GLEN DIMPLEX
NORDIC

Nobo Hub – API version 1.1
for communication between a client and a Nobo Hub

Version 4, 26.3.2018.

CONTENTS

1 DISCLAIMER

This document is published to be intended as a courtesy to our users that would like themselves to integrate Nobo Hub with other smart home solutions.

By making use of the information in this document, you agree to the following:

NO WARRANTIES: All of the information provided in this document is provided "AS-IS" and with NO WARRANTIES. No express or implied warranties of any type, including for example implied warranties of merchantability or fitness for a particular purpose, are made with respect to the information, or any use of the information, in this document. Glen Dimplex Nordic AS makes no representations and extends no warranties of any type as to the accuracy or completeness of any information or content in this document.

DISCLAIMER OF LIABILITY: Glen Dimplex Nordic AS specifically DISCLAIMS LIABILITY FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES and assumes no responsibility or liability for any loss or damage suffered by any person as a result of the use or misuse of any of the information or content in this document. Glen Dimplex Nordic AS assumes or undertakes NO LIABILITY for any loss or damage suffered as a result of the use, misuse or reliance on the information and content in this document.

If you publish your integration with Nobo Hub / Nobo Energy Control, you must make explicitly clear that the system/service/software is not officially supported or endorsed by Glen Dimplex Nordic AS, and that you are not an official partner of Glen Dimplex Nordic as, unless you have a signed formal agreement with Glen Dimplex Nordic AS.

2 INTRODUCTION

This document describes version 1.1 of the API for communication between Nobo Hub (also known as Nobo ecoHUB) and clients.

This document describes how a client can communicate with a Nobo Hub. This API is used today for all communication between Nobo Hub and the Nobo Energy Control app for Android and iOS.

Nobo Hub can also be controlled via the Internet, by wrapping the API described in this document in encryption and extra headers, and sending traffic via a cloud service. Communication via the Internet is not described further in this document.

Glen Dimplex Nordic reserves the right to modify the API in non-backwards compatible ways, but we will avoid this if we can, and inform our official partners as soon as possible if we do.

3 AUTOMATIC UPDATE AND SYNCRONIZATION

- The Nobo Hub has an automatic firmware update function. It checks approximately every hour with Glen Dimplex Nordic's update server.
- If several clients are connected simultaneously, the Hub always pushes any changes to all connected clients, to keep them synchronized.
- The Hub always pushes any changes to all connected clients, also if the change is not initiated by a connected client. Examples include an Override that has expired, or an updated temperature value from a Nobo Switch. There is no need to do continuous polling as long as the client has an open network connection to the Hub that is being kept alive.

For more details of the Hub and Nobo Energy Control system functions, please see our online user guide at http://help.nobo.no.

4 GENERAL NETWORK REQUIREMENTS

In general, the system will work fine if the Nobo Hub is connected via a network cable to a wireless network router with default settings for home use.

Most routers have an included DHCP server enabled (which is a requirement). Also, most routers have no firewall between the wired LAN ports on the router and the wireless connections (which is also a requirement).

Please note that routers that have only one Ethernet-connection usually will not work, because this network port (often referred to as a WAN/Internet port) usually is intended for connecting the router to the Internet and not to internal network equipment such as Nobo Hub. There usually is a firewall between the wireless network and the WAN/Internet port on such routers.

Network requirement details for special network setups (usually only found in office buildings, advanced home network setups etc.):

- The wireless network (which is connected to the APP) must be on the same local area network (LAN) as the wired network (which is connected to the HUB). If this is not the case, the following traffic must be allowed between the wired and wireless networks:
  - TCP traffic must be allowed both ways on port 27779 for the app to talk to the hub.
  - For the APP to discover the HUB automatically, the network must allow UDP broadcast on port 10000 and/or UDP multicast on port 10001. If this is not possible or desirable, the HUB can be given a static IP-address from the router/DHCP server (via DHCP Reservation), and then connected via the “advanced” option in the APP setup wizard.
- HTTP TCP traffic on port 80 (standard HTTP port) must be allowed directly from the hub to the Internet, for the automatic update function to work. Nobo Hub's update function does not (so far) work with Web Proxy Servers.
- The hub requires a DHCP server on the network (this is usually implemented in the network router) to automatically get an IP address assigned.
- The network must allow Nobø Hub outgoing TCP traffic on port 27778 to be controlled via the Internet. Nobo Hub can be controlled via the Internet by wrapping the API described in this document in encryption and extra headers, and sending traffic via a server. Communication via the Internet is not described further in this document

5 LOCAL NETWORK TRAFFIC – GENERAL RULES

1. Network traffic is sent as plain text.
2. All commands sent to/from the Hub must end with a carriage return (ASCII char 13).
3. All strings are UTF-8 encoded.
4. All spaces in names (e.g. the name of a Zone) must be replaced with a different character when sent via the network to the Hub. Use UTF-8 non-breaking space: Replace bytes 32 (ASCII & UTF-8 space) with the two bytes 194 and 160 (UTF-8 non-breaking space).
5. All LAN traffic uses TCP port 27779, except for auto discovery of hubs (described in the next chapter).
6. After a successful “handshake”, the Hub will keep the connection alive as long as there is some network traffic at least every 30 seconds. After 30 seconds without activity, the connection is closed, and a new handshake must be performed. To keep a connection alive, a separate “KEEPALIVE” (followed by carriage return) command can be sent to the Hub. We recommend sending a keepalive request once every 14 seconds as long as you want to remain connected.
7. Please do not cache information stored on the Hub between sessions. This means that if you perform a new handshake, you also need to download all information from the Hub again (with the G00 command).
8. Two devices can be connected directly via LAN to one Hub at the same time. (In addition, up to 10 devices can be simultaneously connected via the Internet.) Devices that are connected must be prepared to receive updates about added/modified/deleted information. This can be caused e.g. by an Override ending, a button on a Nobo Switch being pressed or another user modifying data from a connected app.

6 AUTO DISCOVERY OF HUBS

Every two seconds, the Hub sends one UDP broadcast packet on port 10000 to broadcast IP 255.255.255.255, and one UDP multicast packet on port 10001 to multicast IP 239.0.1.187.

You can use this information to discover Hubs on the network and identify their current IP address and the first 9 of the 12 digits of the Hub’s unique serial number.[1]

The data that is sent from the Hub is the following ASCII string:
“__NOBOHUB__123123123”, where 123123123 is replaced with the first 9 digits of the Hub’s serial number. There are two underscore characters “_” before and after the “NOBOHUB” string. This string is NOT ended with a carriage return.

From time to time the Hub will reboot (automatically approximately every 18 hours) or re-run DHCP to acquire a new IP address from the DHCP server on the network (usually found at the network router). If connection to a Hub is lost, listen for these broadcast/multicast messages to identify the new IP address of the Hub. If several Hubs on the network have the same first 9 digits in their serial number, you have to try connecting to each one until you find the correct one.

[1] Sending 9 of 12 digits of the serial number is not intended to be a security measure, but a way for users to be explicit about which Hub they wat to connect to, to avoid possible confusion when using several Hubs in the same network.

7 HANDSHAKE

This chapter describes the process of establishing a session between a Hub (H) and an application (A). When connecting on the local network (LAN), this is TCP network traffic on a network socket on port 27779:

- A: "HELLO <its version of command set> <Hub s.no.> <date and time in format 'yyyyMMddHHmmss'>\r"
- H: "HELLO <its version of command set>\r", or "REJECT <reject code>\r".
  Reject code
  0=client command set version too old (or too new!).
  1=Hub serial number mismatch.
  2=Wrong number of arguments.
  3=Timestamp incorrectly formatted
- A: Sends “REJECT\r” if command set is not supported. OR sends "HANDSHAKE\r"
- H: "HANDSHAKE\r"
- Now connection is established!

“\r” = a carriage return.

Example:
A: “HELLO 1.1 102000000123 20131220092040\r”
H: “HELLO 1.1\r”
A: “HANDSHAKE\r”
H: “HANDSHAKE\r”

Immediately following a successful handshake, the application should send a “G00\r” command to download all relevant information from the Hub. The application knows that a G00 command is finished when the Hub sends the H05 (hub info) command to the application. See the next chapter for more info.

8 COMMAND SET COMPATIBILITY

The command set version is described by major version X and minor version Y in the format "X.Y". New minor versions will always be backwards-compatible with older minor versions. To ensure this backwards-compatibility, two things are required:

1. You must implement a client to **always ignore unknown commands** (if for example a Y02 command is sent from a hub to a client implemented with the 1.0 version of the API, it will be ignored).
2. More parameters (separated by space) in each command/reply received from the Hub must be allowed (but should be ok to ignore), to support extensions of the API in the future.

We believe that we will never have to update to an incompatible command set version (2.0). That is at least our goal. Glen Dimplex Nordic reserves the right to modify the API in non-backwards compatible ways, but we will try to avoid this, and inform our official partners as soon as possible if we do.

9 COMMAND SET

Please see the end of this document for the commands that are sent to and from the hub.

Feature limitations:
The following functionality is not yet implemented/not supported on the Hub or are limited. Please conform to the following rules:

- **Hub** Snr, ActiveOverrideId, SoftwareVersion, HardwareVersion and ProductionDate are read only fields.
- **Zone** ID is a read only field.
- **Zone** Active override id is always -1 (deprecated field).
- **Component** Serial number cannot be modified – the Component must in that case be removed and added with the new serial number.
- **Component** Active override id is always -1 (local component overrides are not implemented).
- **Component** Status field is not yet in use. Always write a “0” to this field.
- **Week Profile** ID is a read only field.
- **Week Profile** “Profile” can contain maximum 672 timestamps. Timestamps must have 00, 15, 30 or 45 for the minute field to be compatible with the app interface that only allows 15 minute intervals.
- **Override** ID is a read only field.
- **Override** Target must be “0” or "1", and Override target ID must be “-1” (for global hub overrides) or Zone's ID (for local zone overrides). Local Component overrides are not implemented.

Definitions of “status” when describing Week Profiles:
The last digit in a Week Profile “Profile” is coded as follows:

- 0: Eco mode
- 1: Comfort mode
- 2: Anti frost mode (a.k.a. "Away")
- 3: Off

Note regarding error messages:
The Hub might send error messages other than E00.

Note regarding string lengths:
Maximum string lengths are specified in bytes. Note that in the UTF-8 encoding, one character might be stored as more than one byte (1-6 bytes per character)!

Max lengths:

1. Hub name: 150 bytes.
2. Zone name: 100 bytes.
3. Component name: 100 bytes.
4. Week Profile name: 150 bytes.

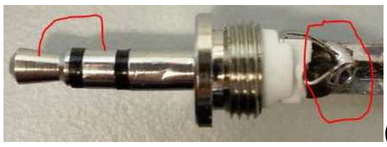
Please make sure that the Hub does not receive strings that are too long.

Note regarding IDs of new elements:
The Hub ignores the incoming IDs of new Zone, Week Profile or Override elements, and instead returns a newly assigned ID for the element. A dummy ID must still be sent with the command from the application that is creating the new element.

10 RESET FUNCTION

To reset the Hub, insert a special 2,5mm mini jack plug into the Hub, and plug in the power. The Hub is reset – all settings are set to default and all Components, Zones, Overrides and added Week Profiles are deleted. Hub Serial number, MAC address, Production date, Hardware version and Software version is not affected by this reset function.

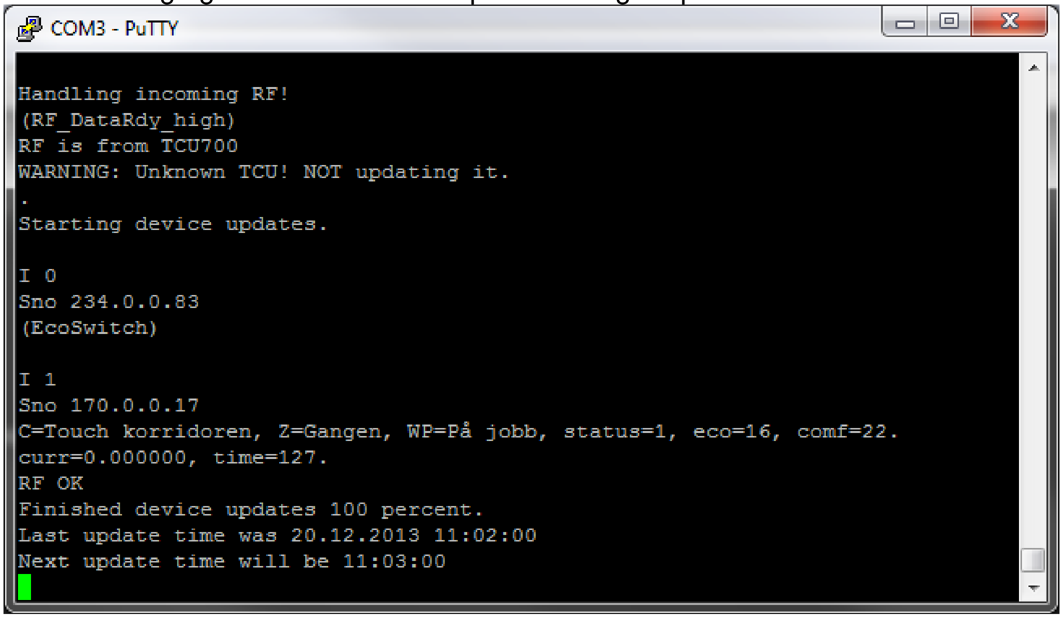
Reset plug: 2,5mm mini jack plug where the Tx and Rx are connected/soldered to each other, and the ground is left not connected:


(The two outmost connectors are soldered to each other.)

11 HOW TO READ DEBUG OUTPUT FROM A HUB

If you connect a micro USB cable from a PC to the Hub, you can read a lot of useful debug info.

The following figure shows an example of debug output from a Hub:



```
Handling incoming RF!
(RF_DataRdy_high)
RF is from TCU700
WARNING: Unknown TCU! NOT updating it.
.
Starting device updates.

I 0
Sno 234.0.0.83
(EcoSwitch)

I 1
Sno 170.0.0.17
C=Touch korridoren, Z=Gangen, WP=På jobb, status=1, eco=16, comf=22.
curr=0.000000, time=127.
RF OK
Finished device updates 100 percent.
Last update time was 20.12.2013 11:02:00
Next update time will be 11:03:00
```

To be able to read debug output from the Hub on a PC, do the following:

11.1 Install USB driver on the Windows computer

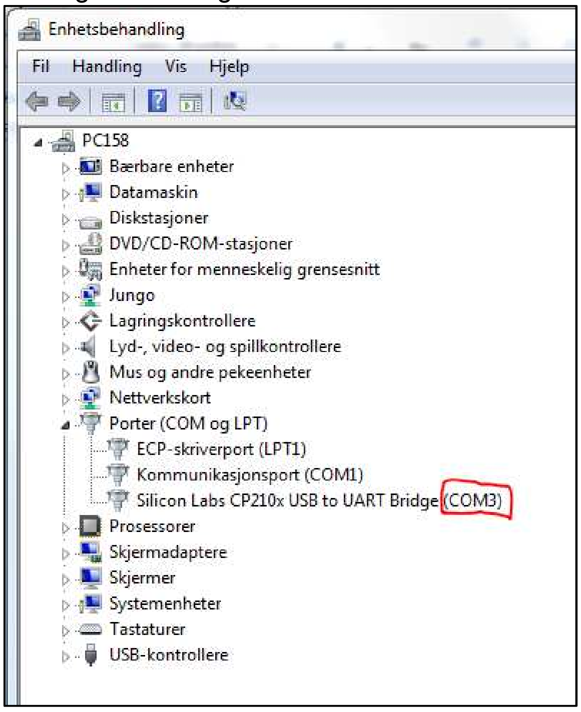
Download and install «CP210x USB to UART Bridge VCP Drivers» from
http://www.silabs.com/products/mcu/Pages/USBtoUARTBridgeVCPDrivers.aspx
(«The CP210x USB to UART Bridge Virtual COM Port (VCP) drivers are required for device operation as a Virtual COM Port to facilitate host communication with CP210x products. »)

11.2 Install PuTTY on Windows computer to read from USB port

Other programs can also be used, but we recommend «PuTTY». Download installation file from here:
http://www.putty.org/
or here
http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html.

11.3 Setup PuTTY to read from the USB port on the EC800 board

First, check which virtual COM port you need to connect to. You can see this in Windows Device Management. E.g. on the screenshot below it became COM3:



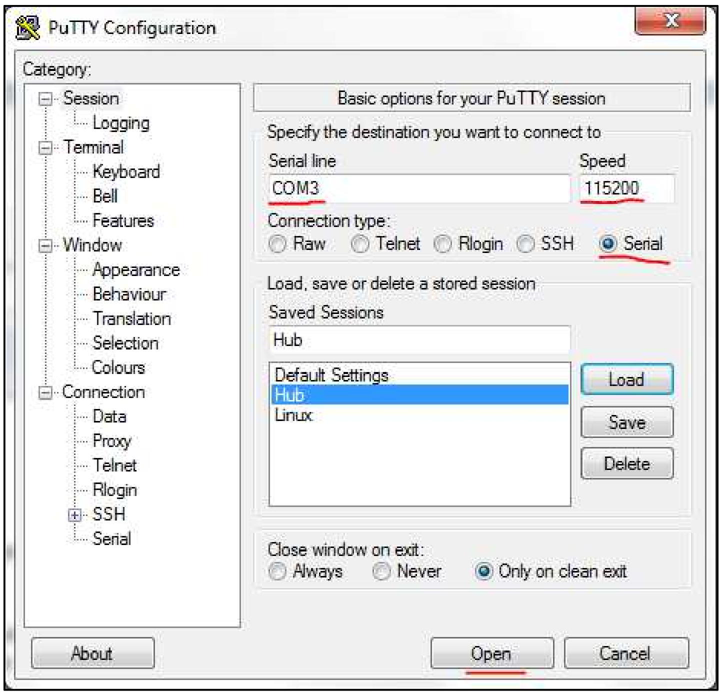
Enhetsbehandling

Fil Handling Vis Hjelp

- PC158
  - Bærbare enheter
  - Datamaskin
  - Diskstasjoner
  - DVD/CD-ROM-stasjoner
  - Enheter for menneskelig grensesnitt
  - Jungo
  - Lagringskontrollere
  - Lyd-, video- og spillkontrollere
  - Mus og andre pekeenheter
  - Nettverkskort
  - Porter (COM og LPT)
    - ECP-skriverport (LPT1)
    - Kommunikasjonsport (COM1)
    - Silicon Labs CP210x USB to UART Bridge (COM3)
  - Prosessorer
  - Skjermadaptere
  - Skjermer
  - Systemenheter
  - Tastaturer
  - USB-kontrollere

Start PuTTY, and setup the connection with

1. Connection type set to «Serial».
2. Serial line with the COM port (e.g. «COM1» or /»COM3»/»COM4» corresponding to your computer settings above.
3. Speed set to "921600" (or «115200» for hubs with very old firmware versions).
4. Click the «Open» button.



PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Serial line: COM3

Speed: 115200

Connection type: Raw Telnet Rlogin SSH [✓ Serial]

Load, save or delete a stored session

Saved Sessions: Hub

Default Settings, Hub, Linux

Close window on exit: Always Never [✓ Only on clean exit]

12 APPENDIX A: HUB API COMMAND SET

Please see the next pages.

Hub Command Set (version: 1.1)

Data structures

Hub

The command structure for hubs

Struct composition

- Snr

  Hub Serial Number

- Name

  Name of hub. Note that any spaces here is encoded as non-brake-space (ASCII character 160).

- DefaultAwayOverrideLength

  Default length over Away Override

- ActiveOverrideId

  ID of Active override. -1 if none.

- SoftwareVersion

  Software version the hub is runnning.

- HardwareVersion

  Hardware version of the hub. Alphanumeric plus '.'.

- ProductionDate

  Production date of the hub. Format: yyyyMMdd.

Struct composition

```
<Snr> <Name> <DefaultAwayOverrideLength> <ActiveOverrideId> <SoftwareVersion> <HardwareVersion> <ProductionDa
```

Struct data example

```
200045185039 name_of_hub 13 23 1.2 1f 20130813
```

Zone

The command structure for zones. Id is according to the hub's database. Temperatures are in celsius. NOTE: This structure does not include components that belong to this zone.

Struct composition

- Zone id
- Name

  Name of zone. Note that any spaces here is encoded as non-brake-space (ASCII character 160).

- Active week profile id

  ID of Active Week Profile. Must always have a valid value (never -1). The Hub has a read-only default Week Profile with ID=1.

- Comfort temperature

Degrees Celsius. Expected to be an integer in the range 7 to 30. The client should ensure that the Comfort temperature setpoint is at least as large as the Eco temperature.

- Eco temperature

  Degrees Celsius. Expected to be an integer in the range 7 to 30. The client should ensure that the Eco temperature setpoint is no larger than the Comfort temperature.

- Allow overrides

  1=Yes, 0=No.

- Active override id

  Deprecated. Always set to -1.

Struct composition

```
<Zone id> <Name> <Active week profile id> <Comfort temperature> <Eco temperature> <Allow
overrides> <Active override id>
```

Struct data example

```
2 ZoneName 1 23 15 0 -1
```

Component

The command structure for components.

Struct composition

- Serial number

  Serial number format is 'aaabbbcccddd'. Each group in the serial number (aaa, bbb, etc) is an 8 bit integer no higher than 255.

- Status

  Not yet implemented - always set to 0.

- Name

  Name of component. Note that any spaces here is encoded as non-brake-space (ASCII character 160).

- Reverse on/off?

  Only applicable for some components and should be 0 unless explicitly specified.

- ZoneId

  Id of the Zone the component is connected to. Value is -1 if no association.

- Active override Id

  Not yet implemented - always set to -1. (ID of Active override. -1 if none.)

- Temperature sensor for zone

  Id of the zone the component is used as a temperature sensor for. Value is -1 if no association.

Struct composition

```
<Serial number> <Status> <Name> <Reverse on/off?> <ZoneId> <Active override Id> <Temperature sensor for
zone>
```

Struct data example

```
200154035201 0 component_name 0 -1 23 2
```

WeekProfile

The command structure for week profiles

Struct composition

- Week profile id

  Id according to the Hub database.

- Name

  Name of week profile. Note that any spaces here is encoded as non-brake-space (ASCII character 160).

- Profile

  Comma separated list of time stamps (HHMM) where the program changes and what status it has in the following time period, e.g. 13451. The time stamp 0000 (midnight) has to be specified for each day no matter if the program changes or not, making a valid week profile containing exactly 7 occurances of these. NOTE: The status here is not the same as status for an override. Mapping of statuses: STATUS_ECO 0 <--- WEEK PROFILE STATUS 0 , STATUS_COMFORT 1 <--- WEEK PROFILE STATUS 1 , STATUS_FROST 2 <--- WEEK PROFILE STATUS 2 (AWAY), STATUS_OFF 3 <--- WEEK PROFILE STATUS 4 (OFF) .

Struct composition

```
<Week profile id> <Name> <Profile>
```

Struct data example

```
12 week_profile_name 00000,02154,13453,00001,08000,16301,00000,00001,00001,00002,00002
```

Override

The command structure for overrides.

Struct composition

- Id

  Id of override

- Mode

  Either 0(Normal), 1(Comfort), 2(Eco) or 3(Away)

- Type

  Type is what kind of an override it is: 0(Now), 1(Timer), 2(From-To) or 3(Constant)

- End time

  Only used when type is Timer or From-To (-1 in all other cases)

- Start time

  Only used when type is From-To (-1 in all other cases.)

- Override target

  Only global (Hub) and zone-local overrides are implemented yet - always set to 0 or 1. (The target for the override: 0 for hub, 1 for zone, 2 for component.)

- Override target ID

  The id for the override target. If override target is 0 (hub), use -1. If override target is 1 (zone), use the zone's ID. If override target is 2 (component), use component's 12 digit serial number. Component overrides (override target id = 2) are not yet supported.

Struct composition

```
<Id> <Mode> <Type> <End time> <Start time> <Override target> <Override target ID>
```

Struct data example

```
1 2 0 -1 -1 0 1
```

Commands

Add

A00

Adds a Zone to the hub's internal database.

Arguments

- zone : Zone

  Zone to add. Zone id and Active override id is ignored.

Command structure

```
A00 <Zone id> <Name> <Active week profile id> <Comfort temperature> <Eco temperature> <Allow
overrides> <Active override id>
```

Command example

```
A00 2 ZoneName 1 23 15 0 -1
```

Return

Either one of the following commands:

- B00

A01

Adds a Component to the hub's internal database.

Arguments

- component : Component

  Component to add. Active override id is ignored.

Command structure

```
A01 <Serial number> <Status> <Name> <Reverse on/off?> <ZoneId> <Active override Id> <Temperature sensor
for zone>
```

Command example

```
A01 200154035201 0 component_name 0 -1 23 2
```

Return

Either one of the following commands:

- B01

A02

Adds a WeekProfile to the hub's internal database.

Arguments

- weekprofile : WeekProfile

  Week Profile to add. Week Profile Id is ignored (it is decided by the Hub).

Command structure

```
A02 <Week profile id> <Name> <Profile>
```

Command example

```
A02 12 week_profile_name 00000,02154,13453,00001,08000,16301,00000,00001,00001,00002,00002
```

Return

Either one of the following commands:

- B02

A03

Adds an overrride to hub internal database.

Arguments

- override : Override

  Override to add. Override Id is ignored (it is decided by the Hub). Only global (Hub) and zone-local overrides are supported currently.

Command structure

```
A03 <Id> <Mode> <Type> <End time> <Start time> <Override target> <Override target ID>
```

Command example

```
A03 1 2 0 -1 -1 0 1
```

Return

Either one of the following commands:

- B03

B00

Response from Hub after an A00 command.

Arguments

- zone : Zone

  the added zone

Command structure

```
B00 <Zone id> <Name> <Active week profile id> <Comfort temperature> <Eco temperature> <Allow
overrides> <Active override id>
```

Command example

```
B00 2 ZoneName 1 23 15 0 -1
```

B01

Response from Hub after an A01 command.

Arguments

- component : Component

  the added component

Command structure

```
B01 <Serial number> <Status> <Name> <Reverse on/off?> <ZoneId> <Active override Id> <Temperature sensor
for zone>
```

Command example

```
B01 200154035201 0 component_name 0 -1 23 2
```

B02

Response from Hub after an A02 command.

Arguments

- weekprofile : WeekProfile

  the added weekprofile

Command structure

```
B02 <Week profile id> <Name> <Profile>
```

Command example

```
B02 12 week_profile_name 00000,02154,13453,00001,08000,16301,00000,00001,00001,00002,00002
```

B03

Response from Hub after an A03 command.

Arguments

- override : Override

  the added override

Command structure

```
B03 <Id> <Mode> <Type> <End time> <Start time> <Override target> <Override target ID>
```

Command example

```
B03 1 2 0 -1 -1 0 1
```

Update

U00

Updates a Zone in the hub's internal database.

Arguments

- zone : Zone

  zone to update

Command structure

```
U00 <Zone id> <Name> <Active week profile id> <Comfort temperature> <Eco temperature> <Allow
overrides> <Active override id>
```

Command example

```
U00 2 ZoneName 1 23 15 0 -1
```

Return

Either one of the following commands:

- V00

U01

Updates a Component in the hub's internal database.

Arguments

- component : Component

  component to update

Command structure

```
U01 <Serial number> <Status> <Name> <Reverse on/off?> <ZoneId> <Active override Id> <Temperature sensor
for zone>
```

Command example

```
U01 200154035201 0 component_name 0 -1 23 2
```

Return

Either one of the following commands:

- V01

U02

Updates a WeekProfile in the hub's internal database.

Arguments

- weekprofile : WeekProfile

  weekprofile to update

Command structure

```
U02 <Week profile id> <Name> <Profile>
```

Command example

```
U02 12 week_profile_name 00000,02154,13453,00001,08000,16301,00000,00001,00001,00002,00002
```

Return

Either one of the following commands:

- V02

U03

Updates hub information

Arguments

- hub : Hub

  hub data

Command structure

```
U03 <Snr> <Name> <DefaultAwayOverrideLength> <ActiveOverrideId> <SoftwareVersion> <HardwareVersion> <Productio
```

Command example

```
U03 200045185039 name_of_hub 13 23 1.2 1f 20130813
```

Return

Either one of the following commands:

- V03

U06

Update hub internet connectivity function and encryption key. This command only works on LAN connections

Arguments

- Enable internet access

  0: no 1: yes (default)

- Encryption key

  128-bit unsigned number, encoded as 16 8-bit unsigned numbers. Must be set by a client before allowing internet access. Should be set to all zeroes (16 zeroes) if disabling Internet access, to keep things tidy.

- Reserved 1

  Not yet in use, always set to 0.

- Reserved 2

  Not yet in use, always set to 0.

Command structure

```
U06 <Enable internet access> <Encryption key> <Reserved 1> <Reserved 2>
```

Command example

```
U06 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 0 0
```

Return

Either one of the following commands:

- V06

V00

Reponse from hub on U00 command.

Arguments

- zone : Zone

  updated zone

Command structure

```
V00 <Zone id> <Name> <Active week profile id> <Comfort temperature> <Eco temperature> <Allow
overrides> <Active override id>
```

Command example

```
V00 2 ZoneName 1 23 15 0 -1
```

V01

Response from hub on U01 command

Arguments

- component : Component

  updated component

Command structure

```
V01 <Serial number> <Status> <Name> <Reverse on/off?> <ZoneId> <Active override Id> <Temperature sensor
for zone>
```

Command example

```
V01 200154035201 0 component_name 0 -1 23 2
```

V02

Response from hub on U02 command

Arguments

- weekprofile : WeekProfile

  updated weekprofile

Command structure

```
V02 <Week profile id> <Name> <Profile>
```

Command example

```
V02 12 week_profile_name 00000,02154,13453,00001,08000,16301,00000,00001,00001,00002,00002
```

V03

Response from hub on U03 command

Arguments

- hub : Hub

updated hub static information

Command structure

```
V03 <Snr> <Name> <DefaultAwayOverrideLength> <ActiveOverrideId> <SoftwareVersion> <HardwareVersion> <Producti
```

Command example

```
V03 200045185039 name_of_hub 13 23 1.2 1f 20130813
```

V06

Hub info sent from a Hub regarding if Internet connection is enabled and encryption key. Will only work if the socket is a local socket (LAN, not via Internet). Sends all zeroes for the encryption key if Internet access is not allowed or allowed but not initialized (a client has not yet set the encryption key).

Arguments

- Enable internet access

  0: no 1: yes (default)

- Encryption key

  128-bit unsigned number, encoded as 16 8-bit unsigned numbers. Must be set by a client before allowing internet access.

- Reserved 1

  Not yet in use, always set to 0.

- Reserved 2

  Not yet in use, always set to 0.

Command structure

```
V06 <Enable internet access> <Encryption key> <Reserved 1> <Reserved 2>
```

Command example

```
V06 1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 0 0
```

Remove

R00

Removes a Zone from the hub's internal database. All values except Zone ID is ignored. Any Components in the Zone are also deleted (and S02 Component deleted messages are sent for the deleted Components). Any Component used as temperature sensor for the Zone is modified to no longer be temperature sensor for the Zone (and a V01 Component updated message is sent).

Arguments

- zone : Zone

  zone to remove

Command structure

```
R00 <Zone id> <Name> <Active week profile id> <Comfort temperature> <Eco temperature> <Allow
overrides> <Active override id>
```

Command example

```
R00 2 ZoneName 1 23 15 0 -1
```

Return

Either one of the following commands:

- S00

R01

Removes a Component from the hub's internal database. All values except Component Serial Number is ignored.

Arguments

- component : Component

  component to removes

Command structure

```
R01 <Serial number> <Status> <Name> <Reverse on/off?> <ZoneId> <Active override Id> <Temperature sensor
for zone>
```

Command example

```
R01 200154035201 0 component_name 0 -1 23 2
```

Return

Either one of the following commands:

- S01

R02

Removes a WeekProfile from the hub's internal database. All values except Week Profile ID is ignored. Any Zones that are set to use the Week Profile are set to use the default Week Profile in stead (and V00 Zone updated messages are sent).

Arguments

- weekprofile : WeekProfile

  weekprofile to remove

Command structure

```
R02 <Week profile id> <Name> <Profile>
```

Command example

```
R02 12 week_profile_name 00000,02154,13453,00001,08000,16301,00000,00001,00001,00002,00002
```

Return

Either one of the following commands:

- S02

S00

Response from hub on R00 command. All values except Zone ID should be ignored.

Arguments

- zone : Zone

removed zone

Command structure

```
S00 <Zone id> <Name> <Active week profile id> <Comfort temperature> <Eco temperature> <Allow
overrides> <Active override id>
```

Command example

```
S00 2 ZoneName 1 23 15 0 -1
```

S01

Response from hub on R01 command. All values except Component Serial Number should be ignored.

Arguments

- component : Component

  removed component

Command structure

```
S01 <Serial number> <Status> <Name> <Reverse on/off?> <ZoneId> <Active override Id> <Temperature sensor
for zone>
```

Command example

```
S01 200154035201 0 component_name 0 -1 23 2
```

S02

Response from Hub on R02 command. All values except Week Profile ID should be ignored. In fact, week profile time and status data is just example data in this S02 message.

Arguments

- weekprofile : WeekProfile

  removed weekprofile

Command structure

```
S02 <Week profile id> <Name> <Profile>
```

Command example

```
S02 12 week_profile_name 00000,02154,13453,00001,08000,16301,00000,00001,00001,00002,00002
```

S03

Sent from hub when an override is no longer valid and is deleted. All fields except Override ID should be ignored.

Arguments

- override : Override

  the invalidated override

Command structure

```
S03 <Id> <Mode> <Type> <End time> <Start time> <Override target> <Override target ID>
```

Command example

```
S03 1 2 0 -1 -1 0 1
```

Get

G00

Gets all information from hub. Will trigger a sequence of series of one H00 message, zero or more H01, zero or more H02, zero or more Y02, zero or more H03, zero or more H04 commands, one V06 message if connected via LAN (not Internet), and lastly a H05 message. The client knows that the Hub is finished sending all info when it has received the H05 message.

Command structure

```
G00
```

Command example

```
G00
```

Return

Either one of the following commands:

- H00
- H01
- H02
- Y02
- H03
- H04
- V06
- H05

G01

(Never used by the Nobo Energy Control app - you should only use G00.) Gets all Zones from hub. Will trigger a series of H01 messages from the Hub.

Command structure

```
G01
```

Command example

```
G01
```

Return

Either one of the following commands:

- H01

G02

(Never used by the Nobo Energy Control app - you should only use G00.) Gets all Components from hub. Will result in a series of H02 messages from the Hub.

Command structure

```
G02
```

Command example

```
G02
```

Return

Either one of the following commands:

- H02

G03

(Never used by the Nobo Energy Control app - you should only use G00.) Gets all WeekProfile data from hub. Will trigger a series of H03 messages from the Hub.

Command structure

```
G03
```

Command example

```
G03
```

Return

Either one of the following commands:

- H03

G04

(Never used by the Nobo Energy Control app - you should only use G00.) Gets all active overrrides from hub. Will trigger a series of H04 messages from the Hub.

Command structure

```
G04
```

Command example

```
G04
```

Return

Either one of the following commands:

- H04

H00

Response to G00 signifying that all relevant info stored in Hub is about to be sent.

Command structure

```
H00
```

Command example

```
H00
```

H01

Response to G00 or G01 containing one Zone.

Arguments

- zone : Zone

  Zone registered on the hub.

Command structure

```
H01 <Zone id> <Name> <Active week profile id> <Comfort temperature> <Eco temperature> <Allow
overrides> <Active override id>
```

Command example

```
H01 2 ZoneName 1 23 15 0 -1
```

H02

Response to G00 or G02 containing one Component.

Arguments

- component : Component

  Component registered on the hub.

Command structure

```
H02 <Serial number> <Status> <Name> <Reverse on/off?> <ZoneId> <Active override Id> <Temperature sensor
for zone>
```

Command example

```
H02 200154035201 0 component_name 0 -1 23 2
```

H03

Response to G00 or G03 containing one WeekProfile

Arguments

- weekprofile : WeekProfile

  Week profile

Command structure

```
H03 <Week profile id> <Name> <Profile>
```

Command example

```
H03 12 week_profile_name 00000,02154,13453,00001,08000,16301,00000,00001,00001,00002,00002
```

H04

Response to G00 or G04 containing one Override

Arguments

- override : Override

  Override

Command structure

```
H04 <Id> <Mode> <Type> <End time> <Start time> <Override target> <Override target ID>
```

Command example

```
H04 1 2 0 -1 -1 0 1
```

H05

Response to G00 with the static information of the hub. Also signifies the end of a G00 request.

Arguments

- hub : Hub

  hub

Command structure

```
H05 <Snr> <Name> <DefaultAwayOverrideLength> <ActiveOverrideId> <SoftwareVersion> <HardwareVersion> <Productio
```

Command example

```
H05 200045185039 name_of_hub 13 23 1.2 1f 20130813
```

Execute

X00

Starts receiver search. Currently automatically stops after 30 seconds if not aborted sooner by a X01 message. Results in a cascade of Y04 commands (for every new Component found) after Y00 is sent.

Command structure

```
X00
```

Command example

```
X00
```

Return

Either one of the following commands:

- Y00
- Y04

X01

Stops receiver search

Command structure

```
X01
```

Command example

```
X01
```

Return

Either one of the following commands:

- Y01

X03

Tells the hub to try to pair with a component with the given serial number. Similar to X00, but only accepts the specified serial number. Use this for Nobo Switch SW4, TCU or Nobo Sense MSW if the user try to add these devices "manually" even if they in reality only supports autosearch.

Arguments

- snr

  Serial number for the component to pair with

Command structure

```
X03 <snr>
```

Command example

```
X03 134250001098
```

Return

Either one of the following commands:

- Y03

Y00

Receiver autosearch started

Command structure

```
Y00
```

Command example

```
Y00
```

Y01

Receiver autosearch stopped

Command structure

```
Y01
```

Command example

```
Y01
```

Y02

Component temperature value sent as part of a G00 response, or pushed from the Hub automatically to all connected clients whenever the Hub has received updated temperature data.

Arguments

- serial number

  Serial number of the component (12 digits).

- temp

  Current temperature. A decimal number, or the string "N/A" if the temperature value is not available (typically because the temperature value stored at the Hub has become too old and outdated).

Command structure

```
Y02 <serial number> <temp>
```

Command example

```
Y02 234000012006 24.125
```

Y03

Reply command to X03 with the serial number of the component and a status on whether the pairing was successful or not.

Arguments

- snr

  serial number.

- success

  status. 1 if success, 0 if not.

Command structure

```
Y03 <snr> <success>
```

Command example

```
Y03 230000008118 1
```

Y04

Command sent with the serial number of a component that was found during a search (that was started with X00)

Arguments

- snr

  serial number.

Command structure

```
Y04 <snr>
```

Command example

```
Y04 230000008118
```

Errors

E00

Error message. Please note: Other error messages than E00 may also be sent from the Hub (E01, E02 etc.).

Arguments

- command

  The command that this error message is about

- message

  Error message. Note: This string may contain normal space characters!

Command structure

```
E00 <command> <message>
```

Command example

```
E00 U06 Wrong number of/format of arguments
```