Cross-Platform Web Remote Center

(xWRC)

Reference Manual

1.Prolog

1.1.Version history

Version	3.1.0.0, 01p01, 09-may-2013
Comments	Initial version

Version	3.3.0.0, 01p01, 15-apr-2014
Comments	Added support for STMv3 iServer XML-RPC

Version	3.3.0.6, 01p01, 24-dec-2014
Comments	Adjusting xml/terse/verbose directives & result syntax, amd/ebd/dim responses. Remove tags: server plugcount & plugins. Add tags: server httpaddr. Adjust jrm/ebr/dim/ebd/fu2 commands and handling of parameters >255.

1.2.Referenced documents

[1]	RFC2616	Hypertext Transfer Protocol 1.1
[2]	RFC2617	HTTP Basic/Digest Access Authentication

1.3.Abbreviations

AJAX	Asynchronous Javascript And Xml
ARM	Acorn RISC Machine processor architecture
CSS	Cascading Style Sheets
DMZ	De-Militarized Zone, traffic forwarding from WAN to LAN host
DNS	Domain Name System, translates logical name into IP@
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HTTPS	Secure HTTP, uses encryption for HTTP communication
IP	Internet Protocol
IP@	IP Address
LAN	Local Area Network, your private in-house network
MIPS	Microprocessor without Interlocked Pipeline Stages processor architecture
MOD	PHC MODule
NAT	Network Address Translation, conversion between WAN/LAN
PC	Personal Computer
PHC	PEHA © Home Control
PPF	PHC Project File
STM	PHC Steering Module
TCP/IP	Transmission Control Protocol over IP
UMDB	User Management DataBase
URL	Uniform Resource Locator, link to a webpage
WAN	Wide Area Network, the public Internet
X86	Intel processor architecture
XML	eXtensible Markup Language
XML-RPC	XML Remote Procedure Call

1.4.Getting support

If you encounter problems with xWRC/xiControl then send an email to 'icontrol@telenet.be', please provide below details, in return we will support you as soon as possible:

- Your configuration:
  - how many modules of each, STM version, xWRC version, ... ‘
  - IP addresses and ports used
- A good description of the problem
- Optional the HTML template files that you are using

1.5.Notational conventions

For ease of reference we will use below notational conventions:

```
'string'      : literal string as it shall appear
[ ]           : everything between the brackets is optional
*             : the following definition can occur zero or more times
#value        : numerical value, either decimal (123) or hexadecimal (0x1B3C)
$value        : string value, i.e. abc
( )           : groups multiple options
|             : denotes 'or' in case of multiple options (# | $ | *)
//            : the rest of the line is comment
```

1.6.Error codes

Error	Description
0	Success
1	Internal error occurred
2	Out of memory
3	Creation error
4	Open error
5	Write error
6	Read error
7	Invalid command
8	Invalid parameter
9	No resources to complete operation
10	Resource not found
11	No license
12	Access denied due to insufficient rights
13	Function or system disabled
14	System busy
15	Request pending
16	Request malformed
17	Response malformed
18	Operation timed out
19	Request aborted
20	Not connected
21	No data available

2.xWRC

2.1.Intro

xWRC (cross-platform Web Remote Center) is a webserver that can be accessed via a regular webbrowser, we will furthermore call xWRC the **server** and a webbrowser the **client**.

Cross-platform means that the same source code is compiled for different platforms into different executable files, but with the same functionality. We will refer to any executable version as 'xwrc'.

Currently we support:

```
    Windows 32bit on x86 CPU’s       xwrc.exe
    Linux on x86 CPU's               xwrc.x86
    Linux on ARM CPU's               xwrc.arm
    Linux on MIPS CPU's              xwrc.mips
```

The server's basic functionality is to serve a collection of webpages that make up a website. The layout of the website and the page content is fully customizable, all you need is basic knowledge of HTML or an HTML editor (i.e.Dreamweaver, …) and an idea how your site should look like. As an alternative the xWizard tool can be used.

The server supports a structured command URL to extend it's functionality beyond serving files. Part of the supported commands will be handled by xWRC itself, and some will be passed along to to xiControl which is an extension to the server.

xiControl is an integrated part of software that handles communication with a PEHA (c) PHC domotic system, allowing to control it from a webpage and report it's status. For more details refer to the xiControl help section.

Files that are served by xWRC can contain tags, these are placeholders for variable data that are replaced on-the-fly when the file is sent back to the client.

2.1.1.Package contents

Following directories and files are included in an xWRC package:

```
  /xwrc/bin/xwrc.exe                     // Windows 32bit executable
  /xwrc/bin/xwrc.arm                     // LINUX ARM executable
  /xwrc/bin/xwrc.mips                    // LINUX MIPS executable
  /xwrc/bin/xwrc.x86                     // LINUX x86 executable
  /xwrc/bin/xwrc.ini                     // default ini-file
  /xwrc/bin/xwrc.umdb.ini                // default umdb configuration
  /xwrc/bin/xwrc.license.bin             // optional license key file
  /xwrc/bin/xwrc.help.vX.Y.Z.T.pdf       // global help file for xWRC/xiControl
  /xwrc/default.html                     // server starting page
  /xwrc/xwrc.*.html                      // xWRC files
  /xwrc/img/*.gif                        // xWRC images
  /xwrc/log/placeholder                  // log file directory
  /xwrc/icontrol/*.html                  // xiControl files
  /xwrc/icontrol/img/*.gif               // xiControl images
```

2.1.2.Package installation

To install a new xWRC package just unzip the archive and maintain the included directory structure. On a LINUX system you use: tar –xvf xwrc.tar.

2.2.Configuring

2.2.1.Using command-line options

When xWRC starts up, it uses a default set of configuration options. Options entered in the command line are handled from left to right and overwrite the previous value of the option.

The used syntax is as follows:

```
xwrc *[ - <option> ]
```

To get a list of the available options, start xWRC as follows:

```
./xwrc -help

e.g. ./xwrc.mips -help
```

2.2.2.Using an ini-file

In order to allow predefined configurations and avoid typing errors, it is possible to specify all options in an ini-file and let xWRC use that.

```
xwrc -inifile <ini-file>

e.g. ./xwrc.mips -inifile ./xwrc.ini
```

Note that an option in the ini-file overrides any previous occurrence of that option, and similar any option specified on the command line after the ini-file will override the previous value of an option.

```
e.g. ./xwrc.mips -inifile ./xwrc.ini -httpport 8080
```

Above sample will use 8080 as the HTTP port to listen on, even if another value was present in the ini-file.

The layout of the ini-file follows the standard Windows ini-file format, it has following syntax:

```
ini-file     = empty-line | comment-line | section-line | item-line

empty-line   = ''

comment-line = ';' comment

section-line = '[' section-name ']'

item-line    = key '=' value
```

2.2.3.xWRC settings

xWRC configuration is specified in the [xwrc] section and supports following items:

```
[xwrc]
; license file to use, default is './xwrc.license.bin'
;licfile=./xwrc.license.bin

; HTTP listening address, default is 0.0.0.0
;httpaddr=127.0.0.1

; HTTP listening port, default is 80
;httpport=8080

; xWRC hostname override
;hostname=myservername

; web page root directory relative to xwrc/bin, default is '..'
;httproot=..

; server theme, default is 'default'
; default page to return when accessing the server root or directory is derived as '<theme>.html'
;theme=default

; fontsize to use for the the fontsize tag, default is 8
;fontsize=8

; user management db file, default is './xwrc.umdb.ini'
;umdbfile=./xwrc.umdb.ini

; authentication method, default is 0 (none)
;   0=none    : no authentication, everybody can access all content
;   1=basic   : basic authentication, user/ugrp/rule logic will be applied to restrict access
;   2=session : session based authentication, allows login/logout, user/ugrp/rule logic will
;               be applied to restrict access
;authmeth=0

; allow authentication by means of peer ip-address, default is 0 (no)
;   0=no      : peer ip-address is not used
;   1=yes     : peer ip-address is checked against user db to grant unauthenticated access
;authaddr=0

; loglevel, decimal or hex 32bit value, default is 0
; only set to non-zero value as per instruction of support engineer
;loglevel=0xFFFFFFFF

; logfile prefix, default is '../log/xWRC'
; will generate logging to ../log/xWRC_YYYYMMDD_hhmmss.log
;logpfx=../log/xWRC

; syslogd settings to report logging to remote syslog deamon, default is disabled
;logaddr=192.168.0.127
;logport=514
```

2.3.Access, restrictions & security

xWRC provides the possibility to limit access to files and objects in a connected PEHA (c) PHC domotic system, but by default this is all turned off, thus granting full access to everybody.

The basic access restriction model used by xWRC is this:

```
     +------+ 1   1-16 +------------+ 1      N +------+
     | USER |--------->| USER-GROUP |--------->| RULE |
     +------+          +------------+          +------+
```

This means that each user can belong to 1 upto 16 user-groups, and each user-group can have an undefined number of rules linked to it.

All the information of users, user-groups and rules are placed in a separate ini-file, by default this is 'xwrc.umdb.ini', this ini-file has sections for each part of information as follows:

2.3.1.User-groups

```
;
; List of user-groups
;
; When ugrp-name or description contains a space, it has to be placed between double quotes
;
; Syntax: ugrp-name '=' flags ',' def-access ',' {description}
;
; ugrp-name     = mandatory, maximum 16 characters, starting with a letter
; flags         = bitwise OR of following values, default is 0
;                 1    group is enabled
; def-access    = default access to apply when no matching rule is found, default is 0
;                 0    none
;                 1    read
;                 2    exec
;                 3    full
; description   = optional, maximum 64 characters
;
[ugrps]
grpAdmin=1,3,"Administrator user-group"
grpGuest=1,1,"Guest user-group"
```

2.3.2.Users

```
;
; List of users
;
; When <user-name>, <pswd>, <description> or <ugrp-name> contains a space, it has to be
placed between double quotes
;
; Syntax: user-name '=' flags ',' pswd ',' {description}  1-16(',' ugrp-name)
;
; user-name     = mandatory, maximum 16 characters, starting with a letter
; flags         = bitwise OR of following values, default is 0
;                 1: user is enabled
;                 2: user is allowed to change password
;                 4: user is admininstrator
;                 8: user is IP-address
; pswd          = plain text or hashed password, maximum 15 characters
; description   = optional, maximum 64 characters
; ugrp-name     = 1 upto 16 user-groups the user belongs to
;
[users]
admin=5,"admin","Administrator",grpAdmin
guest=3,"guest","Guest",grpGuest
```

2.3.3.xWRC access rules

```
;
; This section lists all rules applicable to webpages served
;
; Wildcards that apply: '?' can occur multiple times at any place
;                       '*' can occur once at any place
;
; Syntax: url '=' access 1-*(',' ugrp-name)
;
; url           = '"' *(ascii | '*' | '?') '"'
;
; access        = 0    none
;                 1    read
;                 2    exec
;                 3    full
;
; ugrp-name     = 1 upto 16 user-groups the user belongs to
;
[xwrc]
"*"=1,grpAdmin
"/icontrol/util*"=0,grpGuest
"*"=1,grpGuest
```

2.3.4.Authentication options

By default, xWRC grants full access to all files/PHC elements, to everybody.

There are 2 xWRC options that will limit this, and apply user authentication and access rules.

```
; authmeth specifies the authentication method to use, values supported are:
; 0 = no authentication (default)
; 1 = basic authtentication:
;     - this will let the browser popup a window to prompt for a user/password combination,
;     - this method does not allow you to logout that user, unless your browser supports ;
;       clearing 'authenticated sessions'
; 2 = session authentication:
;     - this will let a user login using a webpage, and also allows the user to logout
authmeth=0

; authaddr specifies whether a browser can login without any password, by using the
; IP-address of the machine it runs on,  note that this is considered insecure and can
; only be applied for local networks
; 0 = do not support IP-address based login (default)
; 1 = support IP-address based login
authaddr=0
```

2.3.5.Access from Internet

Please note that when you are using a router to connect your home network to the Internet, it will not be straight forward possible to access the xWRC server from the Internet.

This because the router applies a private IP addressing scheme on it's home network side (=LAN), and a public IP address on it's Internet side (=WAN). This feature is called NAT (Network Address Translation). So any PC behind the router cannot directly be reached from the Internet.

In order to get access from the Internet you will need to take following steps:

1. Either get to know the public IP address of your router, or register a domain name at a DNS (Domain Name Server) provider like http://www.DynDns.org (this is free). This has the effect that a logical name like 'myserver.dyndns.org' is translated into the public IP address of your router.
2. Setup a port forwarding in your router, this feature is sometimes called DMZ (De-Militarized Zone). The port forwarding looks for incoming traffic at a specific port on the Internet side (i.e. 80), and forwards it to a specific IP address/port combination on the home network side (i.e. 192.168.0.99, port 80).

Example: you try to connect with a browser to http://myserver.dyndns.org/ , an Internet DNS server will redirect your browser to the public IP address of your router. Your router sees the request arrive at port 80 (http default) and forwards it to your xWRC server with IP address 192.168.0.99 on port 80, where xWRC handles the request.

Please note that certain ISP's (like Telenet in Belgium) are blocking incoming traffic on port 80, just to prohibit you from deploying a private webserver on the Internet. In this case you can use another port (i.e. 8000, 8080) to reach your router, and setup a modified forwarding of port 8000 to your xWRC server.

In this case you should enter following in your webbrowser to reach the xWRC server:
http://myserver.dyndns.org:8000

2.3.6.Securing access

Making your PHC system accessible via a webbrowser also implies that unauthorized people may gain access, abuse your system or cause dangerous situations (i.e. opening doors, lock out, ...).

To prevent this you should at least enable xWRC server authentication by putting the 'Authenticate Users' option to 'yes' in the server admin page, this way every user accessing the xWRC server needs to have a user-id and password before access is granted.

This is however a very limited protection because the user-id and password sent over the network can be traced and decoded very easy. This can be considered sufficient for local network (LAN) access only.

When accessing xiControl from the Internet you should make sure that user-id and password are encrypted, this can be achieved by using the Secure HTTP (HTTPS) protocol. HTTPS is the same as HTTP but the data that is sent over the network is encrypted using a certificate, HTTPS uses port 443 as oposed to port 80 for HTTP.

Unfortunately xWRC server does not support HTTPS, but there is an off-the-shelve solution called 'stunnel'. Stunnel is an application that accepts incoming HTTPS connections, and converts them into an outgoing HTTP connection towards xWRC server. Just install Stunnel on the same PC as xWRC.

You should also modify the port forwarding of your router as described in section 4) above, replace port 80 by port 443 and you are ready to go. Now access your xWRC server as https://myserver.dyndns.org/

Note that some ISP's (like Telenet in Belgium) are blocking incoming traffic on port 443, to avoid that you are running a secure web server at home. We just trick them by using port 8443 iso 443, now access your xWRC server from Internet as https://myserver.dyndns.org:8443/

If using port 8443 is too difficult for you or a firewall prevents you from using this port, then you can also install a 'WebHop' at DynDns, this is also free. What a WebHop does is convert the access to an alias into an access with a specific port or another host, i.e. https://myserver.home.dyndns.org/ is converted into https://myserver.dyndns.org:8443/.

For more details on WebHop visit http://www.dyndns.com.

2.4.Structured command URL

xWRC receives requests via a structured command URL containing one or more commands and output directives.

2.4.1.Command URL syntax

```
url               = 'http://' [user [':' pswd] '@'] server [':' port] '/' [req-file-spec] ['?' query]

user              = username in case basic authentication is enabled

pswd              = password in case basic authentication is enabled

server            = name or IP address of xWRC server, i.e. 'myserver', '192.168.0.99'

port              = optional port on which xWRC server is listening, default is 80

req-file-spec     = ( *[file-path '/'] [file-name] | 'icontrol.dll' )

file-path         = optional path containing file to be served, i.e. /dir1/dir2/dir3/

file-name         = optional filename including extension, i.e. abc.html,
                    if omitted then 'default.html' will be used

query             = [ command ] *['&' command ]

command           = wait-directive | output-directive | compound-cmd

wait-directive    = 'wait=' #delay

#delay            = 0...60000 (milliseconds)

output-directive  = verbose-directive | terse-directive | xml-directive |
                    file-directive    | ipfile-directive

verbose-directive = 'verbose'

terse-directive   = 'terse'

xml-directive     = 'xml'

file-directive    = 'file=' file-spec

ipfile-directive  = 'ipfile=' file-spec

file-spec         = ['/'] *[file-path '/'] [file-name]

compound-cmd      = 'ccmd=' ...
                    passed on to xiControl, refer xiControl help files for info
```

Example:

```
  http://myserver/default.html
  http://myserver/?file=/abc.html
```

xWRC will parse the command URL and execute any commands present, during which a $verbose-result, $terse-result and $xml-result will be composed with the outcome of each command.

If req-file-spec equals '/icontrol.dll' then the default file to return will be '/icontrol/default.html', if req-file-spec is missing then '/default.html' will be returned.

After execution of all commands, xWRC will send a response based on the req-file-spec overridden with optional output-directives.

2.4.2.Verbose result

The $verbose-result is a more textual description of the command outcome, with below syntax:

```
verbose-result = cmd-line CRLF *[result-line CRLF]

CRLF           = carriage-return + line-feed
cmd-line       = marker cmd ':' (error-result | success-result)
result-line    = marker $result-data

marker         = last-line | detail-line
last-line      = '-'
detail-line    = '+'

cmd            = command that was executed (without parameters)
error-result   = "err" #error-code $error-string
success-result = "ok" [$result-data]
```

Example:

```
- stm.inquiry: ok 2.18,05/07/10,12:26:46
- omd.0: err 5 Invalid parameter
- omd.0: ok 8
+ stm.read: ok
+ 4000 : 0B60EEC3-40404B60-FFFFFFFF-FFFF4380 .`..@@K`......C.
- 4010 : 44804580-FFFFFF                     D.E....
```

2.4.3.Terse result

The $terse-result is a string concatinating either the error-code of a failing command, or the decimal formatted result-data of each command. A command that returns a string will add "0" to the terse result.

The separating character is the same as the character separating the commands, either a semicolon (;) or a colon (:) is allowed. The use of these characters and their meaning is determined by the command sending application.

Syntax:

```
terse-result   = (error-result | success-result) *[ (';' | ':') (error-result | success-result) ]

error-result   = '-' #err-code

success-result = #result-data
```

Example:

```
0;1;0;-5
```

2.4.4.XML result

The $xml-result is an XML formatted string with an encapsulating <main> tag and one or more <obj> tags representing the outcome of each command.

Syntax:

```
xml-result     = xml-header main-tag

xml-header     = '<?xml version="1.0" encoding="ISO-8859-1"?>'

main-tag       = '<main>' *[obj-tag] '</main>'

obj-tag        = '<obj>' id-tag err-tag value-tag '</obj>'

id-tag         = '<id>' cmd without params '</id>'

err-tag        = '<err>' #error-code '</err>'

value-tag      = '<value>' $result-data '</value>'
```

Example:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<main>
 <obj>
  <id>omd.0.out0</id>
  <err>0</err>
  <value>0</value>
  </obj>
 </main>
```

2.4.5.File directive

The file-directive provides a way to return another file than req-file-spec, each subsequent file-directive overrides the previous one.

2.4.6.Ipfile directive

The ipfile-directive is similar to the file-directive, the only difference is that xWRC will lookup the peer IP-address (this is the IP-address of the browser sending the request) and insert it between $dir-name and $file-name.

Example: ipfile="/abc/main.html" is requested from a browser at IP-address 192.168.0.22, then xWRC will in effect return a file "/abc/192.168.0.22.main.html" as response.

2.4.7.Command processing

This section explains how xWRC handles a command URL.

1. Handle each command and directive from left to right.
2. Each compound-cmd is passed on to xiControl, a concatination of all compound-commands is available as the <?request ccmd?> tag.
3. The wait-directive is deprecated and only supported for backward compatibility, it is on-the-fly replaced with the equivalent wait-cmd. One should use the wait-cmd for any new developments.
4. Each file-directive or ipfile-directive overwrites the previous one, and maps onto the <?request file?> tag.
5. The terse-, verbose- and xml-directives are mutually exclusive, occurrence of one resets presence of the other 2, i.e. ccmd=omd.0&xml&verbose&terse will finally have only the terse-directive remembered.
6. Finally xWRC starts generating response as follows:

```
if      (verbose-directive) {
  response = $verbose-result;
  }
else if (terse-directive) {
  response = $terse-result;
  }
else if (xml-directive) {
  response = $xml-result;
  }
else if (ipfile-directive) {
  response = ['/'] *[ $file-path '/' ] $peer-ip-address '.' $file-name;
  }
else if (file-directive) {
  response = ['/'] *[ $file-path '/' ] $file-name;
  }
else if (req-file-spec == '/icontrol.dll') {
  response = '/icontrol/<?server theme?>.html';
  }
else if (req-file-spec == '/’ *[ $file-path '/' ]) {
  response = '/’ *[ $file-path '/'] <?server theme?> '.html';
  }
else {
  response = req-file-spec;
  }
```

2.5.Structured tags

Tags are placeholders for fixed or variable data, and can be placed in pages to be served by xWRC.

When a client requests a page, the server will load the file representing the page and look for tags, replacing them with their actual value on-the-fly.

If the server does not know a certain tag, it will call xiControl in an attempt to let it replace the tag with actual data.

2.5.1.Tag syntax

Tags have a structured definition and are noted in following syntax:

```
tag      = ssi | format-1 | format-2

ssi      = '<!--#include file="' $template-file '" -->'

format-1 = '<#' branch node [predef=value ...] '#>'

format-2 = '<?' branch node [predef=value ...] '?>'
```

Where 'ssi' stands for Server Side Include, this is no longer supported as it can be replaced by <?include file=”$filespec”?>. Occurrence of an ssi will show a warning message in the response to suggest the adjustment.

Where 'format-1' was the original tag format and is still supported for backward compatibility.

Where 'format-2' was added because it is better suited for maintaining the HTML pages in a website authoring tool like Dreamweaver. Both formats will give the same result.

2.5.2.Tag branches

Branches are the highest hierarchy level in organizing tags, they are choosen such that tags are divided into logical groups, i.e. include, cookie, request, xml, ...

Per branch dedicated nodes are defined, some nodes have pre-defined states (i.e. no, yes), for these cases it is possible to insert an optional [arg=value] specification to replace this value with a customisable output. An example can be:

```
<?cookie abcd empty="use this string"?>

<?request loggedin no="loggedout" yes="loggedin"?>
```

Below table lists all branches and nodes supported by the xWRC server, with a detailed description:

Branch	Node	Predef	Description
include	[path=$pathspec] file=$filespec		Generic form of include, all tags in the included file(s) will be translated.
	path=$pathspec		Deprecated parameter, no longer supported.>
	file=$filespec		Filespec can be a combination of optional directory name(s) and a mandatory filename, either parts can be fixed or contain wildcard characters (* and ?) which will be enumerated upto 4 levels deep. Examples: <?include file="/icontrol/util/menu.html"?> <?include file="/icontrol/*.prj/menu*.html"?>
cookie	$name	empty	Inserts the value of the cookie $name, if the cookie is not set or empty and the 'empty' argument is specified, then this value will be used instead. Examples: <?cookie mycookie?> <?cookie mycookie empty="0001"?>
server	alias		xWRC short product name, **will be all lowercase**. Example: <?server alias?>
	author		xWRC author + copyright statement. Example: <?server author?>
	connections		The current number of network connections handled by the server. Example: <?server connections?>
	created		xWRC creation date of this version. Example: <?server created?>
	date		The current server UTC date in format 'yyyy/mm/dd'. Example: <?server date?>
	description		xWRC product description. Example: <?server description?>
	fontsize		The fontsize in pt to use in all server pages. Example: <?server fontsize?>
	hits		The number of hits (client requests) handled by xWRC. Example: <?server hits?>
	hostname		Name of the machine on which xWRC is running. Example: <?server hostname?>
	httpaddr		The IP address on which the xWRC HTTP server is listening. Example: <?server httpaddr?>

	httpport		The port on which the xWRC HTTP server is listening. Example: <?server httpport?>
	name		xWRC full product name. Example: <?server name?> will be replaced with 'xWRC'.
	platform		The HW platform on which xWRC is running, either 'Win32/x86', 'Linux/MIPS', 'Linux/ARM' or 'Linux/x86'. Example: <?server platform?>
	theme		The current server theme, used to determine server root page as /<theme>.html. Example: <?server theme?>
	time		The current server UTC time in format 'hh:mm:ss'. Example: <?server time?>
	uptime		The time during which xWRC has been running so far. Example: <?server uptime?>
	version		xWRC version using format 'x.y.z.t'. Example: <?server version?>
request	authmeth	none basic session ipadddr	Authentication method used for this request, 0=none, 1=basic, 2=session, 3=ipaddr. Example: <?request authmeth none="None" basic="Basic" session="Session" ipaddr="IpAddr"?>
	file		The name of the file requested as determined by req-file-spec, file-directive or ipfile-directive for this request. Example: <?request file?>
	filepath		The path part of the file requested as determined by req-file-spec, file-directive or ipfile-directive for this request. This is without the trailing slash. Example: <?request filepath?> could return ‘/project’
	loggedin	no yes	Indicates whether this request was made by a logged in user, 0=no, 1=yes. Example: <?request loggedin no="No" yes="Yes"?>
	loggedinas		Returns the name of the logged in user. Example: <?request loggedinas?>
	sessionid		The sessionid allocated for a logged in user. Example: <?request sessionid?>
	terse		Indicates the presence of the terse-directive in the request, there are 2 predefined values: no, yes.

			Example: <?request terse?> Example: <?request terse no="" yes="&terse"?>
	verbose		Indicates the presence of the verbose-directive in the request, there are 2 predefined values: no, yes. Example: <?request verbose?> Example: <?request verbose no="" yes="&verbose"?>
	xml		Indicates the presence of the xml-directive in the request, there are 2 predefined values: no, yes. Example: <?request xml?> Example: <?request xml no="" yes="&xml"?>
response	terse		The $terse-result after handling the request. Example: <?response terse?>
	verbose		The $verbose-result after handling the request. Example: <?response verbose?>
	xml		The $xml-result after handling the request. Example: <?response xml?>

xml	file=$xml-file
	node=$xml-node
			Generic form to include data from an xml formatted file.

	file=$xml-file		Specifies which xml formatted file to use, contents can be like this:

```
<!-- this is comment -->
<?xml optional header ?>
 <node1>
  <!-- this is comment -->
  <subnode1>value1</subnode1>

  <subnode2>
   <!-- this is comment -->
   <subnode21>value21</subnode21>
   </subnode2>
  </node1>
```

	node=$xml-node		Specifies which xml-node value to include from 'file', the format is: `xml-node = node-name *[ '.' node-name ]` Example: <?xml file="/icontrol/test.xml" node="node1.subnode2.subnode21"?> would return 'value21'.

2.6.AJAX support

AJAX stands for Asynchronous JavaScript and XML.

Opposed to the standard HTTP request/response behaviour where a client sends a request and reloads the complete page as returned by xWRC, it is also possible to send commands to xWRC over a separate communication channel that does not require complete reloading of the page.

For this we use JavaScript to initiate a separate connection with the server, send a command and handle the response data. The response data can be XML encoded or plain text depending on the output-directives specified.

Refer to the **wikipedia page on AJAX** for a more complete technical explanation.

Up to now we prepared a static HTML page, with command links and tags to represent the actual state of your PHC system, but it was always xWRC that made sure to compose the final HTML code sent to your browser.

With AJAX we can move part of that intelligence to your browser by using JavaScript code, as a result your webpage will become more responsive and less data is transferred from xWRC to your browser.

xWRC implements 3 modes of AJAX support:

2.6.1.Terse mode

Selected by adding the terse-directive to the structured command URL. Terse mode causes the $terse-result to be returned, it is up to the JavaScript to map these results onto the existing page.

Example:

```
http://myserver/icontrol.dll?ccmd=omd.0.toggle&terse
```

2.6.2.XML mode 0

Selected by adding the xml-directive to the structured command URL. XML mode 0 causes the $xml-result to be returned, it is up to the receiving JavaScript to map results onto the existing page.

Example:

```
http://myserver/icontrol.dll?ccmd=omd.0.toggle&xml
```

2.6.3.XML mode 1

Selected by adding a file-directive or ipfile-directive specifying an XML file (*.xml) to the structured command URL. XML mode 1 causes the XML file to be returned, thereby replacing any present tags, it is up to the receiving JavaScript to map the results onto the existing page.

Example:

```
http://myserver/icontrol.dll?ccmd=omd.0.toggle&ipfile="/icontrol/myproject.xml"
```

3. xiControl

3.1.Introduction

xiControl makes a PHC (© PEHA Deutschland) domotics system accesible via a standard webbrowser like Internet Explorer, Mozilla, Firefox, ... and is an integrated part of xWRC.

Following drawing gives an overview of the xiControl concept:

```
+-----------------------------------------------+                           +--------------------+
| Windows         +-----------+                 |                           |    WebBrowser      |
| or              |           |                 |  IP network               |     (=Client)      |
| Linux           |    xWRC   |<-------------------------------------------->|                    |
|                 |   Server  |                 |     HTTP                  |                    |
|                 |           |                 |                           +--------------------+
|                 |           |                 |
| +----------+    | --------- |                 |                                PHC-system
| |          |    |           |    +----------+ |                           +--------------------+
| |          |    |           |   +----------+| |                           | +-------+  +-----+ |
| | ini-file |<-->| xiControl |<--|   HTML   |+ |        XML-RPC Protocol   | |  STM  |->| MOD | |
| |          |    |           |   | Template |  |      +---------------------->|  v3   |  |     | |
| |          |    |           |   +---------/   |     /  TCP/IP             | +-------+  +-----+ |
| +---------/     +-----------+                 |    /                      +--------------------+
|                      ^                        |   /
|                      |          +----------+  |  /             +-------+  +--------------------+
|                      |          |          |<---+              |       |  | +-------+  +-----+ |
|                      +--------->|  Socket  |  |     TCP/IP     |  TCP  |  | |  STM  |  |     | |
|                                 |          |<----------------->|   2   |--->| v1/v2 |->| MOD | |
|                                 +----------+  |  EXT Protocol  | RS232 |  | +-------+  +-----+ |
+-----------------------------------------------+                +-------+  +--------------------+
```

3.1.1.PHC topology

The PHC system is a hierarchical system, where a STM is the master processor which controls a number of modules over an RS485 bus called the module-bus (MOD). The STM is programmed with the logic of how your system should work, i.e. if button1 is released then toggle output2, ...

The STM also has some internal objects like clocks and markers; clocks can be used to trigger periodic actions at a given time, whereas meakers are used to store a state. **xiControl is not able to read or change clocks or markers**.

The system running xiControl is hooked up to the STM via a TCP connection called the external-link (EXT), xiControl uses this link to send commands and receive responses. Check the **'Configuring'** help section.

The PHC modules itself are divided into classes, per class there is a maximum of 32 modules:

Class 0 Input modules, includes 24V/230V input modules (IMD), built-in modules with or without frame (IMW/ET0) and input/output consoles (TAB)
Class 1 Multifunction modules, includes IR modules (UIM), movement and light modules (BWM), built-in modules with or without frame (UTM/ET1)
Class 2 Output modules, includes 24V/230V/low-power output modules (OMD) and shutter modules (JRM)
Class 3 Analog modules, includes analog modules (AMD) and funk modules (FUI, FU2)
Class 4 Box modules, includes output modules (EBS), shutter modules (EBR) and dimmer modules (EBD)
Class 5 Dimmer modules, includes synchronous, asynchronous and universal dimmer modules (DIM)

A second RS485 bus called the system-bus (SYS) can connect multiple STM's to create a bigger system, this way one STM can request another STM to execute a command on his connected modules. **Attention: xiControl cannot send commands over this bus.**

3.2.Configuring

3.2.1.xiControl settings

xiControl configuration is specified in the [icontrol.0] section and supports following items:

```
[xicontrol.0]
; address of STM module connected to (0-7), default is 0
;stmaddr=0

; remote mode of connected STM module, 0=STMv1/v2, 1=STMv3, default is 0
;remmode=1

; remote ip-addres to connect over TCP to STM module, default is 127.0.0.1
; either the TCP/RS232 convertor ip-address, or the STMv3 ip-address
;remaddr=192.168.0.127

; remote ip-port to connect over TCP to STM module, default is 10000
; when remmode==1, you should set remport=6680
;remport=10000
```

3.2.2.Demo mode

At this point, xiControl will run in demo mode, although full functionality is provided, the number of PHC modules that can be used is limited to following:

- 1 system module
- 1 input module (imd.0/imw.0/tab.0/et0.0)
- 2 universal IR/input modules (uim.0/uim.1/utm.0/utm.1/bwm.0/bwm.1/et1.0/et1.1)
- 2 output/JRM modules (omd.0/omd.1/jrm.0/jrm.1)
- 1 analog module (amd.0/fui.0/fu2.0)
- 1 box module (ebs.0/ebr.0/ebd.0)
- 1 dimmer module (dim.0)

3.2.3.Licensed mode

If you want an unlimited operational version you will need to get a license for your xWRC server PC.

Unfortunately a license is not for free... the good news is that we offer it for 50,00 euros, this money is used to fund development/research costs to further enhance xiControl and give support.

To request a license you start xWRC with the -licreq option as follows, and send the log.txt file to 'icontrol@telenet.be'.

```
  ./xwrc.exe -licreq >log.txt
```

In response you will receive a mail with payment details, after payment is confirmed the license is sent to you.

3.3.Access, restrictions & security

iControl uses a structured command syntax in which modules are referenced with an 'alias', this alias depends on

3.3.1.xiControl access rules

This document describes how to control and report your PHC system using iControl.

```
;
; This section lists all rules applicable to xiControl objects
;
; Wildcards that apply: '?' can occur multiple times at any place except in a range
;                       '*' can occur once at any place except in a range
;                       '[' and ']' indicate a numerical range [from to]
;
; Syntax: { {inst-range} ':'} obj-range ',' access 1-16(',' grp-name)
;
;  inst-range   = '[' inst {'-' inst} *{',' inst {'-' inst} } ']'
;  inst         = 0..7
;
;  obj-range    = (stm-obj | mod-range) *{';' (stm-obj | mod-range) }
;
;  stm-obj      = 'stm.' stm-method
;  stm-method   = 'ping' | 'inquiry' | 'read' | ...
;
;  mod-range    = mod '.' '[' addr {'-' addr}  *{',' addr {'-' addr}} ']' {'.' chn-range }
;  mod          = 'imd' | 'imw' | 'tab' | 'et0' | 'uim' | 'utm' | 'bwm' | 'et1' | 'mcc' |
;                 'omd' | 'jrm' | 'amd' | 'fui' | 'fu2' | 'ebs' | 'ebr' | 'ebd' | 'dim'
;  addr         = 0..31
;
;  chn-range    = chn '.' '[' nr {'-' nr}  *{',' nr {'-' nr}} ']'
;  chn          = 'in' | 'ir' | 'im' | 'il' | 'out' | 'led' | 'fb'
;  nr           = 0..31
;
;  access       = 0    none
;                 1    read
;                 2    exec
;                 3    full
;
; ugrp-name     = 1 upto 16 user-groups the user belongs to
;
[xicontrol]
"stm.version;stm.date;stm.time;stm.read;stm.clkread"=1,grpAdmin
"omd.[0-3,16]*;dim*"=3,grpAdmin
"omd.[0-3,16]*;dim*"=1,grpGuest
```

3.4.Structured command URL

xiControl is called by xWRC when it encounters any of following conditions:

- a req-file-spec equal to ‘icontrol.dll’
- a compound-cmd in the query part of the request

3.4.1.Compound-cmd syntax

This document describes how to control and report your PHC system using iControl.

```
compound-cmd     = 'ccmd=' cmd *[';' cmd]

cmd              = ['!'] (stm-cmd | mod-cmd | wait-cmd)

'!'              = suppresses the outcome of this command from $terse-result and $xml-result

stm-cmd          = $method *['.' #parm]

mod-cmd          = $module '.' #addr ['.' $channel] '.' $method *['.' #parm] *[$oper #value]

wait-cmd         = 'wait.' #delay

#delay           = 0...60000 (milliseconds)

$module          = 'imd' | 'imw' | 'tab' | 'et0' | 'uim' | 'utm' | 'bwm' | 'et1' | 'mcc' |
                   'omd' | 'jrm' | 'amd' | 'fui' | 'fu2' | 'ebs' | 'ebr' | 'ebd' | 'dim'

#addr            = 0...31  // module address

$channel         = see below

$method          = see below

#parm            = see below

#value           = numerical value

$oper            = add-oper | sub-oper | mul-oper | div-oper | mod-oper |
                   xor-oper | and-oper | or-oper  | shl-oper | shr-oper

add-oper         = '%2B' | '+'  // addition operator

sub-oper         = '%2D' | '-'  // subtraction operator

mul-oper         = '%2A' | '*'  // multiplication operator

div-oper         = '%2F' | '/'  // division operator

mod-oper         = '%25' | '%%' // modulus operator

xor-oper         = '%5E' | '^'  // XOR operator

and-oper         = '%26'        // bitwise AND operator

or-oper          = '%7C' | '|'  // bitwise OR operator

shl-oper         = '<<'         // shift left operator

shr-oper         = '>>'         // shift right operator
```

Examples:

```
   http://myserver/icontrol.dll
                     ^
                     |
                     +- return default xiControl homepage


   http://myserver/icontrol/abc.html?ccmd=version;omd.0.out0.toggle;date
                             ^        ^
                             |        |
             file to return -+        +- compound-cmd to execute


   http://myserver/icontrol.dll?ccmd=version;omd.0.out0.toggle;date&file=/icontrol/abc.html
                                 ^                                   ^
                                 |                                   |
        compound-cmd to execute -+                   file to return -+


   http://myserver/?ccmd=version;omd.0.out0.toggle;date&file=/icontrol/abc.html
                     ^                                   ^
                     |                                   |
       compound-cmd  |                                   |
         to execute -+                   file to return -+


   http://myserver/icontrol.dll?ccmd=imd.0.in15.ingt2;wait.1000&file=/icontrol/abc.html
                                     ^                ^           ^
                                     |                |           |
      simulate ingt2 event on input -+                |    return this file as response
                                                      |
                              wait for 1000 millisec -+
```

3.4.2.Compound-cmd reference

Module	Channel	Method	Parm	Description
		wait	#delay	Will pause xiControl processing for the specified number of milliseconds. This command may prove helpful to give the PHC system time to execute a lengthy command (i.e. executing an 'all-off' command, setting a dimmer output, ...), after which xiControl can collect the correct module status to be returned in the response. • #delay (0-60000) expressed in milliseconds. Example: wait.500
[stm]				This section groups the STM commands. Some STM commands can only be issued by the 'admin' user, this is explicitly noted per command. Some STM commands are not implemented in this release, this is explicitly noted per command.
		linkstate		Returns the current state of the connection to the STM module, 0=off, 1=on. Not applicable for STMv3. Example: linkstate
			#state ack	Controls the state of the connection to the STM module, 0=off, 1=on, and returns the actual state. Example: linkstate.0.ack; stm.linkstate.1.ack
		stats		Returns statistics on the communication with the STM module. Example: stats; stm.stats
		ping		Verifies the presence of an STM module by sending a 'ping' message and awaiting a reply or a timeout. Example: ping;stm.ping
		inquiry		Obtains current system date, time and version from STM. Example: inquiry;stm.inquiry
		version		Returns STM version, makes use of the 'inquiry' command. Example: version
		date		Returns STM date, makes use of the 'inquiry' command. Example: date

		time		Returns STM time, makes use of the 'inquiry' command. Example: time
		disable	ack	Disables the connected STM, after this the STM will not process any events from modules on the PHC-bus or from xiControl. Requires the 'ack' parameter to be present as a safeguard measure. Example: disable.ack
		enable	ack	Enables the connected STM after being disabled, after this the STM will respond again to events from modules on the PHC-bus or from xiControl. Modules on the PHC-bus will try to send events that occured while the STM was disabled, this may cause undesired behaviour. To prevent this, use the 'init' command instead of 'enable'. Requires the 'ack' parameter to be present as a safeguard measure. Example: enable.ack
		init	ack	Enables the connected STM after being disabled, after this the STM will respond again to events from modules on the PHC-bus or from xiControl. This command causes a reset to be sent to all modules on the PHC-bus, thereby clearing any outstanding events that occured while the STM was disabled. Requires the 'ack' parameter to be present as a safeguard measure. Example: init.ack
		por	ack	Power-On-Reset, resets the connected STM which will send a reset to all modules on the PHC-bus. Requires the 'ack' parameter to be present as a safeguard measure. Example: por.ack
imd.0 - imd.31				These commands are applicable to following modules: • Input module 24V • Input module 230V • Input module 24V with LED • Input module 24V for switches Use module name and address to return the current status, example: imd.0 If successful, the current status of the inputs and the feedback LED's is returned as a bitstring, where: • bit0-15 = led0-led15, where a '1' indicates an 'on' LED • bit16-31 = in0-in15, where a '1' indicates a closed

				input Note: STM v1 returns the status of the lower 8 LED's only ! Example response: 0x80010104, means that in15 + in0 are closed and led8 and led2 are on.
	in0-in15	ingt0		Simulates 'IN > 0 sec' event, returns 0,example: imd.0.in2.ingt0
		outlt1		Simulates 'OUT < 1 sec' event, returns 0, example: imd.0.in2.outlt1
		ingt1		Simulates 'IN > 1 sec' event, returns 0, example: imd.0.in2.ingt1
		outgt1		Simulates 'OUT > 1 sec' event, returns 0, example: imd.0.in2.outgt1
		ingt2		Simulates 'IN > 2 sec' event, returns 0, example: imd.0.in2.ingt2
	led0-led15			Returns current status (0-1)
		on		Turn LED on, returns the current status, please note that some IMD's only have led0-7. Example: imd.0.led7.on
		off		Turn LED off, returns the current status, please note that some IMD's only have led0-7. Example: imd.0.led7.off
imw.0 - imw.31				These commands are applicable to following modules: • Input module flush mount 24V • Input module flush mount with LED Use module name and address to return the current status, example: imw.0 If successful, the current status of the inputs and the feedback LED's is returned as a bitstring, where: • bit0-3 = mrk0-mrk3, where a '1' indicates an 'on' merker • bit4-7 = led4-led7, where a '1' indicates an 'on' LED • bit16-23 = in0-in7, where a '1' indicates a closed input
	in0-in7	ingt0		Simulates 'IN > 0 sec' event, returns 0, example: imw.0.in2.ingt0
		outlt1		Simulates 'OUT < 1 sec' event, returns 0, example: imu.0.in2.outlt1
		ingt1		Simulates 'IN > 1 sec' event, returns 0, example: imw.0.in2.ingt1
		outgt1		Simulates 'OUT > 1 sec' event, returns 0, example: imw.0.in2.outgt1
		ingt2		Simulates 'IN > 2 sec' event, returns 0, example: imw.0.in2.ingt2
		out		Simulates ‘OUT’ event, returns 0, example: imw.0.in2.out
	led4-			Returns current status (0-1)

led7			
	on		Turn LED on, returns the current status, example: imw.0.led7.on
	off		Turn LED off, returns the current status, example: imw.0.led7.off
	toggle		Toggle LED between on and off, returns the current status, example: imw.0.led7.toggle
	blink		Start LED blinking, example: imw.0.led7.blink
mrk0-mrk3			Returns current status (0-1)
	set		Sets merker channel and returns the current status, example: imw.0.mrk0.set
	reset		Resets merker channel and returns the current status, example: imw.0.mrk0.reset
	setdelayed	#runtime	Sets merker channel after initial delay time, parameters: • #runtime (1-65535) in seconds. Example: imw.0.mrk0.setdelayed.5
	resetdelayed	#runtime	Resets merker channel after initial delay time, parameters: • #runtime (1-65535) in seconds. Example: imw.0.mrk0.resetdelayed.5
	settimed	#runtime	Sets merker channel for given time after which it is reset, parameters: • #runtime (1-65535) in seconds. Example: imw.0.mrk0.settimed.5
	resettimed	#runtime	Resets merker channel for given time after which it is set, parameters: • #runtime (1-65535) in seconds. Example: imw.0.mrk0.resettimed.5
	timeadd	#runtime	Adds time to the current running time, parameters: • #runtime (1-65535) in seconds. Example: imw.0.mrk0.timeadd.5
	timeset	#runtime	Sets new running time, parameters: • #runtime (1-65535) in seconds. Example: imw.0.mrk0.timeset.5
	timeabort		Stops running time, example: imw.0.mrk0.timeabort
fb4-fb7	on		Simulates 'LED on' feedback, returns 0, example: imw.0.fb7.on
	off		Simulates 'LED off' feedback, returns 0, example:

				imw.0.fb7.off
		blink		Simulates 'LED blinking' feedback, returns 0, example: imw.0.fb7.blink
tab.0 - tab.31				These commands are applicable to following modules: • Control and reporting pad with 16 pushbuttons Use module name and address to return the current status, example: tab.0 If successful, the current status of the inputs and the feedback LED's is returned as a bitstring, where: • bit0-15 = led0-led15, where a '1' indicates an 'on' LED • bit16-31 = in0-in15, where a '1' indicates a closed input
	in0-in15	ingt0		Simulates 'IN > 0 sec' event, returns 0, example: tab.0.in2.ingt0
		outlt1		Simulates 'OUT < 1 sec' event, returns 0, example: tab.0.in2.outlt1
		ingt1		Simulates 'IN > 1 sec' event, returns 0, example: tab.0.in2.ingt1
		outgt1		Simulates 'OUT > 1 sec' event, returns 0, example: tab.0.in2.outgt1
		ingt2		Simulates 'IN > 2 sec' event, returns 0, example: tab.0.in2.ingt2
		out		Simulates ‘OUT’ event, returns 0, example: tab.0.in2.out
	led0-led15			Returns current status (0-1), example: tab.0.led0
		on		Turn LED on, returns the current status, example: tab.0.led7.on
		off		Turn LED off, returns the current status, example: tab.0.led7.off
		toggle		Toggle LED between on and off, returns the current status, example: tab.0.led7.toggle
		blink		Start LED blinking, example: tab.0.led7.blink
	fb0-fb14	on		Simulates 'LED on' feedback, returns 0, example: tab.0.fb7.on
		off		Simulates 'LED off' feedback, returns 0, example: tab.0.fb7.off
		blink		Simulates 'LED blinking' feedback, returns 0, example: tab.0.fb7.blink
et0.0 - et0.31				These commands are applicable to following modules: • Flush mount input module with 2 inputs (in0/in4) • Flush mount input module with 4 inputs (in0/in2/in4/in6)

				• Flush mount input module with 8 inputs (in0-in7) Use module name and address to return the current status, example: et0.0 If successful, the current status of the inputs and the feedback LED's is returned as a bitstring, where: • bit0-8 = led0-led8, where a '1' indicates an 'on' LED • bit16-23 = in0-in7, where a '1' indicates a closed input
	in0-in7	ingt0		Simulates 'IN > 0 sec' event, returns 0, example: tab.0.in2.ingt0
		outlt1		Simulates 'OUT < 1 sec' event, returns 0, example: tab.0.in2.outlt1
		ingt1		Simulates 'IN > 1 sec' event, returns 0, example: tab.0.in2.ingt1
		outgt1		Simulates 'OUT > 1 sec' event, returns 0, example: tab.0.in2.outgt1
		ingt2		Simulates 'IN > 2 sec' event, returns 0, example: tab.0.in2.ingt2
		out		Simulates ‘OUT’ event, returns 0, example: tab.0.in2.out
	led0-led8			Returns current status (0-1), example: et0.0.led0
		on		Turn LED on, returns the current status, example: tab.0.led7.on
		off		Turn LED off, returns the current status, example: tab.0.led7.off
		toggle		Toggles feedback LED between on and off and returns the current status, example: tab.0.led7.toggle
		blink		Starts feedback LED blinking, example: tab.0.led7.blink
	fb0-fb8	on		Simulates 'LED on' feedback, returns 0, example: tab.0.fb7.on
		off		Simulates 'LED off' feedback, returns 0, example: tab.0.fb7.off
		blink		Simulates 'LED blinking' feedback, returns 0, example: tab.0.fb7.blink
uim.0 - uim.31				These commands are applicable to following modules: • Infrared input module Use module name and address to return the current status, example: uim.0 If successful, the current status of the inputs is returned as a bitstring, where: • bit23-27 = in7-in11, where a '1' indicates a closed input

	ir0-ir6	onoffin		Channel 'ir0' represents the base level of the IR remote, pressing the '1'...'6' button respectively on the IR remote will select 'ir%' (where %=1...6) to send below commands on. Simulates the ON/OFF button of an IR remote being pressed, returns 0, example: uim.0.ir0.onoffin
		onoffout		Simulates the ON/OFF button of an IR remote being released, returns 0, example: uim.0.ir0.onoffout
		upin		Simulates the UP button of an IR remote being pressed, returns 0, example: uim.0.ir0.upin
		upout		Simulates the UP button of an IR remote being released, returns 0, example: uim.0.ir0.upout
		downin		Simulates the DOWN button of an IR remote being pressed, returns 0, example: uim.0.ir0.downin
		downout		Simulates the DOWN button of an IR remote being released, returns 0, example: uim.0.ir0.downout
	in7-in11	ingt0		Channel 'in7' on the UIM represents the pushbutton of the IR receiver, 'in8' till 'in11' are the 4 external inputs to the UIM. Simulates 'IN > 0 sec' event, returns 0
		outlt1		Simulates 'OUT < 1 sec' event, returns 0
		ingt1		Simulates 'IN > 1 sec' event, returns 0
		outgt1		Simulates 'OUT > 1 sec' event, returns 0
		ingt2		Simulates 'IN > 2 sec' event, returns 0
		out		Simulates ‘OUT’ event, returns 0
utm.0 - utm.31				These commands are applicable to following modules: • Flush mount input module (in0) • Flush mount input module (in0-1) • Flush mount input module (in0-3) Use module name and address to return the current status, example: utm.0 If successful, the current status of the inputs and the feedback LED's is returned as a bitstring, where: • bit0-1 = led0-led1, where a '1' indicates an 'on' LED • bit16-19 = in0-in3, where a '1' indicates a closed input • bit24-27 = in8-in11, where a '1' indicates a closed input
	in0-in3, in8-in11	ingt0		Simulates 'IN > 0 sec' event, returns 0, example: utm.0.in2.ingt0
		outlt1		Simulates 'OUT < 1 sec' event, returns 0, example: utm.0.in2.outlt1
		ingt1		Simulates 'IN > 1 sec’ event, returns 0, example: utm.0.in2.ingt1
		outgt1		Simulates 'OUT > 1 sec' event, returns 0, example: utm.0.in2.outgt1

		ingt2		Simulates 'IN > 2 sec' event, returns 0, example: utm.0.in2.ingt2
		out		Simulates ‘OUT’ event, returns 0, example: utm.0.in2.out
	led0- led1			Returns current status (0-1)
		on		Turn LED on, returns current status, please note that some UTM's only have led0 or no led's, example: utm.0.led0.on
		off		Turn LED off, returns current status, please note that some UTM's only have led0 or no led's, example: utm.0.led0.off
bwm.0 - bwm.31				These commands are applicable to following modules: - Flush mount movement detector 180° Use module name and address to return the current status, example: bwm.0 If successful, the current status of the inputs and the feedback LED's is returned as a bitstring, where: - bit16-17 = im0-im1, where a '1' indicates a 'movement detected' condition - bit19 = il3, where a '1' indicates a 'light' condition - bit24-27 = in8-in11, where a '1' indicates a closed input
	in8-in11	ingt0		Simulates 'IN > 0 sec' event, returns 0, example: bwm.0.in2.ingt0
		outlt1		Simulates 'OUT < 1 sec' event, returns 0, example: bwm.0.in2.outlt1
		ingt1		Simulates 'IN > 1 sec' event, returns 0, example: bwm.0.in2.ingt1
		outgt1		Simulates 'OUT > 1 sec' event, returns 0, example: bwm.0.in2.outgt1
		ingt2		Simulates 'IN > 2 sec' event, returns 0, example: bwm.0.in2.ingt2
		out		Simulates ‘OUT’ event, returns 0, example: bwm.0.in2.out
	im0-im1	stop		Simulates ‘stop movement’ event, returns 0, example: bwm.0.im0.stop
		start		Simulates ‘start movement’ event, returns 0, example: bwm.0.im0.start
	il3	dark		Simulates ‘dark’ event, returns 0, example: bwm.0.il3.dark
		light		Simulates ‘light’ event, returns 0, example: bwm.0.il3.light
et1.0 - et1.31				These commands are applicable to following modules: - Flush mount input module with 2 inputs (in0/in4) - Flush mount input module with 4 inputs (in0/in2/in4/in6) - Flush mount input module with 8 inputs (in0-in7) Use module name and address to return the current status,

				example: et1.0 If successful, the current status of the inputs and the feedback LED's is returned as a bitstring, where: • bit0-8 = led0-led8, where a '1' indicates an 'on' LED • bit16-23 = in0-in7, where a '1' indicates a closed input
	in0-in7	ingt0		Simulates 'IN > 0 sec' event, returns 0, example: tab.0.in2.ingt0
		outlt1		Simulates 'OUT < 1 sec' event, returns 0, example: tab.0.in2.outlt1
		ingt1		Simulates 'IN > 1 sec' event, returns 0, example: tab.0.in2.ingt1
		outgt1		Simulates 'OUT > 1 sec' event, returns 0, example: tab.0.in2.outgt1
		ingt2		Simulates 'IN > 2 sec' event, returns 0, example: tab.0.in2.ingt2
		out		Simulates ‘OUT’ event, returns 0, example: tab.0.in2.out
	led0-led8			Returns current status (0-1), example: et1.0.led0
		on		Turn LED on, returns current status, example: tab.0.led7.on
		off		Turn LED off, returns current status, example: tab.0.led7.off
		toggle		Toggle LED between on and off, returns current status, example: tab.0.led7.toggle
		blink		Start LED blinking, example: tab.0.led7.blink
	fb0-fb8	on		Simulates 'LED on' feedback, returns 0, example: tab.0.fb7.on
		off		Simulates 'LED off' feedback, returns 0, example: tab.0.fb7.off
		blink		Simulates 'LED blinking' feedback, returns 0, example: tab.0.fb7.blink
mcc.0 - mcc.31				These commands are applicable to following modules: • Multi control center Use module name and address to return the current status, example: mcc.0 If successful, the current status of the inputs and the feedback LED's is returned as a bitstring, where: • bit0-8 = led0-led8, where a '1' indicates an 'on' LED • bit16-23 = in0-in7, where a '1' indicates a closed input
	in0-in7	ingt0		Simulates 'IN > 0 sec' event, returns 0, example: tab.0.in2.ingt0
		outlt1		Simulates 'OUT < 1 sec' event, returns 0, example: tab.0.in2.outlt1
		ingt1		Simulates 'IN > 1 sec' event, returns 0, example:

				tab.0.in2.ingt1
		outgt1		Simulates 'OUT > 1 sec' event, returns 0, example: tab.0.in2.outgt1
		ingt2		Simulates 'IN > 2 sec' event, returns 0, example: tab.0.in2.ingt2
		out		Simulates ‘OUT’ event, returns 0, example: tab.0.in2.out
	led0-led8			Returns current status (0-1), example: mcc.0.led0
		on		Turn LED on, returns current status, example: tab.0.led7.on
		off		Turn LED off, returns current status, example: tab.0.led7.off
		toggle		Toggle LED between on and off, returns current status, example: tab.0.led7.toggle
		blink		Start LED blinking, example: tab.0.led7.blink
	fb0-fb8	on		Simulates 'LED on' feedback, returns 0, example: tab.0.fb7.on
		off		Simulates 'LED off' feedback, returns 0, example: tab.0.fb7.off
		blink		Simulates 'LED blinking' feedback, returns 0, example: tab.0.fb7.blink
omd.0 - omd.31				These commands are applicable to following modules: • Output module 230V/4A • Output module 230V/10A • Output module 24V • Output module 230V/16A • Output module EVG Use module name and address to return the current status, example: omd.0 If successful, the current status of the outputs is returned as a bitstring, where: • bit0-7 = out0-out7, where a '1' indicates an 'on' output Example response: 0x00000010, means that out4 is on.
	out0-out7			Returns current status (0-1), example: omd.0.out0
		on		Switches output on, example: omd.0.out2.on
		off		Switches output off, example: omd.0.out2.off
		onlocked		Switches output on and locks it, this prevents any further commands to this channel except 'unlock', example: omd.0.out2.onlocked
		offlocked		Switches channel off and locks it, this prevents any further commands to this channel except 'unlock', example: omd.0.out2.offlocked
		toggle		Toggles output between on and off, example: omd.0.out3.toggle
		unlock		Unlocks output, making it available for any command, example: omd.0.out2.unlock

		ondelayed	#delaytime	Switches output delayed on, parameters: • #delaytime (1-65535 seconds). Example: omd.0.out3.ondelayed.10
		offdelayed	#delaytime	Switches output delayed off, parameters: • #delaytime (1-65535 seconds). Example: omd.0.out3.offdelayed.10
		ontimed	#runtime	Switches output on for given time, parameters: • #runtime (1-65535 seconds). Example: omd.0.out3.ontimed.10
		offtimed	#runtime	Switches output off for given time, parameters: • #runtime (1-65535 seconds). Example: omd.0.out3.offtimed.10
		toggledelayed	#delaytime	Toggles output delayed between on/off, parameters: • #delaytime (1-65535 seconds). Example: omd.0.out3.toggledelayed.10
		toggletimed	#runtime	Toggles output temporarily between on/off, parameters: • #runtime (1-65535 seconds). Example: omd.0.out3.toggletimed.10
		lock		Locks output, preventing any commands to this output except 'unlock', example: omd.0.out2.lock
		locktimed		Locks output for the current running time, example: omd.0.out3.locktimed
		timeadd	#runtime	Extends the current running time, parameters: • #runtime (1-65535 seconds). Example: omd.0.out3.timeadd.10
		timeset	#runtime	Sets the current running time, parameters: • #runtime (1-65535 seconds). Example: omd.0.out3.timeset.10
		timeabort		Terminates the current running time, example: omd.0.out2.timeabort
	out6-out7	timerondelayed	#delaytime	Only applicable to the 16A output module. Turns timer on after initial delay, this is reported as a fb%.timingon event

				• #delaytime (1-65535 1/10 seconds). Example: omd.0.out7.timerondelayed.100
		timeroffdelayed	#delaytime	Only applicable to the 16A output module. Turns timer off after initial delay, this is reported as a fb%.timingoff event • #delaytime (1-65535 1/10 seconds). Example: omd.0.out7.timeroffdelayed.100
		timerontimed	#runtime	Only applicable to the 16A output module. Turns timer on for a given time and then back off, this is reported as a fb%.timingon or fb%.timingoff event respectively • #runtime (1-65535 1/10 seconds). Example: omd.0.out7.timerontimed.100
		timerabort		Only applicable to the 16A output module. Stops the timer but leaves the on/off state, this is reported as a fb%.timingabort event, example: omd.0.out7.timerabort
	fb0-fb7	on		Simulates feedback for an output being switched 'on', can be used in PHC programming to trigger an action, returns 0, example: omd.0.fb2.on
		off		Simulates feedback for an output being switched 'off', can be used in PHC programming to trigger an action, returns 0, example: omd.0.fb2.off
	fb6-fb7	timeron		Only applicable to the 16A output module. Simulates feedback for a timer output being switched on, returns 0, example: omd.0.fb7.timeron
		timerabort		Only applicable to the 16A output module. Simulates feedback for a timer output timer being aborted, returns 0, example: omd.0.fb7.timerabort
		timeroff		Only applicable to the 16A output module. Simulates feedback for a timer output being switched off, returns 0, example: omd.0.fb7.timeroff
jrm.0 - jrm.31				These commands are applicable to following modules: • Rolladen/Jalousiemodul Use module name and address to return the current status, example: jrm.0 Due to it's design a JRM does NOT return it's status, you need to use the feedback signals to determine the actual status.
	out0-out3	stop	#priolevel	Stops either up or down operation, parameters: • #priolevel (0-5) Example: jrm.1.out3.stop.1

toggleup	#priolevel #priolock #runtime	Toggles between stop and up operation, parameters: • #priolevel (0-5) • #priolock (0-1) • #runtime (1-65535) expressed in 1/10 seconds Example: jrm.0.out1.toggleup.1.1.25
toggledown	#priolevel #priolock #runtime	Toggles between stop and down operation, parameters: • #priolevel (0-5) • #priolock (0-1) • #runtime (1-65535) expressed in 1/10 seconds Example: jrm.0.out1.toggledown.1.1.25
up	#priolevel #priolock #runtime	Starts up operation, parameters: • #priolevel (0-5) • #priolock (0-1) • #runtime (1-65535) expressed in 1/10 seconds Example: jrm.0.out1.up.1.1.25
down	#priolevel #priolock #runtime	Starts down operation, parameters: • #priolevel (0-5) • #priolock (0-1) • #runtime (1-65535) expressed in 1/10 seconds Example: jrm.0.out1.down.1.1.25
tipup	#priolevel #priolock #tiptime	Starts tipup operation, parameters: • #priolevel (0-5) • #priolock (0-1) • #tiptime (1-65535) expressed in 1/10 seconds Example: jrm.0.out1.tipup.1.1.5
tipdown	#priolevel #priolock #tiptime	Starts tipdown operation, parameters: • #priolevel (0-5) • #priolock (0-1) • #tiptime (1-65535) expressed in 1/10 seconds Example: jrm.0.out1.tipdown.1.1.5
priolock	#priomask	Locks one or more priority levels, parameters: • #priomask (0x00-0x3F), a bitwise combination of priority levels 0 upto 5, where level 0=1, level 1=2, level 2=4, level 3=8, level 4=16 and level 5=32. Example: jrm.0.out1.priolock.0x1C
priounlock	#priomask	Unlocks one or more priority levels, parameters:

		• #priomask (0x00-0x3F), a bitwise combination of priority levels 0 upto 5, where level 0=1, level 1=2, level 2=4, level 3=8, level 4=16 and level 5=32. Example: jrm.0.out1.priounlock.0x1C
learnon		Turns learning mode on, example: jrm.0.out1.learnon
learnoff		Turns learning mode off, example: jrm.0.out1.learnoff
prioset	#priomask	Sets one or more priority levels, parameters: • #priomask (0x00-0x3F), a bitwise combination of priority levels 0 upto 5, where level 0=1, level 1=2, level 2=4, level 3=8, level 4=16 and level 5=32. Example: jrm.0.out1.prioset.0x12
prioreset	#priomask	Resets one or more priority levels, parameters: • #priomask (0x00-0x3F), a bitwise combination of priority levels 0 upto 5, where level 0=1, level 1=2, level 2=4, level 3=8, level 4=16 and level 5=32. Example: jrm.0.out1.prioreset.0x12
delayedup	#priolevel #priolock #delaytime #runtime	After initial delaytime, the 'up' side of out% will be activated during runtime, parameters: • #priolevel (0-5) • #priolock (0-1) • #delaytime (1-65535) expressed in 1/10 seconds • #runtime (1-65535) expressed in 1/10 seconds Example: jrm.0.out1.delayedup.1.1.5.25
tipdelayedup	#priolevel #priolock #delaytime #runtime #tiptime	After initial delaytime, the 'up' side of out% will be activated during runtime, after that the 'down' side of out% will be activated during tiptime, parameters: • #priolevel (0-5) • #priolock (0-1) • #delaytime (1-65535) expressed in 1/10 seconds • #runtime (1-65535) expressed in 1/10 seconds • #tiptime (1-65535) expressed in 1/10 seconds Example: jrm.0.out1.tipdelayedup.1.1.5.25.1
delayeddown	#priolevel #priolock #delaytime #runtime	After initial delaytime, the 'down' side of out% will be activated during runtime, parameters: • #priolevel (0-5) • #priolock (0-1) • #delaytime (1-65535) expressed in 1/10 seconds • #runtime (1-65535) expressed in 1/10 seconds

				Example: jrm.0.out1.delayeddown.1.1.5.25
		tipdelayeddown	#priolevel #priolock #delaytime #runtime #tiptime	After initial delaytime, the 'down' side of out% will be activated during runtime, after that the 'up' side of out% will be activated during tiptime, parameters: • #priolevel (0-5) • #priolock (0-1) • #delaytime (1-65535) expressed in 1/10 seconds • #runtime (1-65535) expressed in 1/10 seconds • tiptime (1-65535) expressed in 1/10 seconds Example: jrm.0.out1.tipdelayeddown.1.1.5.25.1
	out4-out7	timerondelayed	#delaytime	Turns timer on after initial delay, this is reported as a fb%.timingon event • #delaytime (1-65535 1/10 seconds). Example: jrm.0.out7.timerondelayed.100
		timeroffdelayed	#delaytime	Turns timer off after initial delay, this is reported as a fb%.timingoff event • #delaytime (1-65535 1/10 seconds). Example: jrm.0.out7.timeroffdelayed.100
		timerontimed	#runtime	Turns timer on for a given time and then back off, this is reported as a fb%.timingon or fb%.timingoff event respectively • #runtime (1-65535 1/10 seconds). Example: jrm.0.out7.timerontimed.100
		timerabort		Stops the timer but leaves the on/off state, this is reported as a fb%.timingabort event, example: jrm.0.out7.timerabort
	fb0-fb3	upon		Simulates feedback of a channel starting up operation, can be used in PHC programming to trigger an action, returns 0, example: jrm.0.fb0.upon
		downon		Simulates feedback of a channel starting down operation, can be used in PHC programming to trigger an action, returns 0, example: jrm.0.fb0.downon
		upoff		Simulates feedback of a channel stopping up operation, can be used in PHC programming to trigger an action, returns 0, example: jrm.0.fb0.upoff
		downoff		Simulates feedback of a channel stopping down operation, can be used in PHC programming to trigger an action, returns 0, example: jrm.0.fb0.downoff
	fb4-fb7	timeron		Simulates feedback for a timer output being switched on, returns 0, example: jrm.0.fb7.timeron
		timerabort		Simulates feedback for a timer output timer being aborted, returns 0, example: jrm.0.fb7.timerabort
		timeroff		Simulates feedback for a timer output being switched off,

				returns 0, example: jrm.0.fb7.timeroff
amd.0 - amd.31				These commands are applicable to following modules: • Analogue module Use module name and address to return the current status, example: amd.0 If successful, the current status of the output is returned as a bitstring, where: • bit0 = out0, where a '1' indicates the output is turned 'on' • bit8-15 = level0, a value of 0-255 indicating the output level Example response: 0x00FF0001, means a level0=255 and out0 is 'on'.
	out0			Returns current output level (0-255), example: amd.0.out0
		onmaxmem		Turn output on at maximum level and store in memory, example: amd.0.out0.onmaxmem
		onmax		Turn output on at maximum level, example: amd.0.out0.onmax
		off		Turn output off, example: amd.0.out0.off
		togglemaxmem		Toggle output between off and memory level, example: amd.0.out0.togglemaxmem
		togglemax		Toggle output between off and maximum level, example: amd.0.out0.togglemax
		dimreverse	#runtime	Reverse trimming direction, parameters: • #runtime (1-160 seconds) to dim from off to max level, if omitted then run=5 will be used. Example: amd.0.out0.dimreverse.10
		dimup	#runtime	Trim up from current till maximum level, parameters: • #runtime (1-160 seconds) to dim from off to max level, if omitted then run=5 will be used. Example: amd.0.out0.dimup.10
		dimdown	#runtime	Trim down from current level till off, parameters: • #runtime (1-160 seconds) to dim from max level to off, if omitted then run=5 will be used. Example: amd.0.out0.dimdown.10
		stopmem		Stop output and store current level in memory, example: amd.0.out0.down.10
		togglemem		Toggle output between off and memory level, example: amd.0.out0.togglemem

		onmem		Turn dimmer on using memory level, example: amd.0.out0.memon
		stopdia1		Stop output and store current level in DIA1, example: amd.0.out0.stopdia1
		toggledia1		Toggle output between off and DIA1 level, example: amd.0.out0.toggledia1
		ondia1		Turn output on using DIA1 value, example: amd.0.out0.ondia1
		stopdia2		Stop output and store current level in DIA2, example: amd.0.out0.stopdia2
		toggledia2		Toggle output between off and DIA2 level, example: amd.0.out0.toggledia2
		ondia2		Turn output on using DIA2 level, example: amd.0.out0.ondia2
		stopdia3		Stop output and store current level in DIA3, example: amd.0.out0.stopdia3
		toggledia3		Toggle output between off and DIA3 level, example: amd.0.out0.toggledia3
		ondia3		Turn output on using DIA3 level, example: amd.0.out0.ondia3
		dimset	#level #runtime	Set output to level, parameters: • #level (0-255), if omitted then 128 will be used. • #runtime (1-160 seconds), if omitted then 5 will be used. Example: amd.0.out0.dimset.127.4
		trimcopy		Copy current level to target level and start trim function, example: amd.0.out0.trimcopy
		trimset	#level	Set target level and start trim function, parameters: • #level (0-255), if omitted then 128 will be used. Example: amd.0.out0.trimset.99
		trimtoggle		Toggle trim function between on/off, example: amd.0.out0.trimtoggle
		trimoff		Turn trim function off, example: amd.0.out0.trimoff
		trimon		Turn trim function on, example: amd.0.out0.trimon
	in0	on		Output was turned on, can be used in PHC programming to trigger an action.
		off		Output was turned off, can be used in PHC programming to trigger an action.
		lower		Current output level is lower than target level, can be used in PHC programming to trigger an action.
		equal		Current output level is equal to target level, can be used in PHC programming to trigger an action.
		higher		Current output level is higher than target level, can be used in PHC programming to trigger an action.
		trimon		Trim function was turned on, can be used in PHC programming to trigger an action.

		trimoff		Trim function was turned off, can be used in PHC programming to trigger an action.
	fb0	on		Output was turned on, can be used in PHC programming to trigger an action, returns 0.
		off		Output was turned off, can be used in PHC programming to trigger an action, returns 0.
		trimon		Trim function was turned on, can be used in PHC programming to trigger an action, returns 0.
		trimoff		Trim function was turned off, can be used in PHC programming to trigger an action, returns 0.
fui.0 - fui.31				These commands are applicable to following modules: • Funkinterface (Easyclick) • Funkinterface E/A (Easywave) Use module name and address to return the current status, example: fui.0 As this module doesn't return any status, 0 is returned.
	in0-in31	Oingt0		Simulates pressing the 'O' button on the RF remote control, returns 0, example: fui.0.in31.Oingt0
		Iingt0		Simulates pressing the 'I' button on the RF remote control, returns 0, example: fui.0.in31.Iingt0
		Oout		Simulates releasing the 'O' button on the RF remote control, returns 0, example: fui.0.in31.Oout
		Iout		Simulates releasing the 'I' button on the RF remote control, returns 0, example: fui.0.in31.Iout
	out0-out31	toggle		Toggle output between on and off, only applicable to EasyWave E/A modules, example: fui.0.out31.toggle
		off		Switch output off, only applicable to EasyWave E/A modules, example: fui.0.out31.off
fu2.0 - fu2.31				These commands are applicable to following modules: • Funkinterface E/A (Easyclick) Use module name and address to return the current status, example: fu2.0 As this module doesn't return any status, 0 is returned.
	in0-in31	Oingt0		Simulates 'IN > 0 sec' event for the 'O' button on the RF remote control, returns 0, example: fu2.0.in31.Oingt0
		Iingt0		Simulates 'IN > 0 sec' event for the 'I' button on the RF remote control, returns 0, example: fu2.0.in31.Iingt0
		Oout		Simulates 'OUT' event for the 'O' button on the RF remote control, returns 0, example: fu2.0.in31.Oout
		Iout		Simulates 'OUT' event for the 'I' button on the RF remote control, returns 0, example: fu2.0.in31.Iout
		Ooutlt1		Simulates 'OUT < 1 sec' event for the 'O' button on the RF remote control, returns 0, example: fu2.0.in31.Ooutlt1
		Oingt1		Simulates 'IN > 1 sec' event for the 'O' button on the RF

				remote control, returns 0, example: fu2.0.in31.Oingt1
		Ooutgt1		Simulates 'OUT > 1 sec' event for the 'O' button on the RF remote control, returns 0, example: fu2.0.in31.Ooutgt1
		Oingt2		Simulates 'IN > 2 sec' event for the 'O' button on the RF remote control, returns 0, example: fu2.0.in31.Oingt2
		Ioutlt1		Simulates 'OUT < 1 sec' event for the 'I' button on the RF remote control, returns 0, example: fu2.0.in31.Ioutlt1
		Iingt1		Simulates 'IN > 1 sec' event for the 'I' button on the RF remote control, returns 0, example: fu2.0.in31.Iingt1
		Ioutgt1		Simulates 'OUT > 1 sec' event for the 'I' button on the RF remote control, returns 0, example: fu2.0.in31.Ioutgt1
		Iingt2		Simulates 'IN > 2 sec' event for the 'I' button on the RF remote control, returns 0, example: fu2.0.in31.Iingt2
	out0-out31	on		Switches output on, example: fu2.0.out31.on
		off		Switches output off, example: fu2.0.out31.off
		toggle		Toggles output between on and off, example: fu2.0.out31.toggle
		ondelayed	#delaytime	Switches output delayed on, parameters: • #delaytime (1-65535 seconds). Example: fu2.0.out31.ondelayed.100
		offdelayed	#delaytime	Switches output delayed off, parameters: • #delaytime (1-65535 seconds). Example: fu2.0.out31.offdelayed.100
		ontimed	#runtime	Switches output on for given time, parameters: • #runtime (1-65535 seconds). Example: fu2.0.out31.ontimed.100
		offtimed	#runtime	Switches output off for given time, parameters: • #runtime (1-65535 seconds). Example: fu2.0.out31.offtimed.100
		dimreverse	#runtime	Reverses dimming direction, parameters: • #runtime (1-160 seconds) to dim from off to max level, if omitted then 5 will be used. Example: fu2.0.out31.dimreverse.20
		dimup	#runtime	Dim up from current till maximum level, parameters: • #runtime (1-160 seconds) to dim from off to max level, if omitted then 5 will be used. Example: fu2.0.out31.dimup.10

		dimdown	#runtime	Dim down from current level till off, parameters: • #runtime (1-160 seconds) to dim from off to max level, if omitted then 5 will be used. Example: fu2.0.out31.dimdown.10
		dimset	#level #runtime	Start dimming to level, parameters: • #level (0-255), if omitted then 128 will be used. • #runtime (1-160 seconds), if omitted then 5 will be used. Example: fu2.0.out31.dimset.99.5
		jrmup		Sends an 'up' command to an RF JRM module, example: fu2.0.out31.jrmup
		jrmdown		Send an 'down' command to an RF JRM module, example: fu2.0.out31.jrmdown
		jrmstop		Send a ‘stop’ command to an RF JRM module, example: fu2.0.out31.jrmstop
ebs.0 - ebs.31				These commands are applicable to following modules: • Recessed installation output box Use module name and address to return the current status, example: ebs.0 If successful, the current status of the inputs and the outputs is returned as a bitstring, where: • bit0-5 = out0-out5, where a '1' indicates an 'on' output • bit24-31 = in8-in15, where a '1' indicates a closed input
	in8-in15	ingt0		Simulates 'IN > 0 sec' event, returns 0, example: ebs.0.in8.ingt0
		outlt1		Simulates 'OUT < 1 sec' event, returns 0, example: ebs.0.in8.outlt1
		ingt1		Simulates 'IN > 1 sec' event, returns 0, example: ebs.0.in8.ingt1
		outgt1		Simulates 'OUT > 1 sec' event, returns 0, example: ebs.0.in8.outgt1
		ingt2		Simulates 'IN > 2 sec' event, returns 0, example: ebs.0.in8.ingt2
		out		Simulates ‘OUT’ event, returns 0, example: ebs.0.in8.out
	out0-out5			Returns current status (0-1), example: ebs.0.out0
		on		Switches output on, example: ebs.0.out2.on
		off		Switches output off, example: ebs.0.out2.off
		onlocked		Switches output on and locks it, this prevents any further commands to this channel except 'unlock', example:

		ebs.0.out2.onlocked
offlocked		Switches channel off and locks it, this prevents any further commands to this channel except 'unlock', example: ebs.0.out2.offlocked
toggle		Toggles output between on and off, example: ebs.0.out3.toggle
unlock		Unlocks output, making it available for any command, example: ebs.0.out2.unlock
ondelayed	#delaytime	Switches output delayed on, parameters: • #delaytime (1-65535 seconds). Example: ebs.0.out3.ondelayed.10
offdelayed	#delaytime	Switches output delayed off, parameters: • #delaytime (1-65535 seconds). Example: ebs.0.out3.offdelayed.10
ontimed	#runtime	Switches output on for given time, parameters: • #runtime (1-65535 seconds). Example: ebs.0.out3.ontimed.10
offtimed	#runtime	Switches output off for given time, parameters: • #runtime (1-65535 seconds). Example: ebs.0.out3.offtimed.10
toggledelayed	#delaytime	Toggles output delayed between on/off, parameters: • #delaytime (1-65535 seconds). Example: ebs.0.out3.toggledelayed.10
toggletimed	#runtime	Toggles output temporarily between on/off, parameters: • #runtime (1-65535 seconds). Example: ebs.0.out3.toggletimed.10
lock		Locks output, preventing any commands to this output except 'unlock', example: ebs.0.out2.lock
locktimed	#runtime	Temporarily locks output, parameters: • #runtime (1-65535 seconds). Example: ebs.0.out3.locktimed.10
timeadd	#runtime	Extends the current running time, parameters: • #runtime (1-65535 seconds). Example: ebs.0.out3.timeadd.10

		timeset	#runtime	Sets the current running time, parameters: • #runtime (1-65535 seconds). Example: ebs.0.out3.timeset.10
		timeabort		Terminates the current running time, example: ebs.0.out2.timeabort
	out6-out7	timerondelayed	#delaytime	Only applicable to the 16A output module. Turns timer on after initial delay, this is reported as a fb%.timingon event • #delaytime (1-65535 1/10 seconds). Example: ebs.0.out7.timerondelayed.100
		timeroffdelayed	#delaytime	Only applicable to the 16A output module. Turns timer off after initial delay, this is reported as a fb%.timingoff event • #delaytime (1-65535 1/10 seconds). Example: ebs.0.out7.timeroffdelayed.100
		timerontimed	#runtime	Only applicable to the 16A output module. Turns timer on for a given time and then back off, this is reported as a fb%.timingon or fb%.timingoff event respectively • #runtime (1-65535 1/10 seconds). Example: ebs.0.out7.timerontimed.100
		timerabort		Only applicable to the 16A output module. Stops the timer but leaves the on/off state, this is reported as a fb%.timingabort event, example: ebs.0.out7.timerabort
	fb0-fb5	on		Simulates 'output on' feedback, returns 0, example: ebs.0.fb2.on
		off		Simulates 'output off' feedback, returns 0, example: ebs.0.fb2.off
	fb6-fb7	timeron		Simulates 'timer on' feedback, returns 0, example: ebs.0.fb7.timeron
		timerabort		Simulates 'timer abort' feedback, returns 0, example: ebs.0.fb7.timerabort
		timeroff		Simulates 'timer off' feedback, returns 0, example: ebs.0.fb7.timeroff
ebr.0 - ebr.31				These commands are applicable to following modules: • Recessed installation shutter box Use module name and address to return the current status, example: ebr.0 Due to it's design this module only returns the status of the inputs as a bitstring where: • bit24-31 = in8-in15, where a '1' indicates a closed

				input
	in8-in15	ingt0		Simulates 'IN > 0 sec' event, returns 0, example: ebr.0.in8.ingt0
		outlt1		Simulates 'OUT < 1 sec' event, returns 0, example: ebr.0.in8.outlt1
		ingt1		Simulates 'IN > 1 sec' event, returns 0, example: ebr.0.in8.ingt1
		outgt1		Simulates 'OUT > 1 sec' event, returns 0, example: ebr.0.in8.outgt1
		ingt2		Simulates 'IN > 2 sec' event, returns 0, example: ebr.0.in8.ingt2
		out		Simulates ‘OUT’ event, returns 0, example: ebr.0.in8.out
	out0-out3	stop	#priolevel	Stops either up or down operation, parameters: • #priolevel (0-5) Example: ebr.1.out3.stop.1
		toggleup	#priolevel #priolock #runtime	Toggles between stop and up operation, parameters: • #priolevel (0-5) • #priolock (0-1) • #runtime (1-65535) expressed in 1/10 seconds Example: ebr.0.out1.toggleup.1.1.25
		toggledown	#priolevel #priolock #runtime	Toggles between stop and down operation, parameters: • #priolevel (0-5) • #priolock (0-1) • #runtime (1-65535) expressed in 1/10 seconds Example: ebr.0.out1.toggledown.1.1.25
		up	#priolevel #priolock #runtime	Starts up operation, parameters: • #priolevel (0-5) • #priolock (0-1) • #runtime (1-65535) expressed in 1/10 seconds Example: ebr.0.out1.up.1.1.25
		down	#priolevel #priolock #runtime	Starts down operation, parameters: • #priolevel (0-5) • #priolock (0-1) • #runtime (1-65535) expressed in 1/10 seconds Example: ebr.0.out1.down.1.1.25
		tipup	#priolevel #priolock #tiptime	Starts tipup operation, parameters: • #priolevel (0-5) • #priolock (0-1)

		- #tiptime (1-65535) expressed in 1/10 seconds Example: ebr.0.out1.tipup.1.1.5
tipdown	#priolevel #priolock #tiptime	Starts tipdown operation, parameters: - #priolevel (0-5) - #priolock (0-1) - #tiptime (1-65535) expressed in 1/10 seconds Example: ebr.0.out1.tipdown.1.1.5
priolock	#priomask	Locks one or more priority levels, parameters: - #priomask (0x00-0x3F), a bitwise combination of priority levels 0 upto 5, where level 0=1, level 1=2, level 2=4, level 3=8, level 4=16 and level 5=32. Example: ebr.0.out1.priolock.0x1C
priounlock	#priomask	Unlocks one or more priority levels, parameters: - #priomask (0x00-0x3F), a bitwise combination of priority levels 0 upto 5, where level 0=1, level 1=2, level 2=4, level 3=8, level 4=16 and level 5=32. Example: ebr.0.out1.priounlock.0x1C
learnon		Turns learning mode on, example: ebr.0.out1.learnon
learnoff		Turns learning mode off, example: ebr.0.out1.learnoff
prioset	#priomask	Sets one or more priority levels, parameters: - #priomask (0x00-0x3F), a bitwise combination of priority levels 0 upto 5, where level 0=1, level 1=2, level 2=4, level 3=8, level 4=16 and level 5=32. Example: ebr.0.out1.prioset.0x12
prioreset	#priomask	Resets one or more priority levels, parameters: - #priomask (0x00-0x3F), a bitwise combination of priority levels 0 upto 5, where level 0=1, level 1=2, level 2=4, level 3=8, level 4=16 and level 5=32. Example: ebr.0.out1.prioreset.0x12
delayedup	#priolevel #priolock #delaytime #runtime	Starts up operation after optional delay, parameters: - #priolevel (0-5) - #priolock (0-1) - #delaytime (1-65535) expressed in 1/10 seconds - #runtime (1-65535) expressed in 1/10 seconds Example: ebr.0.out1.delayedup.1.1.5.25
tipdelayedup	#priolevel #priolock	Starts up operation, parameters:

			#delaytime #runtime #tiptime	• #priolevel (0-5) • #priolock (0-1) • #delaytime (1-65535) expressed in 1/10 seconds • #runtime (1-65535) expressed in 1/10 seconds • #tiptime (1-65535) expressed in 1/10 seconds Example: ebr.0.out1.tipdelayedup.1.1.5.25.1
		delayeddown	#priolevel #priolock #delaytime #runtime	Starts down operation after optional delay, parameters: • #priolevel (0-5) • #priolock (0-1) • #delaytime (1-65535) expressed in 1/10 seconds • #runtime (1-65535) expressed in 1/10 seconds Example: ebr.0.out1.delayeddown.1.1.5.25
		tipdelayeddown	#priolevel #priolock #delaytime #runtime #tiptime	Starts down operation, parameters: • #priolevel (0-5) • #priolock (0-1) • #delaytime (1-65535) expressed in 1/10 seconds • #runtime (1-65535) expressed in 1/10 seconds • #tiptime (1-65535) expressed in 1/10 seconds Example: ebr.0.out1.tipdelayeddown.1.1.5.25.1
	out4-out7	timerondelayed	#delaytime	Turns timer on after initial delay, this is reported as a fb%.timingon event • #delaytime (1-65535 1/10 seconds). Example: ebr.0.out7.timerondelayed.100
		timeroffdelayed	#delaytime	Turns timer off after initial delay, this is reported as a fb%.timingoff event • #delaytime (1-65535 1/10 seconds). Example: ebr.0.out7.timeroffdelayed.100
		timerontimed	#runtime	Turns timer on for a given time and then back off, this is reported as a fb%.timingon or fb%.timingoff event respectively • #runtime (1-65535 1/10 seconds). Example: ebr.0.out7.timerontimed.100
		timerabort		Stops the timer but leaves the on/off state, this is reported as a fb%.timingabort event, example: ebr.0.out7.timerabort
	fb0-fb3	upon		Simulates feedback of a channel starting up operation, can be used in PHC programming to trigger an action, returns 0, example: ebr.0.fb0.upon
		downon		Simulates feedback of a channel starting down operation, can

				be used in PHC programming to trigger an action, returns 0, example: ebr.0.fb0.downon
		upoff		Simulates feedback of a channel stopping up operation, can be used in PHC programming to trigger an action, returns 0, example: ebr.0.fb0.upoff
		downoff		Simulates feedback of a channel stopping down operation, can be used in PHC programming to trigger an action, returns 0, example: ebr.0.fb0.downoff
	fb4-fb7	timeron		Simulates feedback for a timer output being switched on, returns 0, example: ebr.0.fb7.timeron
		timerabort		Simulates feedback for a timer output timer being aborted, returns 0, example: ebr.0.fb7.timerabort
		timeroff		Simulates feedback for a timer output being switched off, returns 0, example: ebr.0.fb7.timeroff
ebd.0 - ebd.31				These commands are applicable to following modules: • Recessed installation dimmer box (1-10V) Use module name and address to return the current status, example: ebd.0 If successful, the current status of the inputs and outputs is returned as a bitstring, where: • bit0-3 = out0-out3, where a '1' indicates an 'on' output • bit24-31 = in8-in15, where a '1' indicates a closed input
	in8-in15	ingt0		Simulates 'IN > 0 sec' event, returns 0, example: ebd.0.in8.ingt0
		outlt1		Simulates 'OUT < 1 sec' event, returns 0, example: ebd.0.in8.outlt1
		ingt1		Simulates 'IN > 1 sec' event, returns 0, example: ebd.0.in8.ingt1
		outgt1		Simulates 'OUT > 1 sec' event, returns 0, example: ebd.0.in8.outgt1
		ingt2		Simulates 'IN > 2 sec' event, returns 0, example: ebd.0.in8.ingt2
		out		Simulates ‘OUT’ event, returns 0, example: ebd.0.in8.out
	out0-out3			Returns current status (0-255), example: ebd.0.out0
		onmaxmem		Turns dimmer on at maximum level and store in memory, example: ebd.0.out0.maxonmem
		onmax		Turns dimmer on at maximum level, example: ebd.0.out0.maxon
		off		Turns dimmer off, example: ebd.0.out0.off
		togglemaxmem		Toggle dimmer between off and max, store in memory, example: ebd.0.out0.memtoggle
		togglemax		Toggles dimmer between off and max, example: ebd.0.out0.toggle

reverse	#runtime	Reverses dimming direction, parameters: • #runtime (1-160 seconds) to dim from off to max level, if omitted then 5 will be used. Example: ebd.0.out0.reverse.20
up	#runtime	Dim up from current till maximum level, parameters: • #runtime (1-160 seconds) to dim from off to max level, if omitted then 5 will be used. Example: ebd.0.out0.up.10
down	#runtime	Dim down from current level till off, parameters: • #runtime (1-160 seconds) to dim from max level to off, if omitted then 5 will be used. Example: ebd.0.out0.down.10
stopmem		Stops dimming and store current level in memory , example: ebd.0.out0.stopmem
togglemem		Toggles dimmer between off and memory level, example: ebd.0.out0.togglemem
onmem		Turns dimmer on using memory level, example: ebd.0.out0.memon
stopdia1		Stops dimming and store current level in DIA1, example: ebd.0.out0.stopdia1
toggledia1		Toggles dimmer between off and DIA1 level, example: ebd.0.out0.toggledia1
ondia1		Turns dimmer on using DIA1 value, example: ebd.0.out0.ondia1
stopdia2		Stops dimming and store current level in DIA2, example: ebd.0.out0.stopdia2
toggledia2		Toggles dimmer between off and DIA2 level, example: ebd.0.out0.toggledia2
ondia2		Turns dimmer on using DIA2 level, example: ebd.0.out0.ondia2
stopdia3		Stops dimming and store current level in DIA3, example: ebd.0.out0.stopdia3
toggledia3		Toggles dimmer between off and DIA3 level, example: ebd.0.out0.toggledia3
ondia3		Turns dimmer on using DIA3 level, example: ebd.0.out0.ondia3
set	#level #runtime	Starts dimming to level, parameters: • #level (0-255), if omitted then 128 will be used. • #runtime (1-160 seconds), if omitted then 5 will be used. Example: ebd.0.out0.set.99.5
ondelayed	#delay	Starts dimming to level, parameters:

			#level #runtime	• #delay (1-65535) • #level (0-255), if omitted then 128 will be used. • #runtime (1-160 seconds), if omitted then 5 will be used. Example: ebd.0.out0.setdelayed.100.99.5
		timeabort		Stops timer, example: ebd.0.out0.timeabort
		offdelayed	#delay	Turns dimmer of after delay, parameters: • #delay (1-65535) seconds Example: ebd.0.out0.offdelayed.10
	fb0-fb3	on		Dimmer was turned on, can be used in PHC programming to trigger an action, returns 0, example: ebd.0.fb0.on
		off		Dimmer was turned off, can be used in PHC programming to trigger an action, returns 0, example: ebd.0.fb0.off
		dimmed		Dim level reached, returns 0, example: ebd.0.fb0.dimmed
		dimmedup		Dim up level reached, returns 0, example: ebd.0.fb0.dimmedup
		dimmeddown		Dim down level reached, returns 0, example: ebd.0.fb0.dimmeddown
dim.0 - dim.31				These commands are applicable to following modules: • Phasenanschnittdimmer • Phasenabschnittdimmer • Synchronisierter Phasenanschnittdimmer • Synchronisierter Phasenabschnittdimmer • Universal dimmer Use module name and address to return the current status, example: dim.0 If successful, the current status of the outputs is returned as a bitstring, where: • bit0-1 = out0-out1, on/off status of channel • bit16-23 = level0, 0-255 level of channel • bit24-31 = level1, 0-255 level of channel
	out0-out1			Returns current status (0-255), example: dim.0.out0
		onmaxmem		Turns dimmer on at maximum level and store in memory, example: dim.0.out0.maxonmem
		onmax		Turns dimmer on at maximum level, example: dim.0.out0.maxon
		off		Turns dimmer off, example: dim.0.out0.off
		togglemaxmem		Toggle dimmer between off and memory level, example: dim.0.out0.memtoggle
		togglemax		Toggles dimmer between off and maximum level, example:

		dim.0.out0.toggle
reverse	#runtime	Reverses dimming direction, parameters: • #runtime (1-160 seconds) to dim from off to max level Example: dim.0.out0.reverse.20
up	#runtime	Dims up from current till maximum level, parameters: • #runtime (1-160 seconds) to dim from off to max level Example: dim.0.out0.up.10
down	#runtime	Dims down from current level till off, parameters: • #runtime (1-160 seconds) to dim from max level to off Example: dim.0.out0.down.10
stopmem		Stops dimming and store current level in memory, example: dim.0.out0.stopmem
togglemem		Toggles dimmer between off and memory level, example: dim.0.out0.togglemem
onmem		Turns dimmer on using memory level, example: dim.0.out0.memon
stopdia1		Stops dimming and store current level in DIA1, example: dim.0.out0.stopdia1
toggledia1		Toggles dimmer between off and DIA1 level, example: dim.0.out0.toggledia1
ondia1		Turns dimmer on using DIA1 value, example: dim.0.out0.ondia1
stopdia2		Stops dimming and store current level in DIA2, example: dim.0.out0.stopdia2
toggledia2		Toggles dimmer between off and DIA2 level, example: dim.0.out0.toggledia2
ondia2		Turns dimmer on using DIA2 level, example: dim.0.out0.ondia2
stopdia3		Stops dimming and store current level in DIA3, example: dim.0.out0.stopdia3
toggledia3		Toggles dimmer between off and DIA3 level, example: dim.0.out0.toggledia3
ondia3		Turns dimmer on using DIA3 level, example: dim.0.out0.ondia3
set	#level #runtime	Starts dimming to level, parameters: • #level (0-255), if omitted then 128 will be used. • #runtime (1-160 seconds), if omitted then 5 will be used. Example: dim.0.out0.set.99.5

		ondelayed	#delay #level #runtime	Starts dimming to level, only for universal dimmer module, parameters: • #delay (1-65535) • #level (0-255), if omitted then 128 will be used. • #runtime (1-160 seconds), if omitted then 5 will be used. Example: dim.0.out0.setdelayed.100.99.5
		timeabort		Stops delay timer, only for universal dimmer module, example: dim.0.out0.timeabort
		offdelayed	#delay	Turns dimmer of after delay, only for universal dimmer module, parameters: • #delay (1-65535) seconds Example: dim.0.out0.offdelayed.10
	fb0-fb1	on		Dimmer was turned on, returns 0, example: dim.0.fb0.on
		off		Dimmer was turned off, returns 0, example: dim.0.fb0.off
		dimmed		Dim level reached, only for universal dimmer module, returns 0, example: dim.0.fb0.dimmed
		dimmedup		Dim up level reached, only for universal dimmer module, returns 0, example: dim.0.fb0.dimmedup
		dimmeddown		Dim down level reached, only for universal dimmer module, returns 0, example: dim.0.fb0.dimmeddown

3.5.Structured Tags

xiControl extends the set of tags supported by xWRC, thereby communicating with your connected PHC system as needed.

3.5.1.Tag branches

Below table lists all branches and nodes supported by xiControl, with a detailed description:

Branch	Node	Predef	Description
plugin icontrol			This branch has 2 names that are transparant, ‘plugin’ which is deprecated but provided for backward compatibility, ‘icontrol’ to be more logical and support future extensions of xWRC.
	alias		Plugin alias, all lower case, example: <?plugin alias?> returns 'icontrol'
	author		Plugin author, example: <?plugin author?>
	created		Plugin build date. Example: <?plugin created?> returns 'Apr 22 2013'
	description		Plugin description. Example: <?plugin description?> returns 'Webinterface for PHC'
	href		Plugin menu link. Example: <?plugin href?> returns '/icontrol.dll?'
	img		Plugin menu image. Example: <?plugin img?> returns 'src='/icontrol/img/icontrol.gif"
	lmode	demo licensed	The licensing mode in which the PHC protocol stack is running, 0=demo, 1=licensed. Example: <?plugin lmode?> returns '1' Example: <?plugin lmode demo="demo" licensed="licensed"?> returns 'licensed'
	name		Name of the plugin. Example: <?plugin name?> returns 'xiControl'
	version		Version of the plugin in format 'x.y.z.t'. Example: <?plugin version?> returns '3.0.3.0'
request	ccmd	empty	The contents of all concatinated compound-cmds of a request, if no compound-cmd was given and the 'empty' argument is specified, then this value will be used instead. Example: <?request ccmd?> Example: <?request ccmd empty="ping"?>

phc

This branch uses a different syntax:

```
tag        = <?phc expression *[';' expression]?>

expression = (node | #value) *[oper (node | #value)]
node       = see below
oper       = '+' | '-' | '*'  | '/' | '%' | '&' |
             '^' | '|' | '>>' | '<<'
#value     = numerical value
```

What will be the result of replacing above tag?

In step 1, expression will be evaluated by collecting the actual value of node or #value.

- If this is a string then no [oper (node | #value)] can occur
- If this is a number then every [oper (node | #value)] occurence is calculated from left to right.
- The final result is converted into a decimal string

In step 2, the above is repeated for every expression, and finally all resulting strings are concatinated.

	date		Returns the system date of the connected STM. Example: <?phc date?>
	linkstate	off on	Returns the status of link to the remote STM module, there will be 2 predefined values possible: 0=off, 1=on. Examples: <?phc linkstate?> returns '1' <?phc linkstate on="on.gif" off="off.gif"?> returns 'on.gif'
	remmode	bin xmlrpc	Remote mode of STM, 0=bin (STMv1/v2), 1=xmlrpc (STMv3). Example: <?phc remmode?> returns '1'
	remaddr		Remote IP address via which to reach the connected STM module over a TCP connection. Example: <?phc remaddr?> returns '127.0.0.1'
	remport		Remote IP port via which to reach the conencted STM module over a TCP connection. Example: <?phc remport?> returns '10000'
	stmaddr		The address of the STM (0-7) to which xiControl is connected. Example: <?phc stmaddr?> returns '0'
	time		Returns the system time of the connected STM. Example: <?phc time?>
	version		Returns the firmware version of the connected STM. Example: <?phc version?>

imd.0-31 tab.0-31 et0.0-31 et1.0-31		Returns the actual status of all outputs and inputs as a decimal value, where: • bit0-15 = led0-led15, 0=off, 1=on • bit16-31 = in0-in15, 0=open, 1=closed Note: some modules may have less inputs or LED outputs. Examples: <?phc imd.0?> returns the status of all led% and in%, i.e. '5144' <?phc imd.0 / 256 & 0x01?> returns the status of led8, i.e. '0' or '1' <?phc imd.0 / 65536 & 0x03?> returns the status of in0 and in1, i.e. '0'...'3'.
imd.%.led0-15 imd.%.in0-15 tab.%.led0-15 tab.%.in0-15 et0.%.led0-8 et0.%.in0-7 et1.%.led0-8 et1.%.in0-7	off on	Returns the actual status of ‘in%’ or 'led%', there will be 2 predefined values possible: 0=off (open), 1=on (closed). Examples: <?phc imd.0.led1?> will return '0' or '1' <?phc imd.0.led1 * 4?> will return '0' or '4' <?phc imd.0.led1;imd.0.led7;imd.0.in3?> might return '101' <?phc imd.1.led2 on="ledon.gif" off="ledoff.gif"?>
imw.0-31		Returns the actual status of all outputs and inputs on 'imw.%' as a decimal value, where: • bit0-3 = mrk0-mrk3, 0=off, 1=on • bit4-7 = led4-led7, 0=off, 1=on • bit16-23 = in0-in7, 0=open, 1=closed Examples: <?phc imw.0?> returns the status of all led% and in%, i.e. '5144' <?phc imw.0 / 256 & 0x01?> returns the status of led8, i.e. '0' or '1' <?phc imw.0 / 65536 & 0x03?> returns the status of in0 and in1, i.e. '0'...'3'.
imw.%.led4-7 imw.%.mrk0-3 imw.%.in0-7	off on	Returns the actual status of 'led%', 'mrk%' or 'in%', there will be 2 predefined values possible: 0=off (open), 1=on (closed). Examples: <?phc imw.0.led7?> will return '0' or '1' <?phc imw.0.led7 on="ledon.gif" off="ledoff.gif"?>
uim.0-31		Returns the actual status of all inputs as a decimal value, where: • bit23-27 = in7-in11, 0=open, 1=closed Example: <?phc uim.0?> returns status of all led%/in%, i.e. '5144'
uim.%.in7-11	off on	Returns the actual status of 'in%', there will be 2 predefined values possible: 0=off (open), 1=on (closed). Example: <?phc uim.0.in7?> returns '0' or '1'
utm.0-31		Returns the actual status of all outputs and inputs as a decimal value, where:

		• bit0-1 = led0-led1, 0=off, 1=on • bit16-19 = in0-in3, 0=open, 1=closed • bit24-27 = in8-in11, 0=open, 1=closed Example: <?phc utm.0?> returns status of all led%/in%, i.e. '5144'
utm.%.led0-1 utm.%.in0-3 utm.%.in8-11	off on	Returns the actual status of 'led%' or 'in%', there will be 2 predefined values possible: 0=off (open), 1=on (closed). Example: <?phc utm.0.led1?> will return '0' or '1'
bwm.0-31		Returns the actual status of all inputs as a decimal value, where: • bit16-17 = im0-im1, 0=still, 1=movement • bit19 = il3, 0=dark, 1=light • bit24-27 = in8-in11, 0=open, 1=closed Example: <?phc bwm.0?>
bwm.%.in8-11 bwm%.im0-1 bwm%.il3	off on	Returns the actual status of 'in%', 'im%' or 'il%', there will be 2 predefined values possible: 0=off (open), 1=on (closed). Example: <?phc bwm.0.in8?> returns the status of in8
omd.0-31 jrm.0-31		Returns the actual status of all outputs as a decimal value, where: • bit0-7 = out0-out7, 0=off, 1=on Note: jrm modules always return 0 as output status. Examples: <?phc omd.0?> will return the status of all out%, i.e. '255' <?phc omd.0 ^ 0x55?> will toggle all even bits before displaying the result. <?phc omd.0 / 4 & 0x03?> will retain the status of out2 and out3.
omd.%.out0-7 jrm.%.out0-3	off on	Returns the actual status of out%, there will be 2 predefined values possible: 0=off, 1=on. Examples: <?phc omd.0.out2?> <?phc omd.1.out1 on="on.gif" off="off.gif"?>
amd.0-31 dim.0-31		Returns the actual status of all outputs as a decimal value, where: • bit0-1 = out0-out1, 0=off, 1=on Examples: <?phc amd.0?> will return the status of all out%, i.e. '1' <?phc dim.0?> will return the status of all out%, i.e. '3'
amd.%.out0 dim.%.out0-1	off on	Returns current status of out% (0-255), there will be 2 predefined values possible: off (status = 0), on (status > 0). Alternatively convert status as follows: <?phc dim.0.out0+255/256?> will return 0 or 1

			Example: <?phc dim.0.out0?> may return '240'
	ebs.0-31		Returns the actual status of all output and inputs as a decimal value, where: • bit0-5 = out0-out5, 0=off, 1=on • bit24-31 = in8-in15, 0=open, 1=closed Example: <?phc ebs.0?>
	ebs.%.in8-15 ebs.%.out0-5	off on	Returns the actual status of 'in%' or ‘out’, there will be 2 predefined values possible: 0=off (open), 1=on (closed). Example: <?phc ebs.0.in8?> returns the status of in8
	ebr.0-31		Returns the actual status of all output and inputs as a decimal value, where: • bit0-7 = out0-out7, 0=off, 1=on • bit24-31 = in8-in15, 0=open, 1=closed Example: <?phc ebr.0?>
	ebr.%.in8-15 ebr.%.out0-3	off on	Returns the actual status of 'in%' or ‘out%’, there will be 2 predefined values possible: 0=off (open), 1=on (closed). Example: <?phc ebr.0.in8?> returns the status of in8
	ebd.0-31		Returns the actual status of all output and inputs as a decimal value, where: • bit0-3 = out0-out3, 0=off, 1=on • bit24-31 = in8-in15, 0=open, 1=closed Example: <?phc ebd.0?>
	ebd.%.in8-15	off on	Returns the actual status of 'in%', there will be 2 predefined values possible: 0=off (open), 1=on (closed). Example: <?phc ebd.0.in8?> returns the status of in8
	ebd.%.out0-3	off on	Returns current status of 'out%' (0 to 255), there will be 2 predefined values possible: off (status = 0), on (status > 0). Alternatively convert status as follows: <?phc ebd.0.out0+255/256?> will return 0 or 1 Example:<?phc ebd.0.out0?> may return '224'

3.6.Smartphone Integration

NetIO is a commercial app for your smartphone that provides a visual layout and links it to xiControl commands.

Setting up control of your PHC system can be done via a webbased layout tool that saves it’s info online, then by logging onto that server and synching your project to your smartphone, you can in matter of minutes grab control of your PHC system.

3.6.1.Create a new project

To be specified.

3.6.2.Adding a switch control

To be specified.

3.6.3.Adding a text label

To be specified.