

Phc2Mqtt E2E Config v4.1.x.y Ed01

1 Terms and abbreviations

LAN	Local Area Network, is the network form typically used at home, supporting Ethernet connections.
WLAN	Wireless LAN, is the same as LAN but then the wireless variant, also called Wifi.
IP	Internet Protocol, the basis networking protocol that runs over Ethernet (wired) or Wifi (wireless).
TCP	Transmission Control Protocol, runs on top of IP and is a connection oriented protocol.
HTTP	HyperText Transfer Protocol, runs on top of TCP in a request-reponse model.
XMLRPC	XML encoded Remote Procedure Call, this protocol is used by the PHC Systemsoftware v3 to communicate and configure an STMv3. The commands are encoded in XML and the request is sent over HTTP.
SNTP	Simple Network Time Protocol, protocol used to obtain time from the Internet.
MQTT	A compact telemetry protocol typically used in IoT applications, runs over TCP.
PCB	Printed Circuit Board, a non-conductive material with copper patterns on 1 or 2 sides to interconnect electronic components.
ESP32 DevkitC	The ESP32 dual core module that is used for Phc2Mqtt, we use the WROVER version with 4Mb external memory.
AP-mode	Access Point mode, the module acts as an access point to which you can connect with a Wifi enabled PC.
STA-mode	Station mode, the module connects to a WLAN AP as a station, you can access the module via the WLAN network.
OTA	Over-The-Air, refers to firmware upgrade by simple file upload, as such you don't need to open the module or use specialized hardware.
PHC	PEHA Home Control, a domotic system created by the former company PEHA.
STM	Steering module, the central processor in a PHC system.
Ccmd	Compound-command, the command format as used by xWRC to send commands to a PHC system.
3MOD	3 Module, indicates the physical size of a storage container mounted onto a (DIN) rail, where 1 MOD = 18mm.
RJ11	Typical connector used in low speed communication equipment.
RS232	A low speed single ended communication bus with 2 parties.
RS485	A low speed differential communication bus with upto 32 parties.
Systemsoftware	Refers to the PEHA PHC Systemsoftware v3.x.y, this MS-Windows tool let's you configure your PHC project and define basic/functional programming.

2 Introduction

Phc2Mqtt is the combination of a hardware module (PCB + components) and operating software known as firmware. For simplicity we will furthermore refer to the Phc2Mqtt combination as the **module**.

Purpose of the module is to make a PHC domotic system accessible via MQTT, both for reporting PHC activity and for sending commands to the PHC system.

A PHC domotic system is made up from a steering module (STM) that is the master, and 1 or more PHC modules connected over an RS485 PHC module bus, communicating via a proprietary binary protocol. The STM also has a management interface, for STMv1/2 this is an RS232 serial link with a binary protocol, for STMv3 this is an ethernet interface with an XMLRPC protocol.

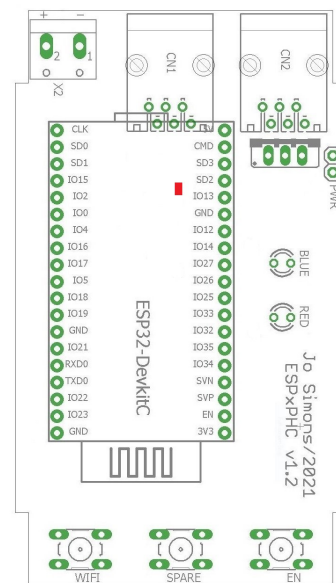
3 Hardware

The initial version of the module PCB was v1.2. Feedback from testers showed some issues in the design that needed fixing, and this resulted in a PCB v1.3. The different hardware versions will be explained here.

3.1 Version 1.2

Below is the layout of the module PCB, it fits in a 3MOD DIN-rail housing, relevant items are:

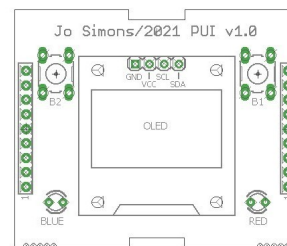
- CN1/CN2 are the PHC module bus connectors
- X2 is an optional 24V power connector, not used
- ESP32-DevkitC with a red power LED, this is the CPU
- Pushbutton at the bottom labeled 'WIFI'
- Pushbutton at the bottom labeled 'SPARE', not used
- Pushbutton at the bottom labeled 'EN'
- 2-pin header at top/right labeled 'PWR'
- 2 LED's on the right labeled 'BLUE' and 'RED'



Powering: Fieldtests have shown that the module cannot be powered reliably from the PHC power supply as it draws upto 140mA of current, resulting in unpredictable behaviour. So therefore it is recommended to connect the ESP32-DevkitC to a USB power supply (or a spare PC) after removing the cap from the PWR connector on the PCB.

To do this, open the module and take out the PCB, now remove the cap on the **PWR** connector on the right/top side of the module.

Warning: Failing to remove the cap can and will most likely damage the module, USB power supply or the used PC !!!



4 Operating modes

Depending on the operating mode of the module (Proxy or PassiveSTMv3), a different setup will be required.

4.1 Proxy mode

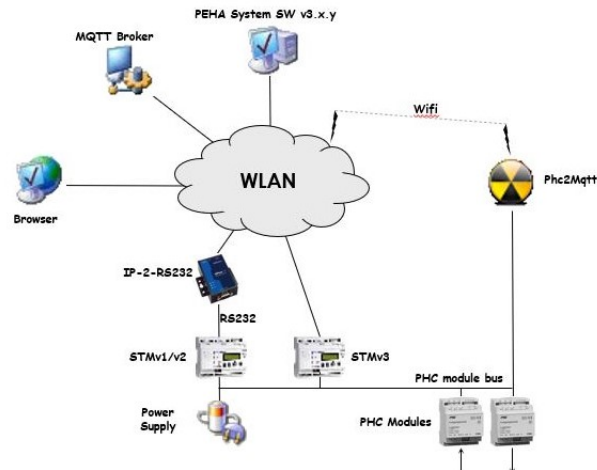
In this mode the module is chained into the PHC module bus using two 6-wire cables with RJ11 connectors, and it will passively monitor messages exchanged between PHC modules and STM.

Following drawing gives an overview of the end-to-end system architecture.

The intention is that the module connects to the wireless LAN (WLAN) such that the Systemsoftware and a web browser have access to it for configuration and reporting purposes.

It is via the same WLAN access that the module will connect to an MQTT Broker to provide it's services to a wider community.

Finally an optional IP-wise connection to the STM management interface is made through the WLAN to send commands to the PHC system. In case of the older STMv1/v2 you also need an IP-2-RS232 convertor like MOXA NPort 5110 and the sorts. In case of the STMv3 you can directly use it's network connection.



4.2 PassiveSTMv3 mode

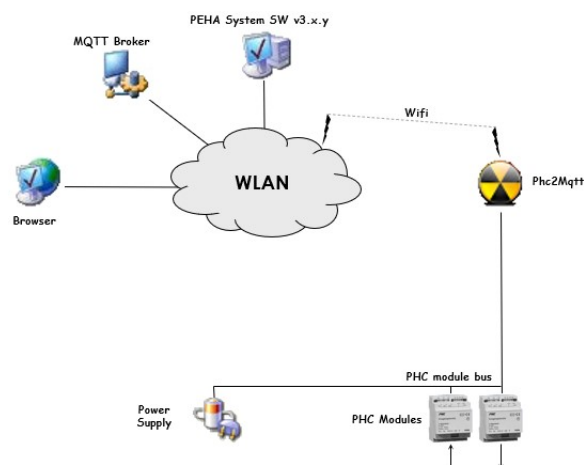
In this mode the real STM module (v1/2/3) is removed from the PHC system, and the module takes it's place. The module will be the master of the PHC module bus and talks directly to the PHC modules.

Same as in Proxy mode, the module connects to the wireless LAN (WLAN) such that the Systemsoftware and a web browser have access to it for configuration and reporting purposes.

It is via the same WLAN access that the module will connect to an MQTT Broker to provide it's services to a wider community.

In PassiveSTMv3 mode no additional hardware is needed to send commands to the PHC system as it is actively connected to the PHC module bus. The module will provide configuration information to the PHC modules when they (re)boot, it will acknowledge events sent by the PHC modules and report them (to logfile, MQTT), and finally it will translate incoming MQTT commands and send them to the PHC modules, all in a timely fashion respecting the bus protocol and timing.

The reason why we call this PassiveSTMv3 mode is because the module does not handle PHC module events, it just forwards them over MQTT. Opposed to a real STM that will lookup the event for the originating module and execute the commands linked to the event.



5 First usage of the module

When you use the module for the first time it will not be configured, you must first setup the Wifi connection to make your module reachable and then do basic configuration before connecting the module to your PHC system.

5.1 Linking the module to your Wifi network

- An unconfigured module cannot connect to your Wifi network, therefore it will startup in AP-mode whereby the red module LED will be on continuously.
- **Note:** In AP-mode only the webserver will run to allow configuration of the module.
- Use a Wifi enabled PC to connect to the module's AP by looking for a network called 'p2m_XXXXXX', the x's represent the lower 3 bytes of the module's MAC address.
- Once connected to the module's AP, use a web browser to access it on **192.168.4.1**, you will receive the main webpage as described in **6.1 Main Menu**.
- Goto **Main Menu**->**Configuration**->**Configure Wifi** and configure the Wifi settings as described in **6.3 Configuring Wifi**.
- After you configured the Wifi settings, reboot the module via **Main Menu**->**Reboot** (preferred), alternatively you can press the **EN** button (less preferred).
- The module starts up and lights both module LEDs, then connects to your Wifi network, after which both LEDs will turn off, the module is now in STA-mode.
- The module will try to obtain date/time from the Internet by means of SNTP or read it from the RTC module if one is installed.

5.2 Finding the module linked to your Wifi network

- Now the module is connected to your Wifi network, you need to find out it's IP address.
- If your module has the optional OLED display, then press button **B1** to display the module's IP address.
- If you don't have the OLED display, we suggest you log on to your Wifi router to consult the list of connected clients. Look for an entry showing 'Espressif', enter the shown IP address in a web browser, you should see the main webpage (ref **6.1 Main Menu**).
- As above IP address is obtained via DHCP from your Wifi network, it may be different each time the module reboots. It is preferred to assign a fixed IP address to the module in your Wifi router menu and save the settings.
- Now reboot the module, your Wifi router will assign it a fixed/DHCP IP address each time the module starts up.
- Use a web browser to access the module on the known IP address.

5.3 Basic configuration steps

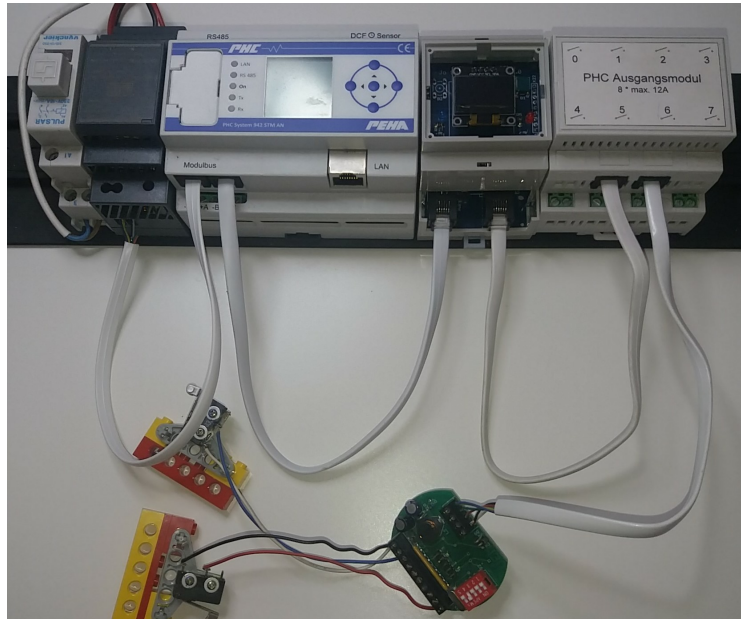
Before connecting the module to your PHC system, it is recommended that you perform below configuration steps:

- Goto **Main Menu**->**Configuration**->**Configure MQTT Client**, configure as described in section **6.2 Configuring MQTT Client**.
- Goto **Main Menu**->**Configuration**->**Configure Logging**, configure as described in section **6.3 Configuring Logging**.
- Goto **Main Menu**->**Configuration**->**Configure STM Daemon**, configure as described in section **6.6 Configuring STM Daemon**.
- Goto **Main Menu**->**Configuration**->**Configure STM Reporting**, configure as described in section **6.7 Configuring STM Reporting**.
- Goto **Main Menu**->**Configuration**->**Configure PHC External Client**, configure as described in section **6.8 Configuring PHC External Client**.

5.4 Connecting the module to your PHC system

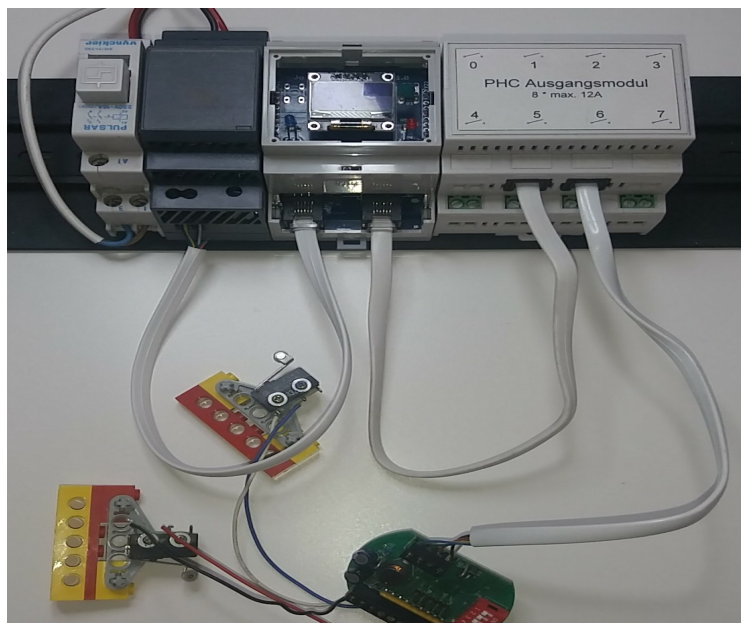
- Turn off your PHC system and make sure it is no longer powered.
- If you intend to run in **Proxy** mode, then chain the module in the PHC module bus by removing a 6-wire cable from a PHC module/STM and plugging it in **CN1/BUS1** on the module. Then plug a new 6-wire cable with 2 RJ11 connectors in **CN2/BUS2** of the module and back in the PHC module/STM to complete the chain.

Make some space and place the module onto the DIN rail such that it is mechanically stable.



- If you intend to run in **PassiveSTMv3** mode, then remove the real STM from your system and connect the module with the existing 6-wire cables.

Place the module onto the free space on the DIN rail such that it is mechanically stable.



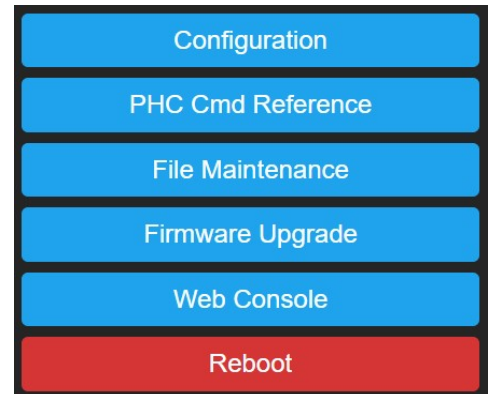
- Power on the module and your PHC system.
- You will see the red power LED on the ESP32-DevkitC turn on.
- The module will start up and connect to your Wifi network.

6 Web interface

Configuration and management of the module can be done via a web browser by entering the known IP address of the module.

6.1 Main Menu

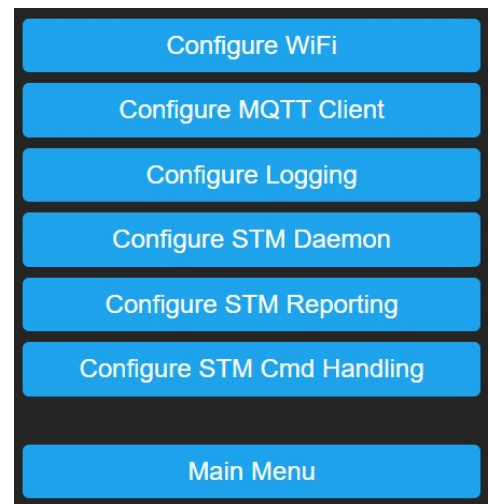
The homepage of the module is the main menu as shown on the right. Click on a button to enter the related page.



6.2 Configuring the module

Main Menu->Configuration

Click on a button to enter the related configuration page.



6.3 Configuring Wifi

Main Menu->Configuration->Configure Wifi

- **Remote AP SSID** (network name) and **Remote AP Password**: The module will use these credentials to connect in STA-mode to the Wifi network. When you leave these fields empty, the module will boot into AP-mode.

- **Web Admin Password**: You can specify a password to protect web access to the module, in this case you need to logon to the module when prompted by the browser, the username is 'admin' completed with the specified password.

- **Save**: Press this button to save settings, then reboot the module to let changes take effect (**Main Menu->Reboot**).

Lockout: If you entered a wrong Remote AP SSID/Password, connecting to the Wifi network will fail. The module will retry connecting 3 times after which it will alternately flash blue/red module LEDs for 3 seconds to indicate the failure, next the module will reboot and connecting starts again... until connection succeeds. This can cause a lockout where you cannot access the module.

The lockout can also happen if you forgot the Web Admin Password or if you replaced your Wifi network with another.

Recovery: You can recover from this lockout by pressing and holding the **WIFI** button, then press/release the **EN** button. The module will reboot, flashing blue/red module LEDs simultaneous for 3 seconds to confirm the pressed **WIFI** button, you can release the **WIFI** button by now.

The module will boot in AP-mode and turn the red module LED on to indicate this.

You can now connect to your module's AP and access it on 192.168.4.1 with a web browser without password. Change the Wifi settings to the correct values, save them and reboot the module to revert to normal STA-mode operation.



6.4 Configuring MQTT Client

Main Menu->Configuration->Configure MQTT Client

The MQTT client is the interface to the MQTT broker (server), when enabled it will connect to the broker and provide a send/receive path to the module for receiving commands and reporting information.

The MQTT client itself will not subscribe to any topic, the different sub-systems in the module will do this to suit their needs. Refer to **6.7 Configuring STM Reporting** and **6.8 Configuring STM Cmd Handling** for details.

When the MQTT client receives published messages, it will offer them to the different sub-systems in the module, it is upto them to determine whether to use the message or not.

- **Broker IP Address** and **Broker IP Port**: Enter the MQTT Broker IP address and port (default 1883), these fields are mandatory.

- **ClientId**: A unique name across all clients that connect to the broker, maximum 24 characters, mandatory.

If this is not unique you will see that the MQTT client loses its connection from time to time.

- **User** and **Password**: Optional, if needed to connect to the broker.

- **RX Topic Prefix**: Max 24 characters. The MQTT client will subscribe to '<rx-topic-prefix>/#' to receive messages with commands for the PHC modules.

We propose to use '<alias>/m2p', where you replace <alias> with a name for your house/location. Notice that 'm2p' is chosen intentionally to indicate direction, from MQTT to PHC.

- **TX Topic Prefix**: Max 24 characters. This prefix is used when publishing messages to the MQTT broker.

We propose to use '<alias>/p2m', where you replace <alias> with a name for your house/location. Notice that 'p2m' is chosen intentionally to indicate direction, from PHC to MQTT.

By using asymmetric <rx-topic-prefix> and <tx-topic-prefix> we can avoid receiving the own published messages to <tx-topic-prefix>. But you can choose whatever you want.

- **Enable**: Allows you to enable/disable the MQTT client.

- **Save**: Press this button to save settings, then reboot the module to let changes take effect (**Main Menu->Reboot**).

6.5 Configuring Logging

Main Menu->Configuration->Configure Logging

The module has extensive logging features that enable you to monitor module behaviour, but also to give you support in case of problems.

Some of them generate serious overhead so they must remain inactive as much as possible and should only be set when asked by support team.

To provide the usable and necessary feedback to the user we suggest below settings:

- **Serial Loglevel**: Set to '0 None' for normal operation, serial logging is sent over the USB connection of the ESP32 DevkitC.

- **Web Loglevel**: Should be set to '2 Info' for normal operation, this logging can be observed in **Main Menu->Web Console**.

- **Syslog Loglevel**: Set to '0 None' for normal operation.

- **Syslog IP Address**: In case syslog logging is needed, fill out the IP address of the target syslog daemon.

- **Syslog IP Port**: In case syslog logging is needed, fill out the listen port of the target syslog daemon (default 514)

- **Save**: Press this button to save settings, **Serial Loglevel** and **Web Loglevel** changes will have immediate effect, the other settings require a module reboot to let changes take effect (**Main Menu->Reboot**).

6.6 Configuring STM Daemon

Main Menu->Configuration->Configure STM Daemon

The STM Daemon (STMD) is the heart of the module which coordinates activities to/from the PHC system, it also provides an XMLRPC management interface to transfer project data to the module using the Systemsoftware.

This project data is stored in 2 files:

- project.prg: This file contains the functional/basic programming of your project in compiled binary form.
- project.zip: This file contains the metadata of your project in zipped format, it is an exact copy of project.zpfx as created by the Systemsoftware.

This ZIP file contains a number of files, but the module is only interested in project.ppfpx, which contains an XML formatted list of PHC modules present in your PHC system. STMD parses that XML file to build a module-list which is needed for correct operation of the **STM Reporting** and **STM Cmd Handling** sub-systems.

To transfer the project data, open your project.zpfx file in the Systemsoftware, goto the properties of the STM in your project and enter the module's IP address, press TAB and you should notice that version 3.30 is reported.

Goto the transfer window of the Systemsoftware and press **Start**, the project data will be transferred to the module, you can observe this in the **Web Console**.

- Operating Mode:

'Proxy': The STMD analyzes packets on the PHC module bus and reports events/status via the MQTT client, it also relays commands coming in from MQTT client to the management interface of the real (remote) STM in your PHC system.

'PassiveSTMv3': The STMD replaces the real STM of your PHC system, it is the master of the PHC module bus and communicates directly with the PHC modules.

- **Management STM Address:** This should match the STM address as defined in your project in the Systemsoftware, by default this is 0.

- **Management Interface:** Enable/disable the management interface of the module.

When enabled the XMLRPC server will be activated on port 6680 by which the Systemsoftware can transfer it's project data to the module.

- Module-List:

'Auto extract from transferred project': The STMD will automatically extract project.ppfpx after transferring the project data with the Systemsoftware, and will rebuild the module-list by parsing project.ppfpx.

'Manual upload project.ppfpx': You will need to upload project.ppfpx yourself (see below), this is intended as backup in case the first option fails.

The window below lists all modules as known in your project, the left column is the logical module type that will be used in reporting/commands, the right column is the physical module type which is more detailed.

Both **STM Reporting** and **STM Cmd Handling** sub-systems use this module-list to correctly convert PHC module bus packets to/from readable format.

When STMD sees activity from PHC modules not in the module-list, it will add them with the default module type depending on the PHC module address.

```
0x00-0x1F : imd
0x20-0x3F : uim
0x40-0x5F : omd
0x60-0x7F : amd
0x80-0x9F : ebs
0xA0-0xBF : dim
```

- **Save:** Press this button to save settings, then reboot the module to let changes take effect (**Main Menu->Reboot**).

- **Upload project.ppfpx:** Here you can manually upload project.ppfpx. You can obtain this file by opening your project.zpfx file with WinZip/7Zip/... and extracting the file project.ppfpx. If you have multiple STM's in your PHC system, then split project.ppfpx into 1 file per STM and reduce the contents before uploading as the module has approximately 600Kb available to store it.

Press **Choose file** to select your project.ppfpx file, then press **Start Upload**, the module will now upload the file and parse it to build the module-list.

6.7 Configuring STM Reporting

Main Menu->Configuration->Configure STM Reporting

The reporting sub-system of the module is the one looking at all the packets that are sent on the PHC module bus it is attached to, and which reports them in readable format based on the module-list.

- **Format:** This selection will determine the reporting format.

-> **xPhcLogd compatible:** This format equals the reporting done by xPhcLogd tool, this means that each item is reported separately, output looks like this:

```
<tx-topic-prefix>/boo/<mod>.<addr>      = [attrib:<value>]
<tx-topic-prefix>/evt/<mod>.<addr>.<chan> = <event>
<tx-topic-prefix>/sta/<mod>.<addr>.<chan> = <state>
<tx-topic-prefix>/cmd/<mod>.<addr>.<chan> = <cmd> *[,<parm>]
```

```
e.g. alias/p2m/evt/imd.0.in0 = ingt0
e.g. alias/p2m/sta/omd.0.out3 = 1
```

-> **Text format:** This format tries to group reportable items per PHC module, it's output is plain text and looks like this:

```
<tx-topic-prefix>/boo/<mod>.<addr> = [attrib:<value>]
<tx-topic-prefix>/evt/<mod>.<addr> = <chan>:<event> *[,<chan>:<event>]
<tx-topic-prefix>/sta/<mod>.<addr> = <chan>:<state> *[,<chan>:<state>]
<tx-topic-prefix>/cmd/<mod>.<addr> = <chan>:<cmd> *[,<parm>]
```

```
e.g. alias/p2m/evt/imd.0 = in0:ingt0
e.g. alias/p2m/sta/omd.0 = out0:1,out1:0,out2:0,out3:1,out4:0,out5:0,out6:0,out7:0
```

-> **JSON format:** This format tries to group reportable items per PHC module; it's output is JSON formatted and looks like this:

```
<tx-topic-prefix>/boo/<mod>.<addr> = { [attrib:<value>] }
<tx-topic-prefix>/evt/<mod>.<addr> = { <chan>:<event> *[,<chan>:<event>] }
<tx-topic-prefix>/sta/<mod>.<addr> = { <chan>:<state> *[,<chan>:<state>] }
<tx-topic-prefix>/cmd/<mod>.<addr> = { <chan>:<cmd> *[,<parm>] }

e.g. alias/p2m/evt/imd.0 = {"in0":"ingt0"}
e.g. alias/p2m/sta/omd.0 = {"out0":1,"out1":0,"out2":0,"out3":1,"out4":0,"out5":0,"out6":0,"out7":0}
```

To see the differences you can later on observe them in the **Web Console**

- **Report To File:** Here you can enable/disable reporting of PHC module packets to a persistent circular log on a per type basis (boo=boot,evt=event,sta=state,cmd=command). The circular log can store upto 96Kb.

This feature is handy to investigate sporadic problems with your PHC system.

For practical reasons it takes 15 seconds for new log data to become permanent and viewable.

- **Report To MQTT:** Here you can enable/disable publishing of PHC module packets via the MQTT client on a per type basis (boo=boot,evt=event,sta=state,cmd=command).

The MQTT client will add the <tx-topic-prefix> in front.

- **Save:** Press this button to save settings after which they take effect.

6.8 Configuring PHC External Client

Main Menu->Configuration->Configure PHC External Client

The cmd-handling subsystem of the module handles PHC commands that come in via MQTT (or the Web Console) depending on the operating mode:

Proxy mode: The commands will be forwarded to the remote STM in your PHC system using the PHC External Client.

If the remote STM is an STMv3, then the external client uses XMLRPC over HTTP and there is no additional hardware needed.

If the remote STM is an STMv1/v2 then the external client uses a binary protocol over TCP/IP and you need an IP/RS232 convertor module to hook the RS232 management port of the STMv1/v2 up to your WLAN network.

PassiveSTMv3 mode: The commands will be sent directly to the PHC module bus without the need for extra hardware.

- **Remote STM Mode:** Select the appropriate remote STM in your PHC system.
- **Remote STM IP Address** and **Port:** IP address and port on which the remote STM in your PHC system is reachable.

For an STMv3 this is the STM's IP address and port will be 6680.

For an STMv1/v2 this will be the IP address of the IP-2-RS232 convertor, the port depends on the settings of the convertor which must be in TCP server mode.

- **Remote STM Mgmt Address:** This is the remote STM management address as defined in the Systemsoftware for the STM component, normally this is 0.

- **Enable Remote STM:** Enable/disable the PHC external client.

- **Save:** Press this button to save settings, then reboot the module to let changes take effect (**Main Menu->Reboot**).

Configure PHC External Client

Remote STM Mode
STMv3

Remote STM IP Address ()
192.168.0.93

Remote STM IP Port (6680)
6680

Remote STM Mgmt Address
0

Enable Remote STM
Disabled

Save

Configuration

6.9 PHC Cmd Reference

Main Menu->PHC Cmd Reference

By clicking on a hyperlink in this page you will get a list of commands applicable to the clicked item in the text area below.

Note: Commands related to input (in%/ir%/im%/il%) and feedback (fb%) channels are not available in PassiveSTMv3 mode as the module does not handle them.

These commands can be executed by publishing an MQTT message to topic '/cmd/ccmd' where payload contains the commands. These commands can also be executed by entering them in the Web Console command line.

PHC Ccmd Reference

Below commands can be published to topic: avdln2/m2p/cmd/ccmd
 payload = <cmd> *[, <cmd>]
 cmd = <root>|<stm>|<imd>|<imw>|<tab>|<et0>|<uim>|<utm>|<bwm>|<et1>|<mcc>|<omd>|<omx>|<jrm>|<amd>|<fui>|<fu1>|<fu2>|<ebs>|<ebr>|<ebd>|<mls>|<dim>|<dal>

Example
 payload = jrm.0.out0.delayeddown.0.0.5.150.20;omd.0.out0.toggle

```

jrm.<0-31>
jrm.<0-31>.status
jrm.<0-31>.boot
jrm.<0-31>.timeEnded
jrm.<0-31>.out<0-3>
jrm.<0-31>.out<0-3>.status
jrm.<0-31>.out<0-3>.stop.<prioLevel:0-5>
jrm.<0-31>.out<0-3>.toggleUp.<prioLevel:0-5>.<prioLock:0-1>.<run:1-65535 1/10s>
jrm.<0-31>.out<0-3>.toggleDown.<prioLevel:0-5>.<prioLock:0-1>.<run:1-65535 1/10s>
jrm.<0-31>.out<0-3>.up.<prioLevel:0-5>.<prioLock:0-1>.<run:1-65535 1/10s>
jrm.<0-31>.out<0-3>.down.<prioLevel:0-5>.<prioLock:0-1>.<run:1-65535 1/10s>
jrm.<0-31>.out<0-3>.tipUp.<prioLevel:0-5>.<prioLock:0-1>.<run:1-65535 1/10s>
jrm.<0-31>.out<0-3>.tipDown.<prioLevel:0-5>.<prioLock:0-1>.<run:1-65535 1/10s>
jrm.<0-31>.out<0-3>.priolock.<prioMask:0-63>
jrm.<0-31>.out<0-3>.priounlock.<prioMask:0-63>
jrm.<0-31>.out<0-3>.prioreset.<prioMask:0-63>

```

Main Menu

6.10 File Maintenance

Main Menu->File Maintenance

In this menu you can download or clear/delete files:

- **Download File:** Select the desired file in the dropdown list, then click **Download File** to start download to your PC.

- **Clear File:** Select the desired file in the dropdown list, then click **Clear File**. You will be prompted to confirm the clear action.

File Maintenance

Download File

Logfiles

Download File

Clear File

Logfiles

Clear File

Main Menu

6.11 Firmware Upgrade

Main Menu->Firmware Upgrade

The module supports OTA (over-the-air) firmware upgrades, this means that you can upload a new firmware file via a webpage, after which the module will execute the new firmware when it boots.

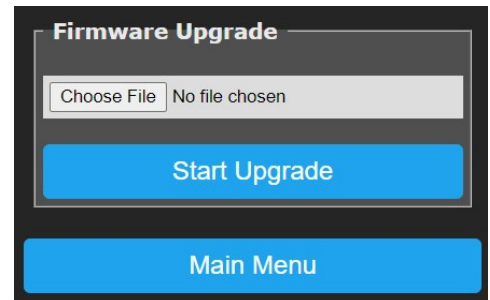
From **Main Menu** select **Firmware Upgrade** and follow below instructions.

- **Choose File:** Press this button to select a new firmware file that is stored on your PC.

- **Start Upgrade:** Press this button to start the upgrade, you will be prompted to confirm or cancel this operation.

When confirmed the firmware file will be uploaded while showing a rotating caret/progress marker.

After completion of the upload, you need to reboot the module to let changes take effect (**Main Menu**->**Reboot**).

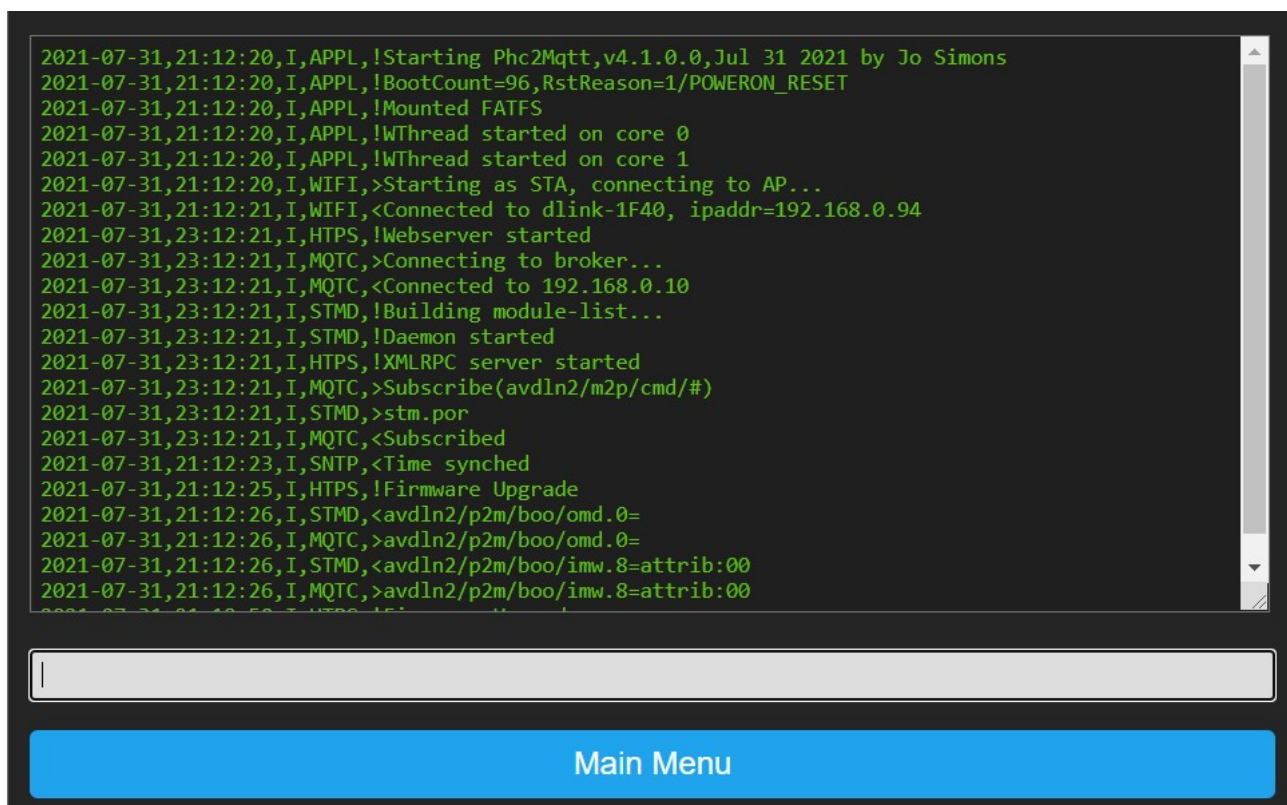


6.12 Web Console

Main Menu->Web Console

The web console is the place where logging is shown and you can observe what the module is doing in the big text window.

The smaller window near the bottom is the command-line, you can enter commands here. Enter '?' and press <enter> to get a list of commands, you can also enter the commands as listed in **6.9 PHC Cmd Reference**.



6.13 Reboot

Main Menu->Reboot

Clicking the **Reboot** button will prompt you for confirmation, confirm or cancel at your will.

When confirmed the module will stop the different sub-systems in an orderly fashion and restart itself.

After a few seconds the webpage will be redirected to the module's homepage.