neo.cortec

Communication Specification System UART

Doc Status:	**Released**
Doc version:	**2.0**
Date:	**November 2018**

Table of Contents

1 Introduction

This document is in addition to the Integration Manual for NCxxxx Series Modules document. This documents provides more detail on how to configure settings in the NeoCortec NeoMesh modules through the System API port. In addition it provides instructions for how to upload new firmware in the modules from an embedded controller.
Please refer to the Integration Manual for NCxxxx Series Modules document for details on how to connect to the System UART.

2 Communication procedure

During normal operation of the NeoMesh module the protocol stack is running and the System UART is used solely for outputting trace messages. If one wishes to modify settings in the module, the communication protocol stack first need to be stopped, and the module need to be put into bootloader mode.
Once in bootloader mode, the module will accept commands on the System UART.

To start modifying settings, login with a password is required. Only settings which are permitted by a certain password level can be read or written.

When the Bootloader is enabled, a copy of the settings is stored in the RAM of the module. In the RAM block it is possible to modify settings. Once the settings are modified, they can be written back to flash.

It is possible to read a setting either from the RAM copy or from the flash memory with the Get Setting RAM or the Get Setting Flash commands.

3 Communication Specification – System UART

3.1 Logical data exchange

Data is exchanged over the interface in Big Endian byte order.

The general format of a data frame is:

Section	Header	Length	Cmd1	Cmd2	Data	Trail
Length	1 byte	1 byte	1byte	1byte	n	1byte

Header: Shall always be 0x3E when sending data to the module. When the module sends data to the host controller it will mimic the Header from the send frame.

Length: The length of the frame after the Length field.

Cmd1: Identifies the destination for the command – currently only bootloader will accept messages, and the Cmd value for bootloader is 0x01.

Cmd2: The actual System command or System Response.

Data: Optional field for arguments.

Trail: Shall always be 0x21.

3.2 Commands and System responses

There are two types of frames exchanged, Commands and System Responses:

Commands are frames accepted by the NeoMesh module.

System responses is messages sent from the NeoMesh module in response to a previously issued Command.

3.2.1 List of system commands:

Command name	Cmd1	Cmd2	Data	Description
Login	0x01	0x03	5 Byte password	Login shall be performed prior to reading or writing settings to the module.
Reset List Iterator	0x01	0x04	non	Resets the settings list iteration pointer to the beginning of the settings list. Should always be called first prior to reading the Settings List.
Get List	0x01	0x05	non	Outputs up to 10 elements of the settings list. Can be called multiple times to iterate through the settings list.
Get Setting Flash	0x01	0x06	1 byte id	Reads a particular setting from the Flash memory
Get Setting RAM	0x01	0x07	1 byte id	Reads a particular setting from the RAM memory
Commit Settings	0x01	0x08	non	Writes the settings currently stored in RAM to Flash
Discard Settings	0x01	0x09	non	Reloads settings from Flash into RAM
Set Setting	0x01	0x0A	x bytes setting value	Modifies a particular setting value in RAM. See format below.
Start Protocol Stack	0x01	0x12	non	Exit Bootloader and start the protocol stack.
Start Bootloader	0x01	0x13	non	Stop protocol stack and start Bootloader.

Note: The only System Command which can be executed when the bootloader is not started, is the Start Bootloader (Cmd2 = 0x13). All other commands will be silently discarded.

3.2.1.1 Setting Value format

When modifying a setting (using Set Setting command), the setting is given by this format:

ID	Data
The id of the setting which is being modified	The new setting value given in x bytes.

3.2.2 List of system responses:

Response name	Cmd1	Cmd2	Data	Description
Login_Ok	0x03	0x80	non	Indicated that the previously send login command was successful
Login_Error	0x03	0x81	non	Indicates that the previously send login command was not successful – bad password. Will also be send if a setting is being read or written to which the password level currently entered does not give access to.
Boot_Loader_Started	0x03	0x82	non	Response to a successful Start Bootloader command
Protocol_Stack_Started	0x03	0x83	non	Response to a successful Start Protocol Stack command
Protocol_Stack_Error	0x03	0x84	non	Response to a Start Protocol Stack command when the FW in module is faulty.
Settings_List_Output	0x01	0x85	Settings list	Response to a Get List command. See detailed format of data below.
Setting_Value_Flash	0x01	0x86	Setting Value	Response to a Get Setting Flash command. The Setting Value will be the raw setting bytes.
Setting_Value_RAM	0x01	0x86	Setting Value	Response to a Get Setting RAM command. The Setting Value will be the raw setting bytes.

3.2.2.1 Settings list format

The Settings_List_Output sends the Settings List in the following format.

Each element is 3 bytes long. Each element contains the following information:

Byte#	0	1	2
Value	Setting Id	Setting Value Length	Access right: 3 Read/Write 2 Read

To read the complete list of settings, repeat the get list command until either a non-full list (less than 10 elements) or an empty list is returned.

3.2.2.2 Setting Value format

The Setting_Value_Flash or Setting_Value_RAM sends the setting value in the following as the raw settings bytes. No header is included.

4 Examples

4.1 Start Bootloader

Command	3E 03 01 13 21
Response	3E 03 03 82 21

4.2 Start Protocol Stack

Command	3E 03 01 12 21
Response	Ok: 3E 03 03 83 21 Error: 3E 03 03 80 21

4.3 Login

Command	3E 08 01 03 4C 76 6C 31 30 21
Response	Ok: 3E 03 03 80 21 Error: 3E 03 03 81 21

Bytes marked in Green text above, is the password. In this case the bytes corresponding to password Lvl10.

4.4 Reset List Iterator

Command	3E 03 01 04 21
Response	3E 03 03 80 21

4.5 Get list

Command	3E 03 01 05 21
Response	3E 21 01 85 00 05 03 04 05 03 05 05 03 06 05 03 07 05 03 0A 02 03 0F 01 03 10 06 03 11 03 03 12 03 03 21

Bytes marked in Green text above, is the settings list.
In the example above, there are 10 settings identified. Next time Get List is called, then next 10 settings will be transmitted.

4.6 Get Setting flash:

Command	3E 04 01 06 0A 21
Response	3E 05 01 86 00 03 21

In this example, settings Id 0x10 is being read which is the Node Id. The response is the actual Node Id of the module (0x00 0x03).

4.7 Get Setting RAM:

Command	3E 04 01 07 0A 21
Response	3E 05 01 87 00 03 21

This is the same as for Flash, but read form the RAM copy. So when settings are modified it is possible to read back the new RAM value or the old Flash value, before committing the new settings.

4.8 Commit settings:

Command	3E 03 01 08 21
Response	3E 03 03 80 21

4.9 Discard settings:

Command	3E 03 01 09 21
Response	3E 03 03 80 21

It will reload the ram copy of settings from flash, there by efficiently discards any changes.

4.10 Set Setting:

Command	3E 06 01 0A 0A 00 04 21
Response	3E 03 03 80 21

The example above sets the Node Id to 0x00 0x04.

5 Flash module firmware

5.1 Introduction

This sections describes the required steps to upload new firmware to the module from an external controller. It is considered an advanced topic, and is not the recommended module firmware flash procedure.

CAUTION: When updating the firmware, there is a risk that the module can be permanently damaged. It is recommended to use the tools provided by NeoCortec when updating the firmware, and only in special situations should it be necessary to update the firmware from an embedded controller.

5.2 "neo" file format

The firmware for NeoMesh modules is distributed in a binary file which contains the actual firmware as well as relevant meta data for the particular firmware. The firmware is divided into two parts: A bootloader and the protocol stack. Each part is encrypted, and will be decrypted by the NCxxxx module once it has been uploaded.

For firmware updates, NeoCortec use two different file formats:

1. xxxxx.neo
2. xxxxx.combined.neo

As an example, the file could be named like this:

1159_NC2400 .combined.neo

Where:

- 1159 is the build number of the image(s)
- NC2400 indicates which module the FW is intended for

The first file contains a single part – usually only the protocol stack. The second file contains multiple parts – usually the bootloader and the protocol stack. Normally the second file type shall be used, and both the bootloader and the protocol stack shall be updated at the same time.

The two file types use the same format:

Byte	Example (Hex)	Description
0	10	Layout format (this)
1-2	0037	Offset to image start (from layout format) Big-Endian
3-4	0800	Start address of the image (when writing to the NCxxxx module)
5-20	78 6A 10 B6 2F 97 4D B4 8D 2F 3F EE 38 54 3C 56	AES128 – Init Vector.
21-22	1010	Length of image
23-24	0483	Build number
25-44	70 69 63 c0 3d 37 37 63 d2 f8 9d af 65 51 63 49 8c 4c d2 4f	Internal reference – ignore.
45	08	Length of Image name
46-54	6E 65 6F 2E 63 6F 72 65 00	Null terminated Image name
55-4167	Image Data	The actual FW (encrypted)
4168	10	Start of next image (layout format)

Please note that the Byte column in the table above is only true for this example. If the length of the image or the length of the image name is different, then the indexes will also be different.

5.3 Uploading new firmware to the NCxxxx module

To upload the image the following steps are needed:

1. Start Bootloader
2. Send AES128 Init Vector
3. Erase Flash
4. Upload image
5. Check image
6. Restart image
7. Jump to step 1 if more than one part is being updated

Please note that when the file “combined.neo” is being used, it is important to adhere to the order in which the parts are arranged in the file. It is further necessary to perform step 6 even if more parts are being uploaded.

The following sections of the document lists all the commands necessary to complete the steps above.

5.3.1 List of system commands for uploading firmware

Command name	Cmd1	Cmd2	Data	Description
Start Bootloader	0x01	0x13	non	Stop protocol stack and start Bootloader. Module will respond "Bootloader Started" if successful.
Set Init_Vector	0x01	0x01	16 bytes	Sets the AES128 Init Vector which is needed by the module to decrypt the firmware. Module will respond "OK" if successful.
Erase flash	0x01	0x00	non	Erases the entire flash memory. Module will respond "OK" if successful.
Upload Image	0x01	0x02	Address/2 (2 byte) "N" Number of blocks (1 byte) Image data (N * 16 bytes)	The firmware shall be written in blocks of 16 bytes. The address shall be the start address of the first block divided by 2. Each upload can be maximum 11 blocks of 16 bytes. The firmware is always larger than 11 blocks, and command must be used multiple times. Remember to move the address reference at each call. Module will respond "OK" if successful.
Check Image	0x01	0x19	Non	Perform CRC check on protocol stack image. Module will respond "OK" if successful.
Start Protocol Stack	0x01	0x12	Non	Exit bootloader and start the protocol. Module will respond: "Pre Image Start Delay" <1 sec delay> "Protocol Stack Started" If successful.

5.3.2 List of system responses

Response name	Cmd1	Cmd2	Data	Description
OK	0x03	0x80	Non	OK response from the last issued command if successful.
Error	0x03	0x81	Non	Indicates the last issued command completed with an error.
Bootloader Started	0x03	0x82	Non	Indicates that the command "Start Bootloader" was successful.
Protocol Stack Started	0x03	0x83	Non	Indicates that the command "Start Protocol Stack" was successful.
Pre Image Start Delay	0x03	0x85	Non	1 second delay before starting Protocol Stack to allow cancelling Protocol Stack Start.