Basic / **Advanced** / Expert

Configuration Guide

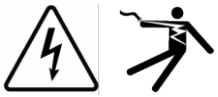
NAC CoolMasterNet Module

Using a NAC/SHAC to integrate to a CoolMasterNet Thermostat

NAC/SHAC CoolMasterNet Module
CB-CON-xx-1.0

Safety Information

Important Information



Read these instructions carefully before trying to install, configure, or operate this software. The following special messages may appear throughout this bulletin or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.

The addition of either symbol to a “Danger” or “Warning” safety label indicates that an electrical hazard exists which will result in personal injury if the instructions are not followed.

This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

⚠DANGER
DANGER indicates an imminently hazardous situation which, if not avoided, will result in death or serious injury.

⚠WARNING
WARNING indicates a potentially hazardous situation which, if not avoided, can result in death or serious injury.

⚠CAUTION
CAUTION indicates a potentially hazardous situation which, if not avoided, can result in minor or moderate injury.

NOTICE
NOTICE is used to address practices not related to physical injury. The safety alert symbol shall not be used with this signal word.

Please Note

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction, installation, and operation of electrical equipment and has received safety training to recognize and avoid the hazards involved.

Safety Precautions

⚠WARNING

HAZARD OF INCORRECT INFORMATION

- Do not incorrectly configure the software, as this can lead to incorrect reports and/or data results.
- Do not base your maintenance or service actions solely on messages and information displayed by the software.
- Do not rely solely on software messages and reports to determine if the system is functioning correctly or meeting all applicable standards and requirements.
- Consider the implications of unanticipated transmission delays or failures of communications links.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

The information provided in this documentation contains general descriptions and/or technical characteristics of the performance of the products contained herein. This documentation is not intended as a substitute for and is not to be used for determining suitability or reliability of these products for specific user applications. It is the duty of any such user or integrator to perform the appropriate and complete risk analysis, evaluation and testing of the products with respect to the relevant specific application or use thereof. Neither Schneider Electric nor any of its affiliates or subsidiaries shall be responsible or liable for misuse of the information that is contained herein. If you have any suggestions for improvements or amendments or have found errors in this publication, please notify us.

All pertinent state, regional, and local safety regulations must be observed when installing and using

this product. For reasons of safety and to help ensure compliance with documented system data, only the manufacturer should perform repairs to components.

When devices are used for applications with technical safety requirements, the relevant instructions must be followed.

Failure to use Schneider Electric software or approved software with our hardware products may result in injury, harm, or improper operating results.

Failure to observe this information can result in injury or equipment damage.

Revision Number	Created by	Date
1.0	Project Services AU	04/10/2018

Table of Contents

Contents

1 Introduction

This configuration guide describes the process of setting up a C-Bus Network Automation Controller (5500NAC) with the CoolMasterNet module. This module integrates to the CoolMasterNet thermostat via ethernet using Lua inside the 5500NAC/SHAC.

Competencies

This document is intended for readers who have been trained on 5500NAC products. The configuration should not be attempted by someone who is new to the installation of the C-Bus system or the 5500NAC. In addition, it is mandatory to have a knowledge of C-Bus commissioning through the use of C-Bus Toolkit and basic knowledge in Lua scripting language.

Familiarity with the CoolMasterNet product range is also advised. This guide only goes into connecting the NAC/SHAC to a CoolMasterNet, and does not cover off configuration of the thermostat.

System Prerequisites

Software Version	Version	Download link
C-Bus Toolkit for NAC	1.15.0	https://www.clipsal.com/Trade/Support/Software

*To program a 5500NAC/SHAC you will require C-Bus Toolkit 1.15.0 or higher. This software includes the USB drivers needed to utilize the USB to ethernet interface provided on the front of the NAC/SHAC. It also includes the option to export to CGL file which is required to import any C-Bus tags.

2 Configuration

2.1 NAC/SHAC CoolMasterNet Module

The NAC/SHAC CoolMasterNet module supports the following functions:

Zone On/Off

Set Point Temperature Control

Mode Control

Fan Speed Control

Swing Mode Control

Zone Feedback

2.2 Importing CoolMasterNet module into NAC/SHAC

This section describes the steps needed to setup 5500NAC using a web browser.

Uploading the script library onto 5500NAC/SHAC

a) Log into the 5500NAC/SHAC configuration screen on the web browser.
b) Click on the 'Configurator' button to log into the configuration screen.
c) Click on the 'Scripting' tab, 'Tools' button and 'Restore scripts'.
d) Select "Append keeping existing scripts" to prevent losing any existing scripts then, using the Choose File option, open 'Scripting-5500NAC CoolMasterNet.tar.gz'

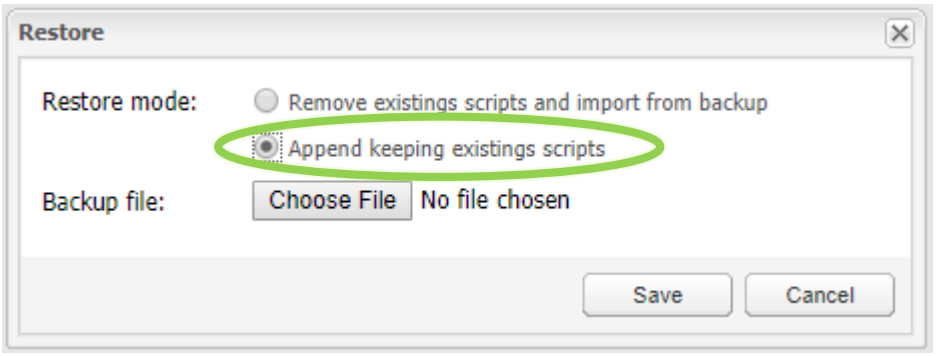


Figure 1: Restoring Script

e) After the 5500NAC restarts, you will find under user libraries: user.coolmasterCore and user.hvacConfig.

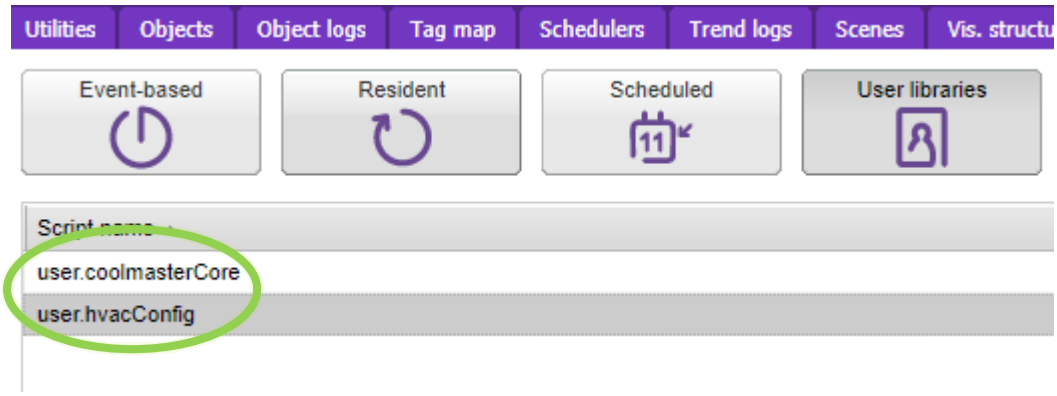


Figure 2: User libraries

f) Under Event-based, you will also find a list of example scripts, these scripts demonstrate how to use the CoolMasterNet Module.

This now should complete the importing process for the CoolMasterNet module. The next part of this guide will show how to configure the user.hvacConfig user library.

3 How to configure user.hvacConfig user library

Now the CoolMasterNet module has been imported, the first step is to configure the IP address for where the NAC/SHAC will find the CoolMasterNet thermostat.

Note: Please refer to the CoolMasterNet manual for information on how to configure an IP address within the unit.

Under User Libraries, open the editor for user.hvacConfig and modify line 3 to the correct IP address. Example hvac_ipAddress = '192.168.0.219'

List the AC unit's address in an array of strings for example modify the line 7 with right unit address from CoolMasterNet module.

Note: Do not add extra characters or space, please find the below example.

Example hvac_units = {'L2.104','L2.102','L2.103'}

Provide the NAC application address assigned for CoolMasterNet module in line 10.

Example coolmaster_appAddress = 10

Provide the NAC group address start assigned for CoolMasterNet module in line 13.

Example coolmaster_groupAddressStart = 0

4 Objects

Order of HVAC control objects should follow the same order of HVAC units specified in *user.havcConfig* user library and also objects should strictly follow the order of precedence. Please refer to the sample project.

Example: *hvac_units = {'L2.104','L2.102','L2.103'}*

First set of objects should be for L2.104, second set for L2.102 and third for L2.103.

Order of precedence of objects must be as following:

1) Power On/Off control for first unit (ex: **0/10/0**)
2) Temperature control for first unit (ex: **0/10/1**)
3) Fan control for first unit (ex: **0/10/2**)
4) Mode control for first unit (ex: **0/10/3**)
5) Swing control for first unit (ex: **0/10/4**)

Follow the same pattern for all other units to be controlled

User Parameters

Two sets of user parameters can be provided for one unit.

1) CoolMaster-SetTemp _x
2) CoolMaster-RoomTemp_x

x is the number of order.

Example: *hvac_units = {'L2.104','L2.102','L2.103'}*

CoolMaster-SetTemp _1 and *CoolMaster-RoomTemp_1* for *L2.104*

CoolMaster-SetTemp _2 and *CoolMaster-RoomTemp_2* for *L2.102*

CoolMaster-SetTemp _3 and *CoolMaster-RoomTemp_3* for *L2.102*

User parameters also must follow the order of configuration of units in user.hvacConfig user library. User parameter name must follow strict naming convention except the trailing number of order as described in the example above.

5 Event Based Scripts

5.1 Event-based script examples

This Section outlines the different examples provided when the coolmaster module is imported into the NAC or SHAC.

Example 1: Event-based script – Toggle Power

The objective of this script is to toggle the power of HVAC unit configured. User can copy paste this section of code for controlling the power of one single unit.

Level 0 to turn Off and 255 to turn On the HVAC unit.

In this case, the code for this is as follows:

```
require("user.coolmasterCore")
require("user.hvacConfig")

network, application, group = event.dst:match("([%d,]+)/([%d,]+)/([%d,]+)")
level = GetCBusLevel(network, application, group)

--update input parameters according to the project settings: See the example
--*      group = x                    (number) group address of object      *--
--*      level                              (number)  0 - OFF/255 - ON    *--

aircon:TogglePower(group,level)
```

Example 2: Event-based script – Temperature control

The objective of this script is to control the temperature of one single unit. Here user can copy paste this section of code and may specify the *stepby* value which is to increment or decrement the temperature by the value provided in *stepby*.

Level 0 to decrement and level 255 to increment

In this case, the code for this is as follows:

```
require("user.coolmasterCore");
require("user.hvacConfig")

network, application, group = event.dst:match("([%d,]+)/([%d,]+)/([%d,]+)")
level = GetCBusLevel(network, application, group)

--*update input parameters according to the project settings: See the example
--*      group = x                    (number) group address of object      *--
--*      stepby = x                   (number)                               --*
 --*     level = x                    (number)                               *--
local stepby = 1;
aircon:TemperatureControl(group,stepby,level)
```

Example 3: Event-based script – Fan speed select

The objective of this script is to select the fan speed of single HVAC unit. Here user can copy paste this section of code. Each mode is designated with specific object levels.

10 – auto

20 – low

30 – med

40 – high

In this case, the code for this is as follows:

```
require("user.coolmasterCore")
require("user.hvacConfig")

network, application, group = event.dst:match("([%d,]+)/([%d,]+)/([%d,]+)")
level = GetCBusLevel(network, application, group)
--update input parameters according to the project settings: See the example
--*      group = x                    (number) group address of object      *--
--*      level = x                (number)    10-auto                              *--
--*                                            20-low                               *--
--*                                            30-med                        *--
--*                                           40-high                               *--
aircon:SelectFanSpeed(group,level)
```

Example 4: Event-based script – Mode Select

The objective of this script is to select the HVAC operating mode of single unit. Here user can copy paste this section of code. Each mode is designated with specific object levels.

10 – heat

20 – cool

30 – fan

40 – dry

50 – auto

In this case, the code for this is as follows:

```
require("user.coolmasterCore")
require("user.hvacConfig")

network, application, group = event.dst:match("([%d,]+)/([%d,]+)/([%d,]+)")
level = GetCBusLevel(network, application, group)

--update input parameters according to the project settings: See the example
--*      group = x                                  (number) group address of object      *--
--*      level = x                  (number)   10-heat                                    *--
--*                                            20-cool                                    *--
--*                                            30-fan                                  *--
--*                                            40-dry                                  *--
--*                                            50-auto                                  *--
aircon:SelectMode(group,level)
```

Example 5: Event-based script – Swing control

The objective of this script is to select the swing mode of single unit. Here user can copy paste this section of code. Each mode is designated with specific object levels.

10 – no swing

20 – horizontal

30 – vertical

40 – auto

50 – 30°

60 – 45°

70 – 60°

In this case, the code for this is as follows:

```
require("user.coolmasterCore");
require("user.hvacConfig")

network, application, group = event.dst:match("([%d,]+)/([%d,]+)/([%d,]+)")
level = GetCBusLevel(network, application, group)

--update input parameters according to the project settings: See the example
--*      group = x                                    (number) group address of object      *--
--*      level = x                  (number)  10- no swing                                  *--
--*                                            20-horizontal                                 *--
--*                                           30-vertical                                    *--
--*                                           40-auto                                        *--
--*                                           50-30°                                              *--
--*                                           60-45°                                              *--
--*                                           70-60°                                              *--
aircon:SelectSwingMode(group,level)
```

6 Resident Script

Resident script is used to update the visualization based on the status of the HAVC units configured. Resident script polls *UpdateUI* function in a loop of specified seconds. Default time configured on sample project is 5 seconds.

7 CoolMaster Function usage

TogglePower(group,level)

Parameter	Acceptable Variable Types	Valid Input Range	Default Value
group	Int	0-254	
level	Int	0 Or 255	

TemperatureControl(group,stepby,level)

Parameter	Acceptable Variable Types	Valid Input Range	Default Value
group	Int	String or 0-254	
stepby	Int	1 to any	1
level	Int	0 Or 255	

SelectFanSpeed(group,level)

Parameter	Acceptable Variable Types	Valid Input Range	Default Value
group	Int	0-254	
level	Int	Or 255	

SelectMode(group,level)

Parameter	Acceptable Variable Types	Valid Input Range	Default Value
group	Int	0-254	
level	Int	Or 255	

SelectSwingMode(group,level)

Parameter	Acceptable Variable Types	Valid Input Range	Default Value
group	Int	0-254	
level	Int	Or 255	

8 CoolMaster Sample Project

Notes

This Section outlines notes about the provided sample project “CoolMaster Example Project.tar.gz”

When importing the sample project, this will override any existing project on the unit. Any previous configuration will be lost.

If you require just the code, make sure to follow the steps provided in Step 2 of this document

The sample project includes a list of objects which are referred to from the example Event-based scripts.

Included is a 3-visualization pages allowing for the provided examples to be tested.

Sample project was built on a NAC running firmware 1.6.0

9 Miscellaneous functions

TogglePowerAll(level) - Power toggle all the units

TogglePowerLine(line,level) – Power Toggle a specified line

GetSystemStatus(line,unit) – Get the system status of a single unit

10 Conclusion

This configuration guide describes how the NAC Application CoolMaster script can be used and what options are available to the user when leveraging it. The provided examples show how the CoolMaster module can be used. The sample project includes a working example of this script including this visualization aspect of it.

Schneider Electric (Australia) Pty Ltd	Schneider Electric (New Zealand) Ltd
Customer Care Australia: Phone 1300 369 233 Email: customercare.au@schneider-electric.com 33 – 37 Port Wakefield Rd Gepps Cross, South Australia Australia, 5094	Customer Care New Zealand Phone 0800 652 999 Email: sales@nz-schneider-electric.com 38 Business Parade South East Tamaki 2013 Auckland, **New Zealand**