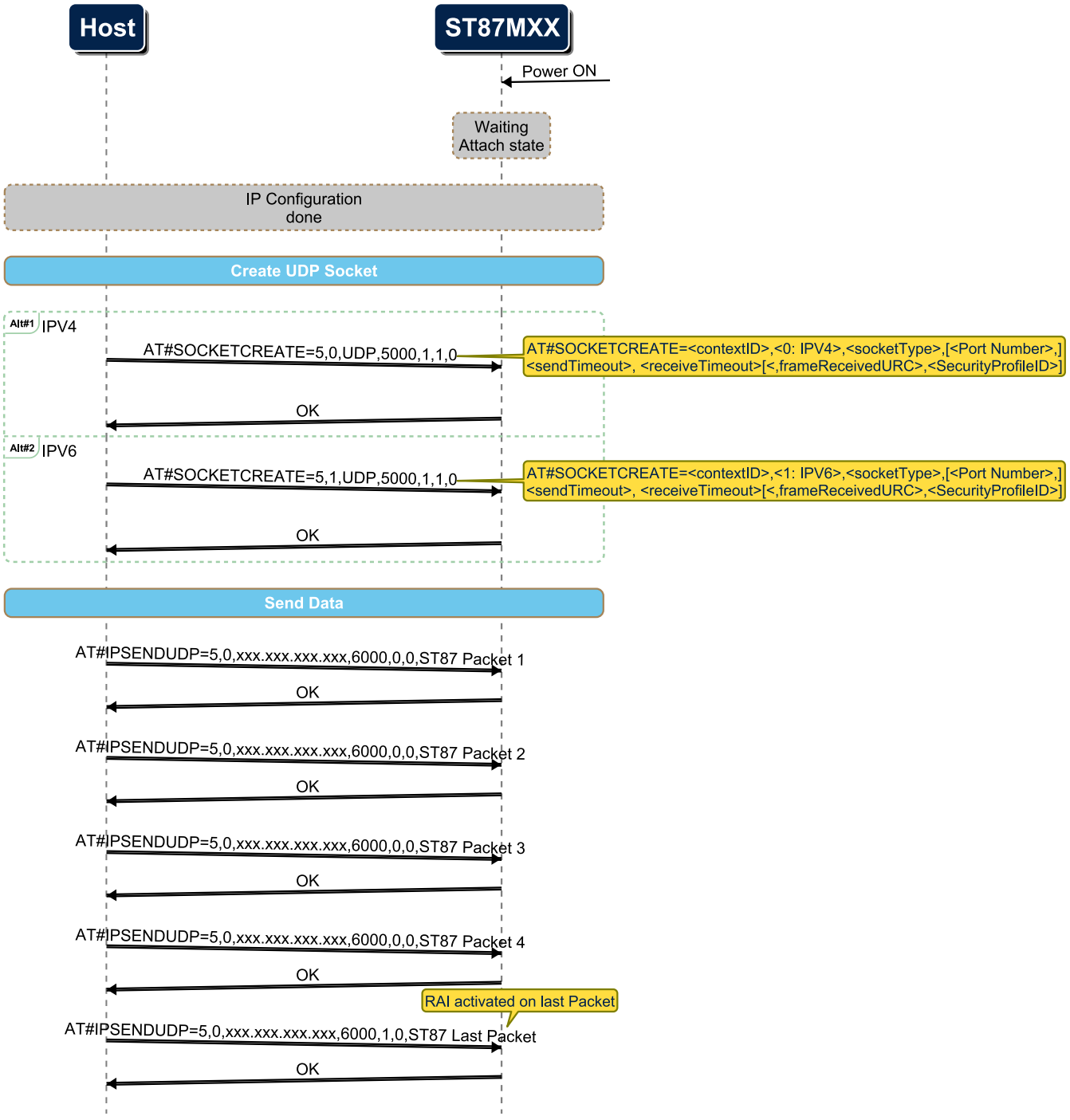
AT#IPSENDUDP=5,0,<xxx.xxx.xxx.xxx>,6000,0,0,ST87 Packet 2
OK

AT#IPSENDUDP=5,0,<xxx.xxx.xxx.xxx>,6000,0,0,ST87 Packet 3
OK

AT#IPSENDUDP=5,0,<xxx.xxx.xxx.xxx>,6000,0,0,ST87 Packet 4
OK

AT#IPSENDUDP=5,0,<xxx.xxx.xxx.xxx>,6000,1,0,ST87 Last Packet /* RAI requested on the last packet */
OK

6.5 MSC for UDP multiple packets with RAI



6.6 Send multiple packets with NB_SENT URC activated

The goal of this use case is to address the scenario where a customer wants to send multiple packets and determine if the packets are successfully sent over the radio interface.
The URC message “NB_SENT” is sent in addition to the OK / ERROR answer in case of UDP packet transmission.
The NB_SENT response occurs when the packet is successfully received and acknowledged by the eNodeB. However, it does not guarantee that the packet has been received by the remote server.
To activate this NB_SENT URC feature, the IP stack configuration must be changed as the following:

AT#IPPARAMS=<auto_ip>,<preferred_ip_version>,<max_ipstack_periodicity>,<auto_ip_timeout>,<nbsent>

For example: if the customer wants to have:

- AutoIp
- IPv4
- NB_SENT URC activated.

The command is as follows:

AT#IPPARAMS=1,0,65535,60,1

Afterward, a save and reset of the new IP configuration is required:

AT#RESET=1

The following lines show a scenario where the user sends 5 packets to the UDP server with the NB_SENT URC enabled.

AT#IPSENDUDP=5,0,<xxx.xxx.xxx.xxx>,6000,0,0,ST87 Packet 1
OK

AT#IPSENDUDP=5,0,<xxx.xxx.xxx.xxx>,6000,0,0,ST87 Packet 2
OK

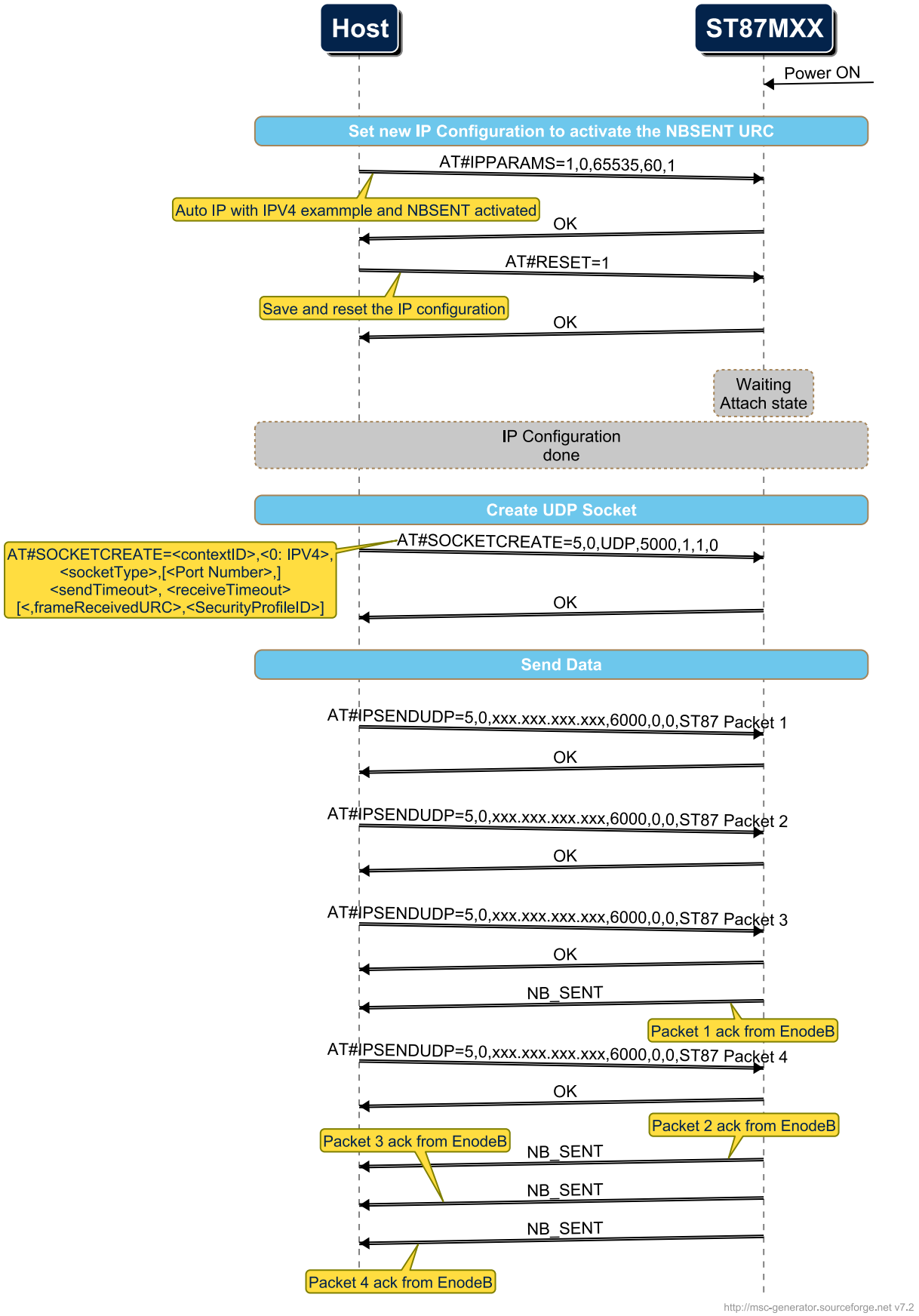
AT#IPSENDUDP=5,0,<xxx.xxx.xxx.xxx>,6000,0,0,ST87 Packet 3
OK

NB_SENT /* Packet 1 ack from EnodeB */

AT#IPSENDUDP=5,0,<xxx.xxx.xxx.xxx>,6000,0,0,ST87 Packet 4
OK

NB_SENT /* Packet 2 ack from EnodeB */
NB_SENT /* Packet 3 ack from EnodeB */
NB_SENT /* Packet 4 ack from EnodeB */

6.7 MSC for UDP multiple packets with NBSENT URC activated



7 UDP/TCP connection (binary mode)

7.1 Preamble

For UDP/TCP connections and UDP/TCP packet exchanges, the ME must configure the IP layer if the AutoIP configuration is not set by default (see chapter “3.2 IP default parameters command”). Afterward, the ME can initiate a UDP or TCP connection with a UDP or TCP server.

To run a test with UDP/TCP functionality, a UDP/TCP server must be available inside the Internet network.

7.2 Create UDP or TCP socket to send data in binary mode

To exchange UDP or TCP packets, first a socket must be created. It is then used to exchange data.

The socket to transfer ASCII, binary, or hexadecimal data is created with a unique syntax. The command is the same as for ASCII, like described before.

For example, the ME creates a socket with the following AT command:

/* UDP socket Creation */
AT#SOCKETCREATE=5,0,UDP,5000,100,100,0

/* TCP socket in creation */
AT#SOCKETCREATE=5,0,TCP,5000,100,100,0

7.3 Connect TCP socket

After creating the socket, the ME can connect the TCP layer to the remote server. For binary transfer, the socket connection uses the same command.
Note: A UDP socket does not require any connection request.

AT#TCPCONNECT=5,0,<IP Remote Server address>,<TCP Remote Server port>

5: is the context ID used for IP connection: 5 it is the one used by default by the ST87MXX platform
0: is the Socket ID received during socket creation.
XXX.XXX.XXX.XXX: IP address
XXXX: TCP port

7.4 Send data

The socket is now connected from ME to the server; the data socket can be used.

7.4.1 Send binary data from UE to TCP server

The command to send data in binary mode during a TCP connection is the following:

AT#IPSENDTCP=5,0,1,<size of binary data to transfer in bytes> <CR>

Notes:
- A <CR> at the end of the command is required to initiate the binary transfer process.
- The data must be sent after receiving the OK.

The following lines show a scenario where the user has a TCP IPv4 connection and sends data transfer in binary mode with the following parameters for the AT#IPSENDTCP command:

- **5**: is the context ID used for IP connection: 5 it is the one used by default by the ST87MXX platform.
- **0**: Is the Socket ID received during socket creation
- **1**: Data type is BINARY
- **10**: Example of 10 bytes

AT#SOCKETCREATE=5,0,TCP,5000,100,100,0<CR>

#SOCKETCREATE: 0

OK

AT#TCPCONNECT=5,0,9x.1xx.1xx.1xx,1883<CR>

OK

AT#IPSENDTCP=5,0,1,10<CR>

OK /* Wait OK before sending the data in binary mode */

1234567890<CR> /* 10 bytes sent in binary mode. The <CR> is needed only when using the ST GUI. */

OK /* OK received after sending process */

7.4.2 Send binary data from UE to UDP server

The command to send data in binary mode during a UDP connection is the following:

AT#IPSENDUDP=5,0,<IP address>,<port>,<RAI mode>,1,<size of binary data to transfer in bytes> <CR>

Notes:

- A <CR> at the end of the command is required to initiate the binary transfer process.
- The data must be sent after receiving the OK.

The following lines show a scenario where the user has a UDP IPv4 connection and sends data transfer in binary mode with the following parameters for the AT#IPSENDUDP command:

- **5**: is the context ID used for IP connection: 5 it is the one used by default by the ST87MXX platform.
- **0**: Is the Socket ID received during socket creation
- **9x.1xx.1xx.1xx: Address of UDP server**
- **1883: UDP server port**
- **RAI mode:** nothing, 0, 1, or 2 can be used depending on whether the user sends more packets or is waiting for packets from the network. In the example below, 1 is used.
- **1**: Data type is BINARY
- **10**: Size of binary data

AT#SOCKETCREATE=5,0,UDP,5000,100,100,0 <CR>

#SOCKETCREATE: 0

OK

AT#IPSENDUDP=5,0,9x.1xx.1xx.1xx,1883,1,1,10 <CR>

OK /* Wait OK before sending the data in binary mode */

1234567890 <CR> /* 10 bytes sent in binary mode. The <CR> is needed only when using the ST GUI.*/

OK /* OK received after sending process */

IMPORTANT NOTICE – PLEASE READ CAREFULLY

STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST’s terms and conditions of sale in place at the time of order acknowledgement.
Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of
Purchasers’ products.
No license, express or implied, to any intellectual property right is granted by ST herein.
Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.
ST and the ST logo are trademarks of ST. For additional information about ST trademarks, please refer to www.st.com/trademarks. All other product or service names are the property of their respective owners.
Information in this document supersedes and replaces information previously supplied in any prior versions of this document.