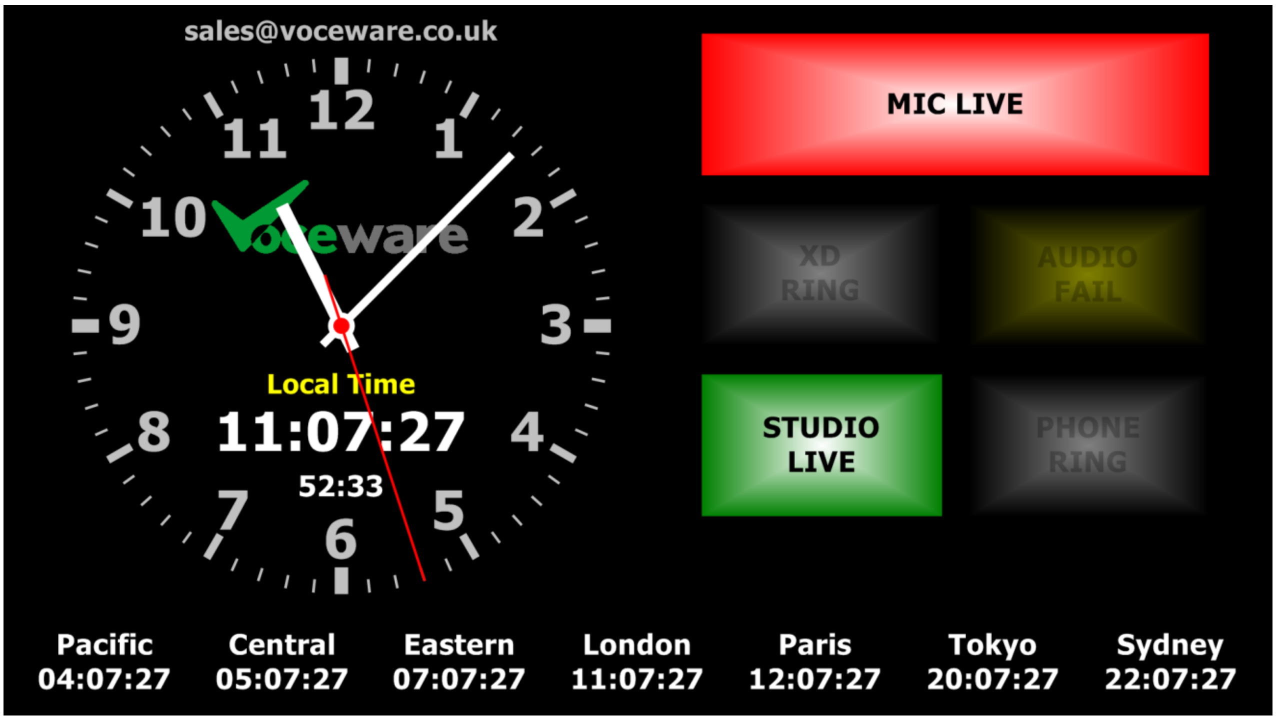
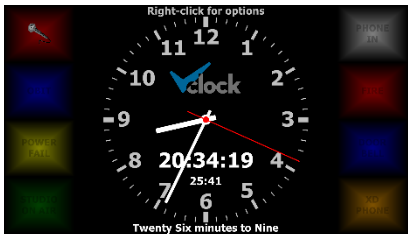
Installation and Reference Manual

V5.1.8





Contents

Overview

VClock is a highly configurable clock primarily for use in a studio environment. It can show the time in several formats (English, Digital, Analogue), with or without the date, it can back-time to several junctions an hour, its colours can be configured to match your décor, and it can include your stations logo in the center of the clock.

It also includes up to 32 "lamps" which can be triggered externally in many ways (RS232, IP, GPI, http posts and integration with many other systems). They can be set to turn on, off or flash, the text and colour can be changed, and they can each have a separate logo (a picture of a mic for example). They can be positioned automatically beside or above/below the clock depending on your screen's orientation, or manually positioned for more control.

The application can be set to launch in a certain position on a screen, and can be pointed at another "Master" clock to mimic its commands or share its GPI cards.

VClock has an integrated VLC player to play video, and integrated browser to show web pages (can be full screen as studio signage), and can integrate with Google and Office365 calendars to show bookings for studios or meeting rooms.

VClock can also control other systems to make audio routes, give out GPO, control DMX lighting, and more.

Features

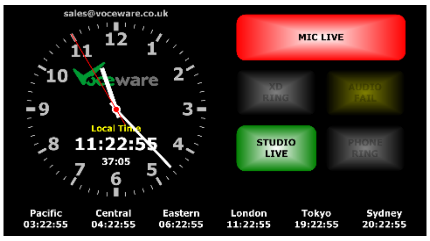
- Analogue clock face with ability to
  - Change background colour (outer and inner colours to give gradient effect)
  - Change colour of numbers, hands, second hand (and font of numbers)
  - Display day of week and / or date of month and set the TimeZone
  - Display a name (colour / font configurable) or logo (size configurable) in the center of the clock face
  - Display count-up/down stopwatches/programme timers started by a trigger such as mic live
- Digital clock
  - Colour, font and TimeZone configurable
  - Backtiming with ability to backtime to several junctions per hour or to specific times of day
  - Can be as well as or instead of Analogue clock face
- "Language" clock with ability to
  - Choose between several internal formats
  - Can show a digital clock (and/or date) as an alternative to inside the clock
  - Choose a "Custom" format and specify in a text file (German example shipped – see appendix B)
  - Choose from several different languages (French, German, Italian, Spanish, Dutch, Danish)
  - Colour, font and TimeZone configurable
- Top / Middle / Bottom Captions
  - Colour / font configurable for each
  - Display a static message or rotate several messages to help the presenter with websites, phone numbers, etc.
  - Messages can contain "VClock Variables" to show day of week, date, time, etc.
  - "VClock Variables" are available to show the "Language" Clock messages on these captions, and they can be rotated with other messages, and have a separate TimeZone
- 32 x lamps
  - Colour / font / position / shape and TimeZone for any time-based "VClock Variables" configurable
  - 14 states – On / Off / OnOff / OnOffSolid / OnOffFast / OnOffFastSolid / OnDisabled / OnDisabledSolid / Phone / PhoneSolid / Flash / Solid / SolidDim / Disabled
  - Each lamp can have a logo or the caption can be user defined
  - Lamps can surround the clock, to one side / top / bottom or manually positioned.
- 3 x Embedded VLC Video Players
  - Size / Position configurable
- Audio file playback for alarms such as transmitter fail, etc
- A simple Programming Language allows all configuration options to be changed when a trigger (such as a GPI contact closure) is received. So for example:
  - Flash a "Mic Live" lamp, changing the caption to bright white when the "mic live" GPI is received from a Studio mixer. Dim the lamp and change the text to black when the mic is not live. Some basic AND/OR logic can also be applied to the GPIs (for example sound an alert when tx fails, but mute whenever the mic is live).
  - Display a count-up timer as part of the Caption, started by a trigger such as turning the lamp on (for example a Mic Live lamp can have a talk-time counter).
- Support of the popular Advantech range of GPI cards, as well as a standard windows games controller ports. Advantech card outputs can also be turned on/off by VClock, and can read analogue values (voltages) as well as GPIs. XKeys USB Interfaces are also supported, and the Barix Barionet 50 network device.
- Support of Serial or TCP IP / UDP ports to send and receive commands, It can receive SNMP traps, and an embedded WebServer allows other applications to do http posts or gets to send VClock commands. It can also take MIDI messages as triggers, and give an RSS feed of lamp states.
- Native support of popular equipment such as AEQ (Capitol and Forum), Audionics (emm88), Axia, DHD (ECP and Ember+), Lawo (RAS and Ember+), PhoneBOX (v3 or above), Wheatstone. Also the Blackmagic Smart Videohub range of video routers.
- Time Of Day events to trigger salvos (at a time on selected days, repeat every x seconds until another time).
- An "instant alarm" (mainly for me boiling eggs for the kids to be honest as we have a laptop in the kitchen but no egg timer!!).
- Use the clock in a master/slave mode to allow a master clock (for example with GPI card) to control the lamps on other slave clocks via IP or RS232
- Run several clocks on the same machine (on dual or quad video card), each able to act independently on triggers.
- Embedded WebBrowser allows you to display web pages (twitter, facebook, etc) within the clock – which scales to the clock size.
- Ability to offset the entire clock to take into account delay units such as the Telos PDM or CloudCast BDS.
- DMX control of lighting using Art-Net or ESP protocols
- Control (and tally) of the Blackmagic Smart Videohub range of video routers.
- Integration with Google and Office365 calendars to show room availability.
- Ability to convert text to speech ("Speaking Clock")

Version Comparison Grid

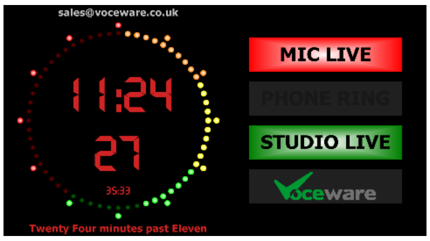
VClock comes in a Lite, Standard and Plus version, each offering extra functionality beyond the last. Here is a list of features that each version supports:

	VClock Lite	**VClock Standard**	**VClock Plus**
Fully configurable Analogue / Digital Clock Face	✓	✓	✓
Receive commands from most external interfaces / GPIs (exceptions are below)	✓	✓	✓
Display / Control of up to 32 Lamps		✓	✓
Embedded Web Browser		✓	✓
Embedded VLC Players			✓
Send commands to external equipment / GPOs / HTTP Get / launch external apps			✓
Write entries to CSV and text files			✓
Lamps can act as Buttons			✓
DMX Lighting Control			✓
Email sending			✓
Office 365 and Google Calander Integration			✓
Integration with PDM delay units			✓
Integration with StreamDeck panels			✓
Text to Speech ("Speaking Clock") functionality			✓

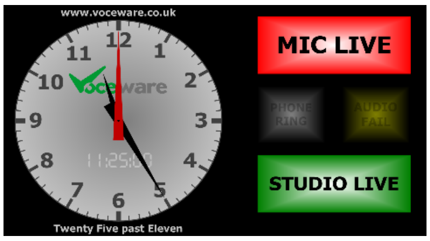
Some Example Layouts



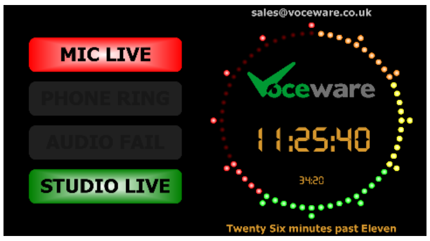
Different Timezones



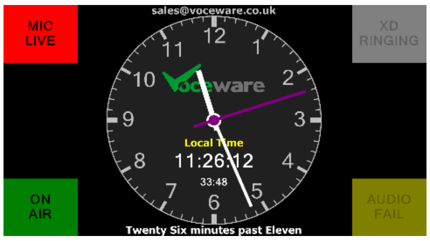
Large Digital Clock



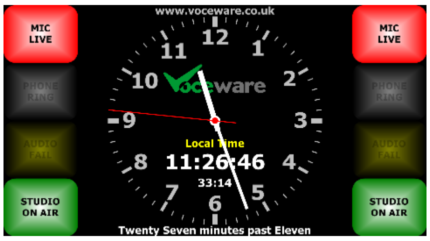
Different Hand Styles



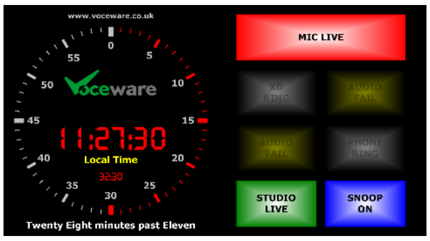
Smaller Digital Clock



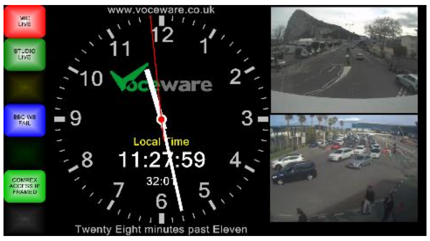
Flatter Lamp Style



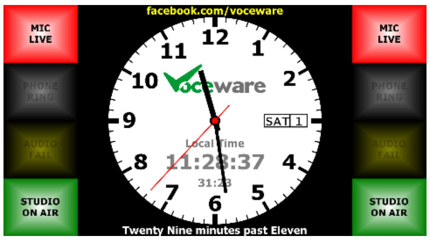
Rounded Lamp Style



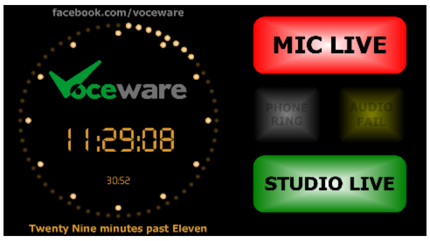
Coloured Ticks for Secs



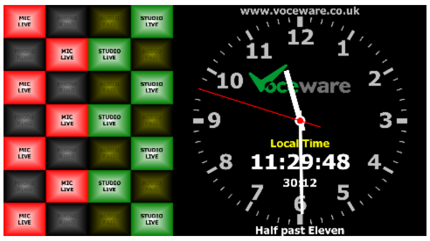
Embedded Web Browser



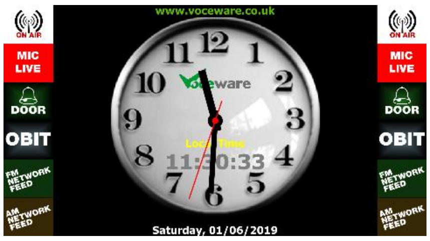
Day / Date Display



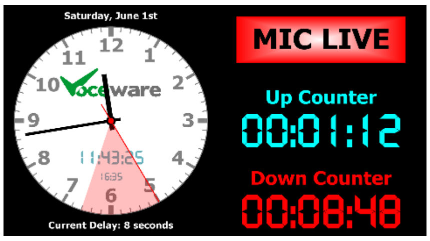
Lamps User Positioned



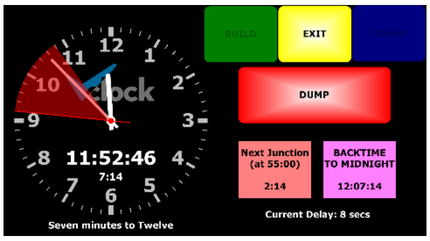
Up to 32 Lamps



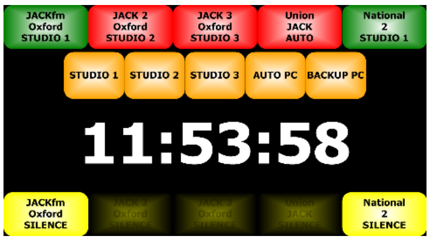
Use your own images



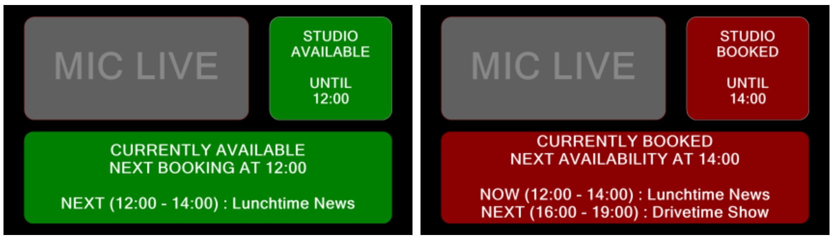
Up/Down Counters



PDM / Cloudcast control



Transmitter Switching


MIC LIVE

STUDIO AVAILABLE
UNTIL
12:00

CURRENTLY AVAILABLE
NEXT BOOKING AT 12:00

NEXT (12:00 - 14:00) : Lunchtime News

MIC LIVE

STUDIO BOOKED
UNTIL
14:00

CURRENTLY BOOKED
NEXT AVAILABILITY AT 14:00

NOW (12:00 - 14:00) : Lunchtime News
NEXT (16:00 - 19:00) : Drivetime Show

Studio Signage

.... and many more! The possibilities are endless.

System Requirements

VClock has been tested on XP (32 bit) up to version 4.37. This used the Microsoft .Net Framework 4 (Full Profile). Versions after this used Microsoft .Net v4.5.2, so are no longer compatible with Windows XP. Version 5.0 onwards use Microsoft .Net v4.7.2.

Windows 7 (32 and 64 bit), Windows 8 and Windows 10 are fully supported, however it should work on any Windows OS supporting Microsoft .Net Framework 4.7.2.

VClock requires Microsoft .Net Framework v4.7.2 to be installed on your system. It will not run without this.

VClock also has several other dependencies for parts of the program to work:

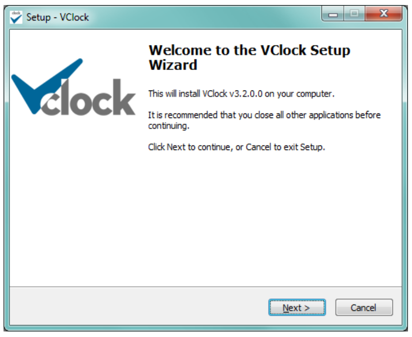
- DirectX Version 9.0 is required in order to support a games port for GPI use
- Advantech DAQNavi drivers are required in order to support an Advantech GPI card
- VLC is required to be installed on the machine, if you want to use the embedded VLC Players.

… however the application will function without any of these being present, the relevant functions will simply not be available.

The VClock Installation needs to be run as an Administrator. It will tell you if you are not, or prompt if "User Account Control" is enabled (windows Vista onwards). Normal operation of the clock does not need elevated permissions, but creating IP ports (for IP Server or HTTP Server) does.

Installation

Simply run the Installation file. Accepting the defaults should work in most cases. It will prompt you if "User Account Control" is enabled (Vista onwards), as it needs to run as an Administrator:


Setup - VClock

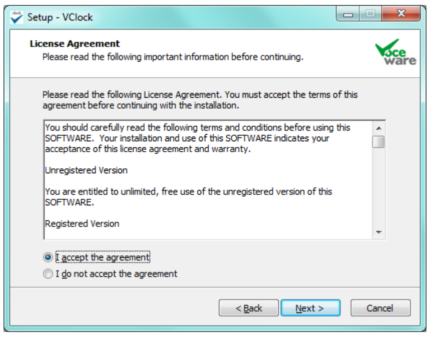
Welcome to the VClock Setup Wizard

This will install VClock v3.2.0.0 on your computer.

It is recommended that you close all other applications before continuing.

Click Next to continue, or Cancel to exit Setup.

Press Next…


Setup - VClock

License Agreement
Please read the following important information before continuing.

Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.

You should carefully read the following terms and conditions before using this SOFTWARE. Your installation and use of this SOFTWARE indicates your acceptance of this license agreement and warranty.

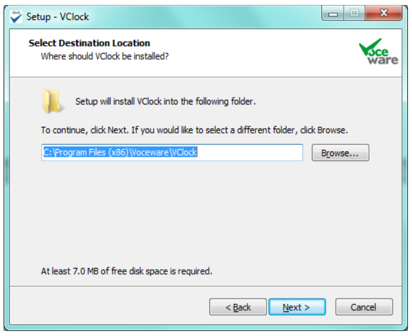
Unregistered Version

You are entitled to unlimited, free use of the unregistered version of this SOFTWARE.

Registered Version

[✓ I accept the agreement]
I do not accept the agreement

Accept the agreement and press Next…


Setup - VClock

Select Destination Location
Where should VClock be installed?

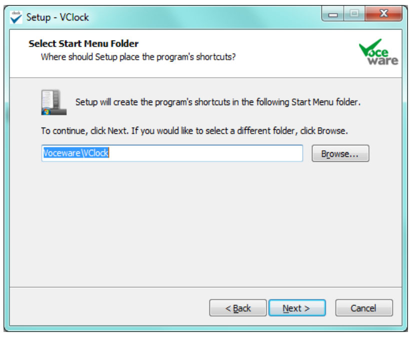
Setup will install VClock into the following folder.

To continue, click Next. If you would like to select a different folder, click Browse.

C:\Program Files (x86)\Voceware\VClock

At least 7.0 MB of free disk space is required.

Choose a different folder or press Next…


Setup - VClock

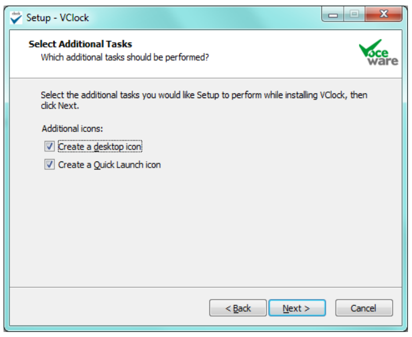
Select Start Menu Folder
Where should Setup place the program's shortcuts?

Setup will create the program's shortcuts in the following Start Menu folder.

To continue, click Next. If you would like to select a different folder, click Browse.

Voceware\VClock

Choose a different Menu, or press Next…


Setup - VClock

Select Additional Tasks

Which additional tasks should be performed?

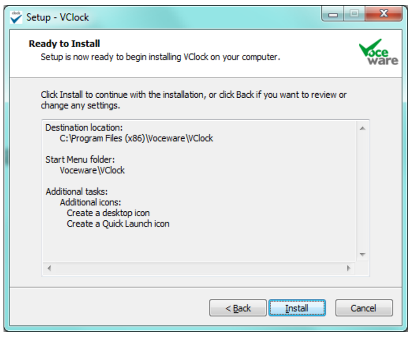
Select the additional tasks you would like Setup to perform while installing VClock, then click Next.

Additional icons:

☑ Create a desktop icon

☑ Create a Quick Launch icon

Deselect either option if you want to.. then Next..


Setup - VClock

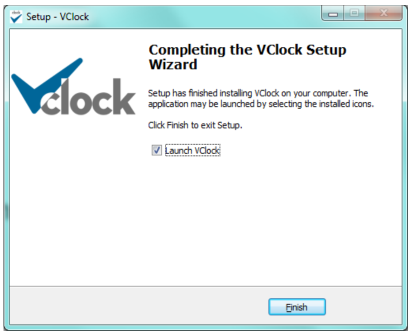
Ready to Install

Setup is now ready to begin installing VClock on your computer.

Click Install to continue with the installation, or click Back if you want to review or change any settings.

```
Destination location:
    C:\Program Files (x86)\Voceware\VClock

Start Menu folder:
    Voceware\VClock

Additional tasks:
    Additional icons:
        Create a desktop icon
        Create a Quick Launch icon
```

Press Install to complete the installation…


Setup - VClock

Completing the VClock Setup Wizard

Setup has finished installing VClock on your computer. The application may be launched by selecting the installed icons.

Click Finish to exit Setup.

☑ Launch VClock

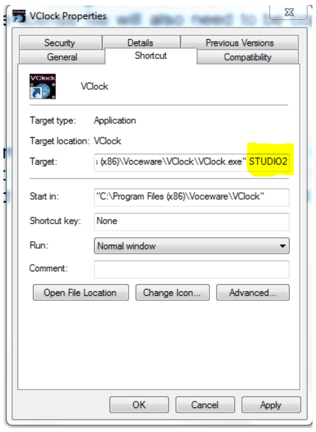
Press Finish and VClock will launch.

Configuring multiple clocks on a single PC

The VClock Setup program will create a shortcut to VClock on the start menu (by default under Programs\Voceware\VClock.

If you wish to run several clocks on a single PC, you will want them to have independent settings such as screen position, as well as separate salvos, defaults, etc.

This is achieved by making a copy of the shortcut to VClock, right clicking it to edit its properties and adding a unique commandline argument to the target. VClock will create a subfolder with this name, and the settings for that instance of the clock will be contained within this folder. Note that the license file will also need to be copied to this folder for this instance of the clock.



Target type: Application
Target location: VClock
Target: : (x86)\Voceware\VClock\VClock.exe" STUDIO2
Start in: "C:\Program Files (x86)\Voceware\VClock"
Shortcut key: None
Run: Normal window
Comment:

The license file specifies how many instances you are allowed to run concurrently (it does not specify how many different configurations you can have, only how many can run at the same time). If this is exceeded then **all** of the clocks licenses will become invalid until the extra clocks are closed.

Note that some features can only run on 1 clock per machine, such as opening the same IPServer port on 2 instances, or connecting to an Advantech GPIO card. If you want multiple clocks to act on these devices, you will need to network the clocks together using serial or IPClient/Server or Serial connections, then relay the commands from the "master" clock to the "slave".

Licensing

By default, with no license, the clock will function, but with the following limitations:

- External triggers (such as GPIs, serial data and IP commands) will only work for the first 10 commands received. After this, VClock will not trigger any events until the application is restarted. It will, however, still "relay" them to other slave clocks (see "relay mode").
- The clock will not allow a Logo to be added
- The Caption for the clock cannot be set, and will display an "UNLICENCED VERSION" message, along with how many "Grace Events" you have remaining.

Other than this, the clock will function completely, allowing you to ensure the look fits in with your environment, interfaces with your equipment, etc. If you need a fully functional, time limited trial license, please get in touch with us.

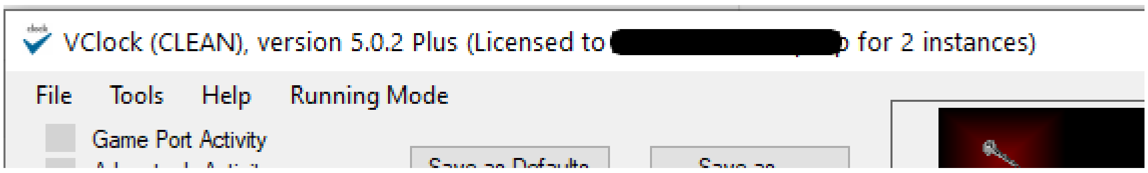
To license the clock, you need to supply us with your name, the hostname of the machine that the clock will be installed on, and the machine's "key". The license will be tied to that machine name and key.

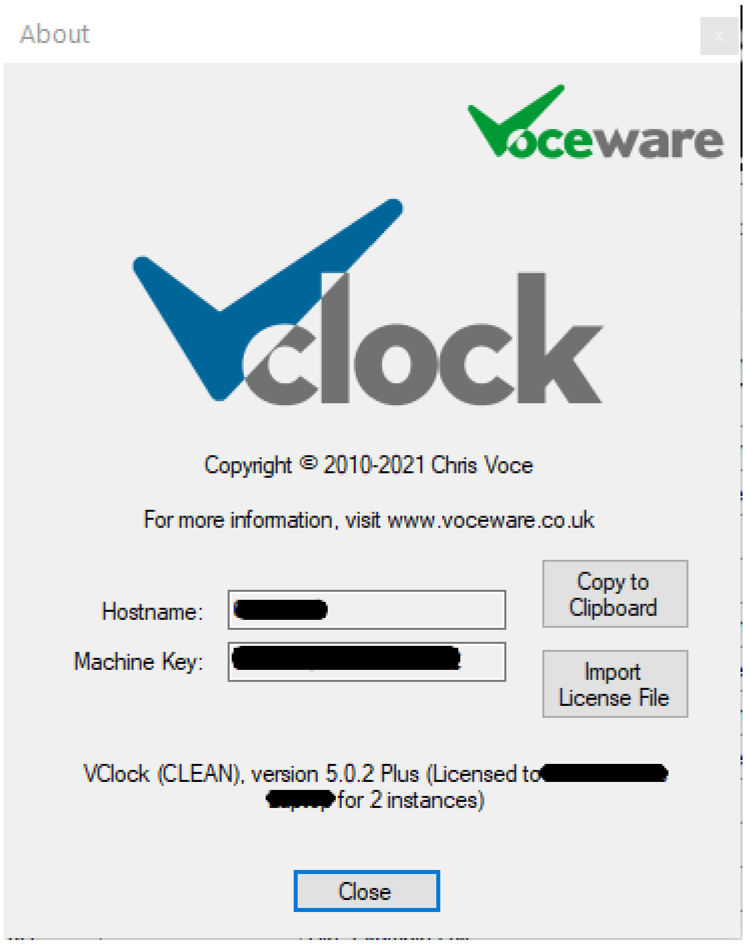
The hostname and machine key can be found on the Help/About screen. For ease, you can press the "Copy to Clipboard" button and paste these into an email when requesting your license. This page also gives you information about the version and the licensee (when licensed!).

When you receive the link to your licence file, download it to a folder then press the "Browse to Licence File" on the main config page of VClock, choose Tools / Import Licence File on the menus, or press "Import Licence File" on the help/about screen. All of these options simply copy the VCLOCK.LIC file to the C:\ProgramData\Voceware\VClock folder.

If you are running multiple VClocks on a single machine, you need to do this on each instance (or copy the VCLOCK.LIC file to each C:\ProgramData\Voceware\VClock\Instance subfolder).

The current licence status/details are shown in the title bar of the config screen, and on the help/about screen:


VClock (CLEAN), version 5.0.2 Plus (Licensed to [redacted] for 2 instances)



Copyright © 2010-2021 Chris Voce

For more information, visit www.voceware.co.uk

Hostname: [redacted]

Machine Key: [redacted]

VClock (CLEAN), version 5.0.2 Plus (Licensed to [redacted] for 2 instances)

There are 3 levels of license available.

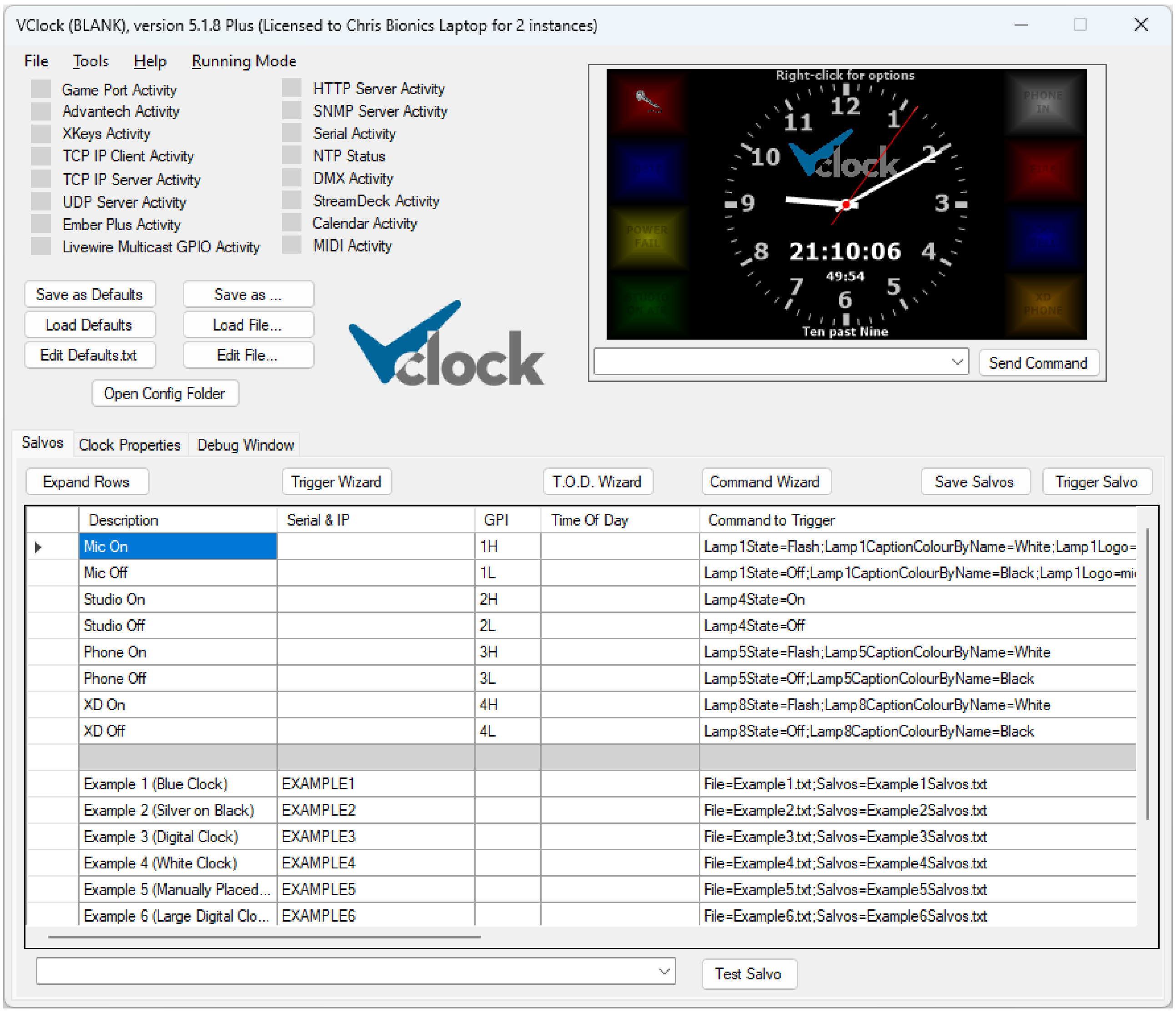
> “**VClock Lite**” disables the lamps and webbrowser entirely. Only the clock is available.
>
> “**VClock Standard**” enables the lamps and webbrowser. External triggers such as GPIs can change the state of the lamps.
>
> “**VClock Plus**” adds the ability for VClock to send commands as well as receive them. Commands can be sent to external equipment such as Wheatstone SLIOs and Axia VMixes / xNode crosspoints, via Advantech or XKeys GPO, Serial, IP Server and IP Client connections. It also enables the use of a StreamDeck button panel, embedded VLC players, DMX control, Google/Office365 calendar integration and the ability to send emails.

The simplest way to install the **VCLOCK.LIC file** is to save it from the email we send you to a known place (such as C:\ or the desktop), then use the “Browse to License File” button on the clock to select this file. This will copy the file to the program installation folder for you.

Another alternative, at a small extra cost, is a **Hardware Licensing Key**. The USB key will be pre-loaded with a number of instances and a license level. This can be transferred between machines easily as the key contains the license information. Please email sales@voceware.co.uk for further details.

Configuration

When you first launch VClock it will start in Running Mode. Right-click anywhere on the clock and choose "Exit Running Mode", to change to configuration mode:



Description	Serial & IP	GPI	Time Of Day	Command to Trigger
Mic On		1H		Lamp1State=Flash;Lamp1CaptionColourByName=White;Lamp1Logo=
Mic Off		1L		Lamp1State=Off;Lamp1CaptionColourByName=Black;Lamp1Logo=mi
Studio On		2H		Lamp4State=On
Studio Off		2L		Lamp4State=Off
Phone On		3H		Lamp5State=Flash;Lamp5CaptionColourByName=White
Phone Off		3L		Lamp5State=Off;Lamp5CaptionColourByName=Black
XD On		4H		Lamp8State=Flash;Lamp8CaptionColourByName=White
XD Off		4L		Lamp8State=Off;Lamp8CaptionColourByName=Black
Example 1 (Blue Clock)	EXAMPLE1			File=Example1.txt;Salvos=Example1Salvos.txt
Example 2 (Silver on Black)	EXAMPLE2			File=Example2.txt;Salvos=Example2Salvos.txt
Example 3 (Digital Clock)	EXAMPLE3			File=Example3.txt;Salvos=Example3Salvos.txt
Example 4 (White Clock)	EXAMPLE4			File=Example4.txt;Salvos=Example4Salvos.txt
Example 5 (Manually Placed...	EXAMPLE5			File=Example5.txt;Salvos=Example5Salvos.txt
Example 6 (Large Digital Clo...	EXAMPLE6			File=Example6.txt;Salvos=Example6Salvos.txt

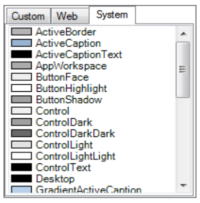
The application will export a file called "DEFAULTS.TXT" and will create some default "salvos" (stored in "SALVOS.TXT"). This sets the default look and feel of the clock, and creates some example events to turn lamps on and off.

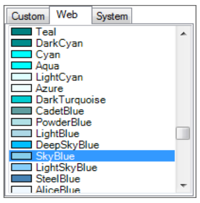
There are 3 main pages used for configuration, on tabs below the clock. **Salvos** allow you to set triggers for incoming data to change any property / properties of the clock or lamps. **Clock Properties** allows you to set all of the properties instantly, whilst **Debug Window** shows some useful incoming data (GPI states and IP / Serial strings recently received).

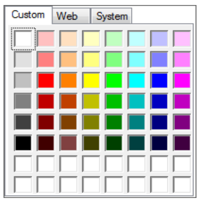
Resizing the app will expand the size of the tabbed area, to let you see as many salvos / properties as you can. The preview clock at the top is in the same aspect ratio as the main clock. Moving the mouse over it will make it double in size to be seen more easily. Clicking on a lamp / clock face / top or bottom caption will jump to that section in the property grid on the "Clock Properties" tab below.

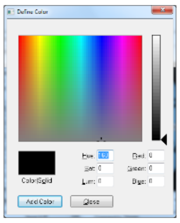
There are many different characteristics of the clock and lamps that can be changed (colours, text, etc). The "**Clock Properties**" tab allows you to change each of these properties instantly and see the results. This is by far the easiest way of getting the clock to look the way you want it to. You can sort alphabetically, or sort into sections (Clock Settings, Lamp 01, Lamp 02, etc) by using the icons at the top of the Property Panel. The Property Panel will only show settings relevant to your license (so no "lamp" settings for VClock Lite, for example).

Some properties are text strings – just type a new one ("\n" means new line). Some are logos (browse to an image file), and some are selectable from a pre-defined list of options (choose from the drop down list). Some are fonts (select from the dialogue that appears – but note that the size of any font is ignored as it is auto-calculated depending on the size of the clock face). Some are positions (select from the graphic that appears). Finally, some are colours. You can either select a predefined colour (from the System and Web tabs across the top of the selection box that appears), or you can be very specific and choose a custom colour… to do this select the Custom tab, then right-click one of the white boxes at the bottom… then select the exact colour and brightness you require:





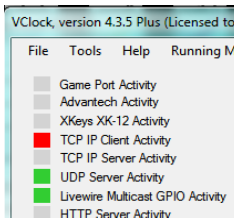
or


then right click a bottom box:


You can send these commands as a macro to the clock, a full list of which is in Appendix A. These macros are written into DEFAULTS.TXT to define how VClock looks when started. They are also written into the salvos to define what changes when Serial/IP commands or GPIs are received. They can also be written to any other text file, and a special macro can be used to trigger the contents of that file (File=filename.txt). Finally, the macro can be sent to the clock via the IP or Serial interface.

The clock ships with example configurations, saved to files Example1.txt Example2.txt etc. The default Salvos have lines that will load these (and associated example Salvo settings), or you can load them via the dropdown commands box. You can use these as a starting point, then save over the top of defaults.txt/salvos.txt. Example 1 is identical to the original defaults.txt created when you first install VClock.

There is a lamp next to each type of command that can be received (GPI, IP, Livewire, http, snmp, serial etc), which turns green if connected, flashes black as data arrives, and shows red if there is a fault. In "fault" condition, the clock face will also flash "FAULT" once per second in the center of the clock (unless disabled on the settings page):



Backups

VClock will automatically take backups of the DEFAULTS.TXT and SALVOS.TXT whenever the configuration is saved., keeping the last 10. These are available in the File/Backups menu and can be reloaded from there. If you want to retain these old settings, you must subsequently save them again.

VClock will also zip up all config files (and any graphics that they refer to) into a BACKUPS subfolder of the config folder. It will keep the last x (1-99, default 10) based on the setting under the Miscellaneous tab in Settings. Setting to zero disables the feature. It does this whenever a config/salvo is saved, the settings pages are visited, or at startup if the config/salvos are detected as being different from the last one run (if you have edited defaults.txt outside of VClock for example).

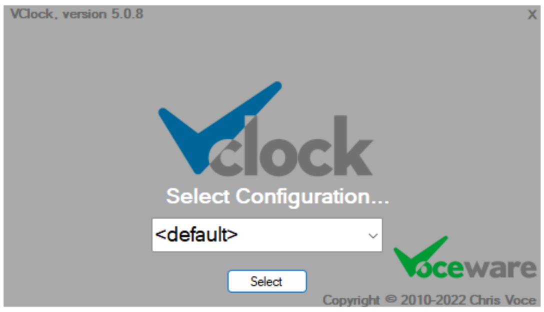
VClock will also save the zip file to a 2nd, "Remote Backup Path" if specified in the settings pages.

Startup

VClock will display a splashscreen for a short while whilst it loads its configuration. This will display details of the version and the licence.

An undocumented feature (well I guess it is documented now!) is the ability to select a Configuration at startup. It will list all subfolders of the ProgramData\Voceware\VClock folder and allow you to select one to launch VClock with (similar to specifying as a commandline variable). This is enabled by adding the following entries to VClock.ini:

```
[SplashScreen]
ConfigSelect=true
```

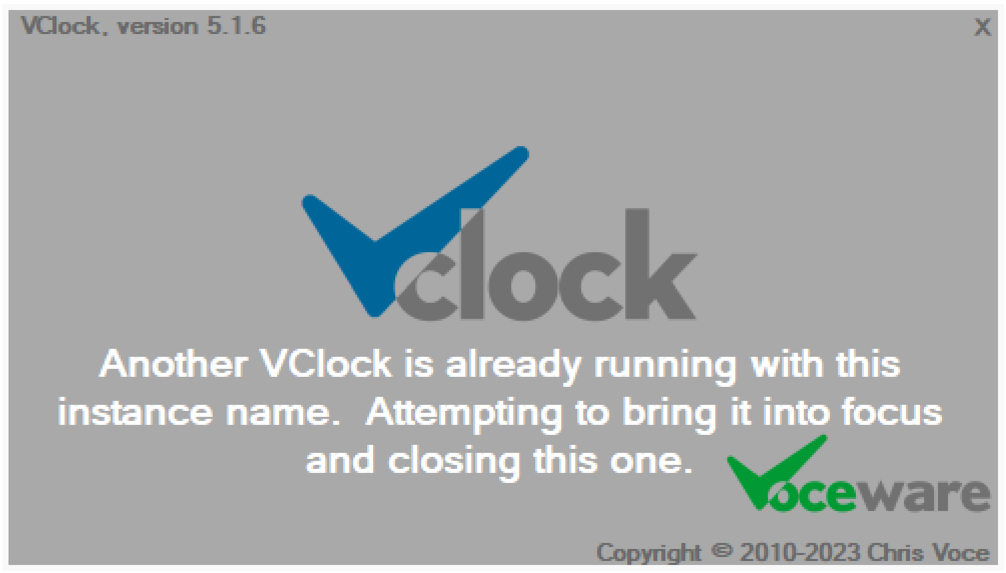


Useful if you are working on several configurations, or have a backup machine that could become one of several roles.

Duplicate Instances

You can run multiple instances of VClock, by specifying an instance name on the command line (see above). Each instance has its own subfolder for configuration files, and therefore its own unique configuration.

VClock will prevent multiple copies of the SAME instance from running, to prevent people accidentally launching the same VClock with the same configuration twice. If this happens, the new instance will display a message as below for a few seconds, and attempt to bring the original instance into focus.


VClock, version 5.1.6

Another VClock is already running with this instance name. Attempting to bring it into focus and closing this one.

Copyright © 2010-2023 Chris Voce

Menus

File / Updates: Manage the auto-update feature:

> **Check Now**: Checks www.voceware.co.uk immediately and informs you if there is a newer version, asking you if you wish to download and install it
>
> **Manual / Alert**: Set the mode of the automatic checking. VClock will check www.voceware.co.uk once every 24 hours for a new version, unless set to manual. **Alert** will inform you and you will then choose "check now" to download and install it.

File / Clock Properties: Allows you to **Save/Load/Edit the defaults**, **Save/Load/Edit any other file**, and **Edit the CustomClocks.txt file**. See "Other Settings on the Main Page" section shortly… these menu items do the same as the buttons on the main config page.

File / Backups: Shows the last (up to) 10 backups of DEFAULTS.TXT (Clock Config) and SALVOS.TXT (Salvos), and allows you to load these configuration. If you want to then retain them, you need to "Save as Defaults" or "Save Salvos".

File / Exit: Allows you to exit the application.

Tools / Settings: Opens the settings page to allow you to set up GPIs, IP, Serial, etc.

Tools / Generate Test File for Current Clock Language writes a text file, LanguageClockText.txt, which contains every minute of every hour, along with what the translation would be on the "language clock" on the bottom of VClock. Really useful to check that you have set the custom file up correctly.

Help / About: Gives you details of the applications author and contact details

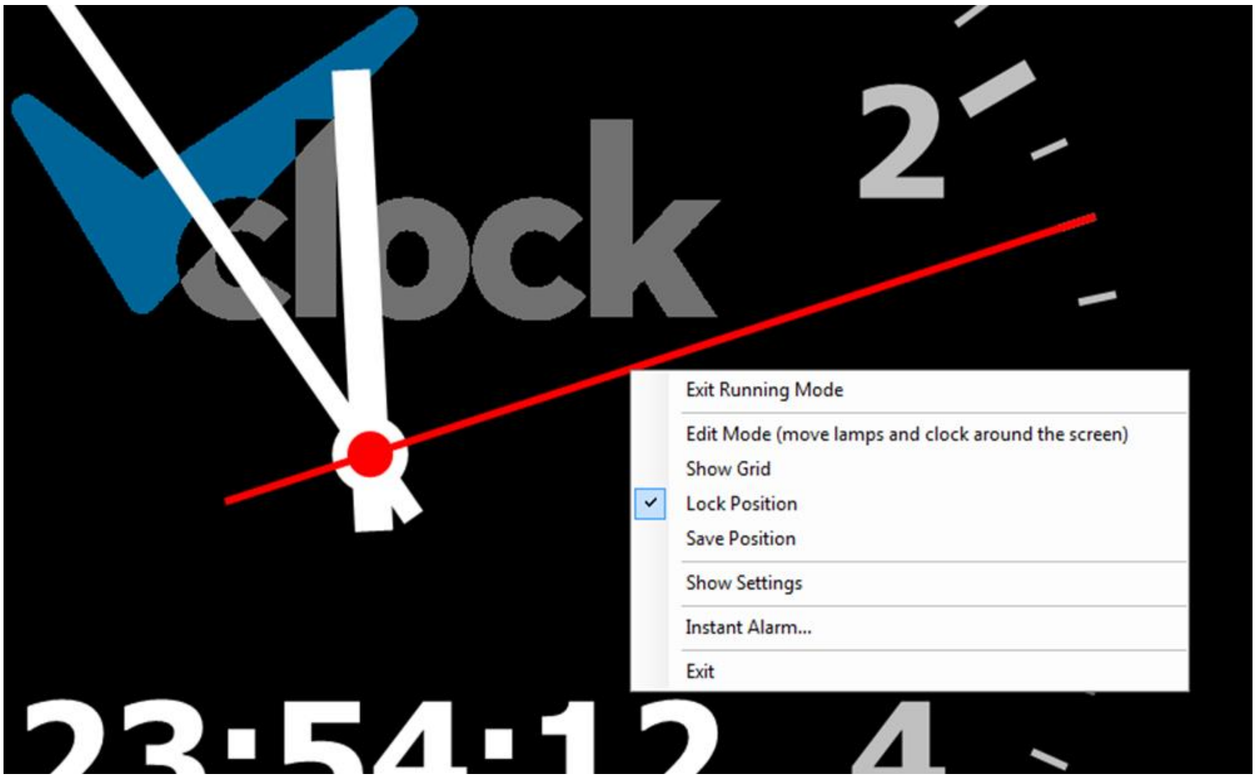
Help / View Manual: Views the version of this manual shipped with the clock

Help / View Revisions: Views the changes and revisions to each version of the clock

Running Mode: Changes to the "running" mode (removes the configuration parts of the screen and moves to the screen co-ordinates you choose). CTRL-F1 switches back, or ALT-F4 exits once in the "running" mode. There is also a command that can be used in Salvos, RUNNINGMODE=True|False.

Right-Click Menu

Right clicking the clock face also gives you a menu:

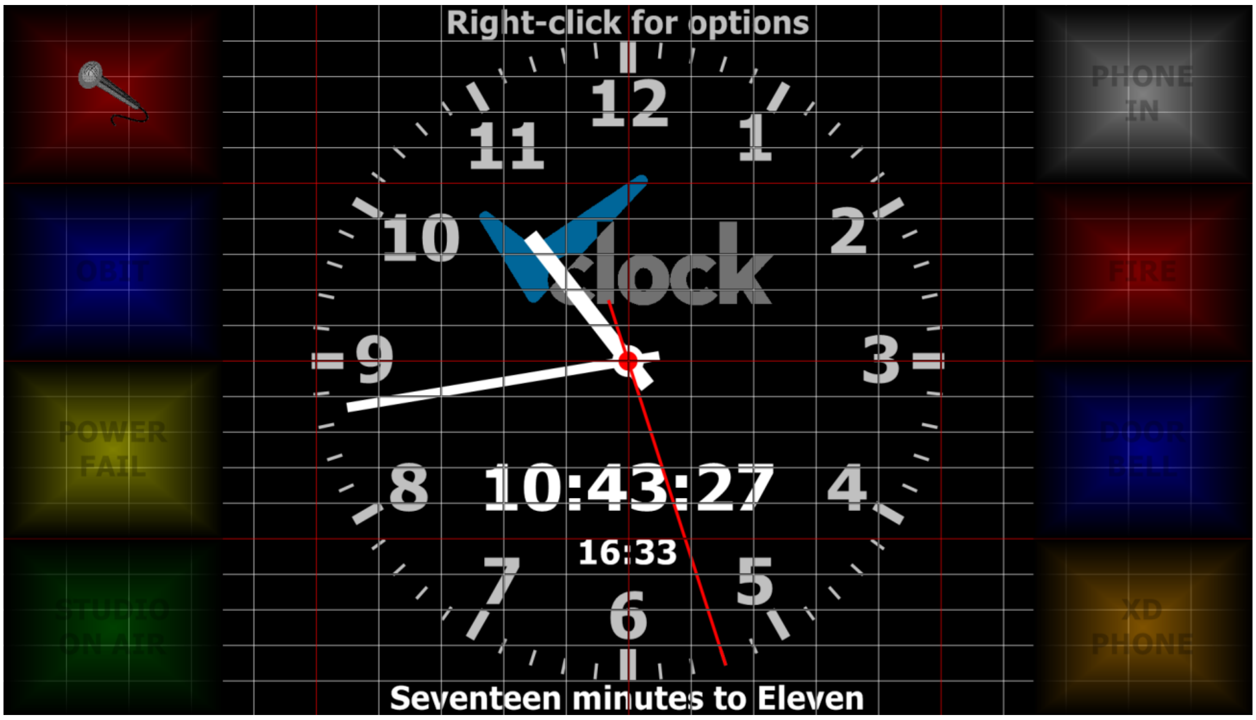


Exit Running Mode will toggle back to the configuration screen.

Edit Mode puts the clock into a special mode whereby you can click on any visible lamps and the clock face and drag them to different positions on the screen. This will change the **LampXPositionMode** or **ClockPositionMode** to Manual as soon as you move them, and sets the **LampPosition** or **ClockPositionCenter** settings. CTRL-DRAG will resize the elements and set the **ClockRadiusSizePercentage** for the clock face (it is part of LampPosition for a Lamp)

Edit mode is a useful way to get the lamps to approximately the position that you want, then tweaking the settings on the Properties pages later. CTRL-click and dragging the lamp will resize it rather than move it.

Show Grid will display white lines every 5% of the width and height, and red lines every 25%. This helps with lining up elements when dragging around in Edit Mode:



Lock Position stops click-dragging moving the VClock around the screen. It is the same as the Screen Coordinates Lock option on the settings page.

Save Position saves the current setting (and lock mode) to the config file, so it is remembered when VClock is next restarted.

Show Settings opens the settings page.

Instant Alarm allows you set a simple alarm which will play a sound repeatedly and flash "INSTANT ALARM" on the screen until the clock is clicked with a mouse. The sound that plays is a property of the clock, or can be set with the script command "InstantAlarmFile=", for example "InstantAlarmFile=alarm.wav".

Buttons on the Main Page

Save as Defaults will save the current properties to defaults.txt, overwriting the existing contents. This will be used whenever the clock is restarted. So don't save the settings whilst a lamp is lit(!).

Load Defaults will reload the contents of DEFAULTS.TXT into the clock

Edit Defaults.txt will open the defaults.txt file in Notepad for manual editing

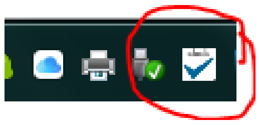
Save as … / Load File … / Edit File… do the same as the above, but allow you to select a file to save to / load / edit. These can be loaded with a special macro, "FILE=" , so can be used as pre-defined configs, such as changing the look for each presenter when they load a config in the studio console, or when 2 or more stations share a studio and want their own logos and colours when the studio is in use by them, etc.

Open Config Folder will open an explorer window in the program installation folder

The Combo Box below the clock contains examples of all of the different Macros you can use. Select one, modify the values then press "**Send Command**" to see what it looks like. Useful to test Salvos, etc – but the properties list is probably easier. It does however contain examples of some of the more complex settings, such as multiple backtime points and multiple TopCaptions to sequence through.

There are then 3 tabs across the bottom half of the screen. Salvos, Clock Properties and Debug Window. See their own sections for further details.

Tray Icon



VClock has a tray icon (a blue tick), next to the clock on the taskbar.

Right-Click Menu

By right-clicking the tray icon, the clock can be "hidden", and not shown on-screen or as a taskbar icon.   The tray icon can then be used to "show" the clock again. This is most useful when VClock is in "Relay Mode", and in fact when in this mode, the application will start up minimized to the tray.

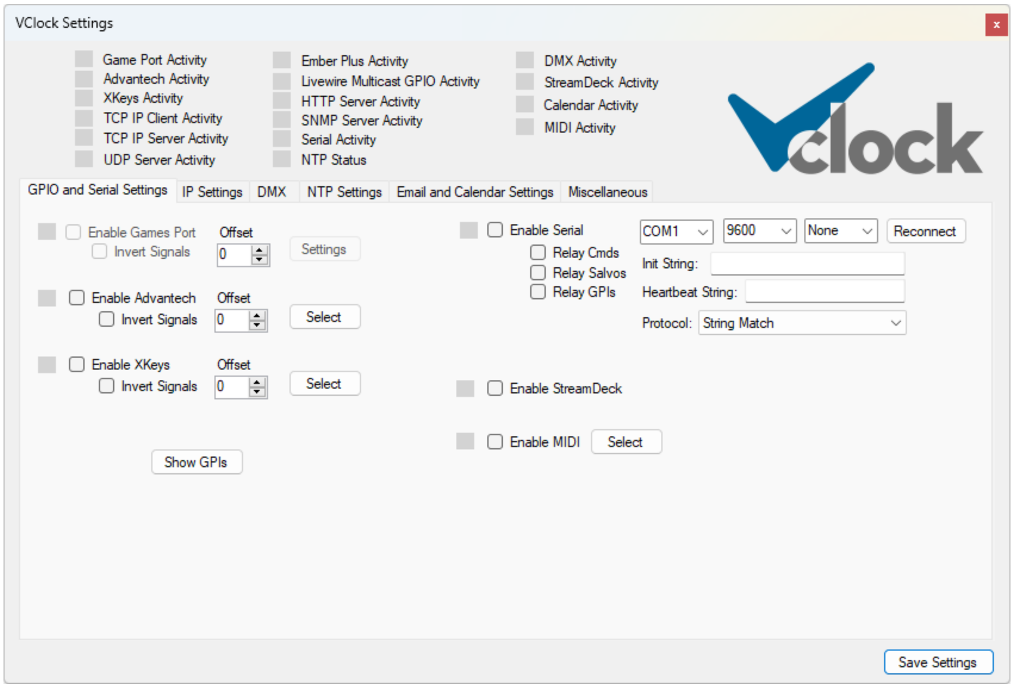
Settings Pages

Accessed from the Tools menu, the Settings pages give you access to all of the settings for the various interfaces of VClock, along with some other settings.

There are several interfaces to communicate with VClock (Game Port, XKeys devices, Advantech, TCP IP Clients, TCP IP Server, UDP Server, Livewire Multicast GPIO, HTTP Server, Ember Plus, SNMP Server and Serial port). Each of these has a status lamp on the settings page and on the main clock config page. Gray is disabled, Red is a fault and Green is a good connection. They also flash Black when data arrives on the ports. In the case of TCP IP Clients, where you can define several, each one has its own status indication in the table, and the overall status is a summary. If all lamps match then so does the summary lamp. If there is a mixture of Red and Green states, the summary lamp will show Orange. Similarly the Ember+ status lamp will show orange if there is 1 or more entry in the salvos table for Ember+ parameters that do not exist (or if there are none in the Salvos list but the connection is established).

The built in NTP Client also has a summary lamp on the main and settings pages. Red means there is an error communicating with the NTP Servers, orange means the time is out of sync, green means all is good.

GPIO and Serial Settings Tab



Offset: 0 (Games Port), 0 (Advantech), 0 (XKeys)

Serial: COM1, 9600, None

Protocol: String Match

Enable Games Port tells the clock to monitor a standard games controller port for GPI closures. A standard games port allows for 4 x "joystick buttons", which with a small adaptor plug can be used as a cost-effective solution to receiving GPI Commands. It functions in exactly the same way as an Advantech GPI (see below). 2 or more games ports can be used to allow more GPI closures. Other USB games controllers also work – if it is detected by Windows then VClock should be able to see its closures (buttons).

> **Invert Signals** changes a high input to trigger low and a low trigger as if it were high. See Appendix C for wiring of standard 4-button games ports.
>
> **Offset** applies an offset to any GPI numbers received. This is most useful when linking several clocks together – especially across different sites. If the offset it set to 20 and GPI 1 is enabled, this is seen as GPI 21. If each clock or site has a different offset then they can each act on the others GPI closures independently. It can also be useful if you want to use a Games Port **and** an Advantech card. The first of each is usually GPI 1. Applying an offset to one of them allows you to see them as separate GPIs.
>
> **Settings** opens the windows interface for games controllers, for testing purposes. If the device is not detected in here, VClock will not see it.

Enable Advantech tells the clock to monitor an Advantech GPI card for GPI contact closures and/or Analogue inputs (depending on what the card supports). When a GPI state changes on a pin, the relevant pin/state is checked in the salvo, and that salvo is triggered. Analogue values are available to be used in captions and so on via %variable settings (see "VClock Variables"). If the settings are grayed out then an Advantech card is not detected (or the Advantech drivers are not installed!).

> **Invert Signals** changes a high input to trigger low and a low trigger as if it were high. It is useful for some Advantech devices that have an inverter as part of the circuitry, so applying +v actually makes the card go low.
>
> **Offset** works in the same way as for games ports.
>
> The **Select** button allows you to select which Advantech device to use (though usually the default of the first device is the one that is required).

Advantech cards can also have GPO outputs, which VClock can also control (a Plus licence is required).

Enable XKeys tells the clock to monitor an XKeys XK-12, USB GPIO, USB In-a-box, XK-24, XKE-40 or other USB Interface from P. I. Engineering (https://www.x-keys-uk.com or https://xkeys.com). These are a range of devices from 3 / 12 GPI inputs to 128 key keyboards and rackmount panels. The settings work in the same way as the Advantech card above. Some XKeys devices (the USB GPIO and In-a-Box device) also have GPO outputs, which VClock can control. A Plus licence is required for this. The USB GPIO device has 10 selectable I/O pins which can be active low inputs, active high inputs, or outputs. This is configurable in the Selection dialogue box when the GPIO device is selected.

The **Show GPIs** button will show you a snapshot of all GPI's values (Advantech, XKeys and games port). It does not refresh automatically, but is useful to see how many GPI cards it has detected and so on.

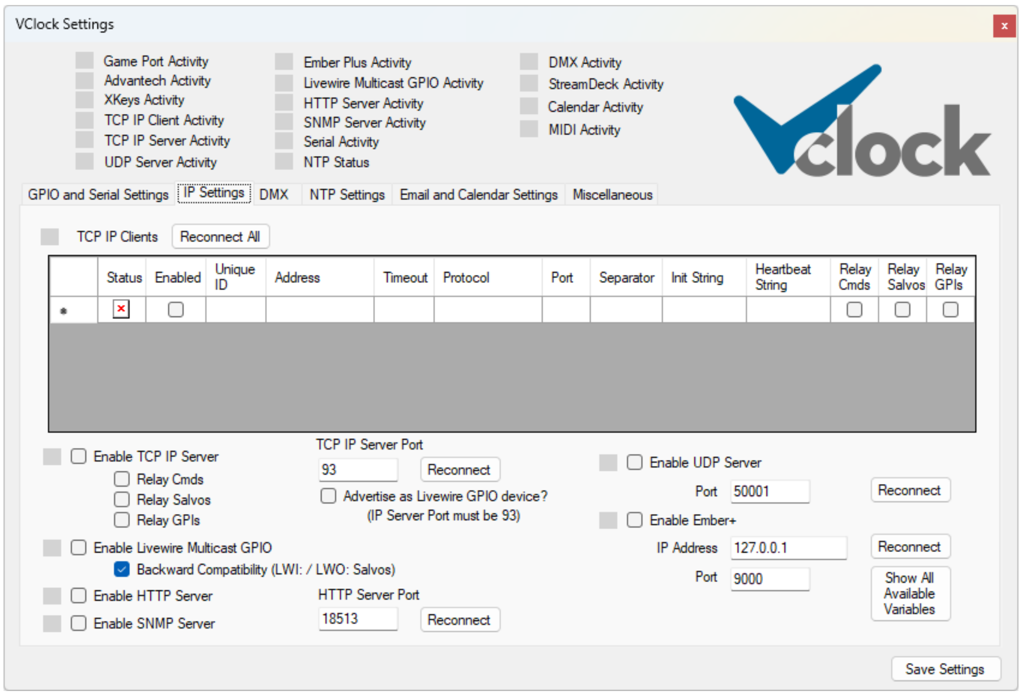
Enable **Serial Input** will open a COM Port (RS232 Port) at a configurable Baud rate and Parity. You can set an Init and Heartbeat command, and relay commands/salvos/GPIs from other sources to the serial port (see later for details). Text strings sent to the client will be looked up in the salvo table (first column) and a match will trigger that salvo.

> **Init String** is a string that is sent to the Serial port whenever it successfully connects to a device (usually only at the startup of VClock), used for asking for initial status, logging in, etc. Special examples for the serial port are:
>
> > ***GPIDUMP*** (request current status of all GPIs from another VClock)
>
> **Heartbeat String** is a string that is sent once per second to the port once a connection is established.
>
> **Protocol** specifies what protocol the device it is connecting to will speak to VClock with. By default it is "String Match", which will move character by character through the incoming data looking for a literal match. This can be time consuming for long strings, so specifying the protocol is best when possible. Protocols are:
>
> > ***String Match*** (check no protocols, just look for an exact string match)
> > ***BT SRC-8*** (Broadcast Tools SRC-8 GPIO device)
> > ***BT SRC-16*** (Broadcast Tools SRC-16 GPIO device)
> > ***BT SRC-32*** (Broadcast Tools SRC-32 GPIO device)
> > ***BT ACS 8.2C*** (Broadcast Tools ACS 8.2C GPIO and Audio switcher)
>
> The status lamp (repeated at the top of the settings page and on the main clock config page) shows the current status of the connection (Grey = Disabled, Red = fault, Green = Connected, Black (flash) = Incoming Data).
>
> The **Reconnect** button will reset the connection to the port if you change details, without restarting the clock.

Enable StreamDeck tells the clock to interface with an Elgato StreamDeck (https://www.elgato.com/en/stream-deck-xl).   This is a button panel (available with 6, 15 or 32 keys), each of which has a small display.   VClock can be triggered by key presses and can update images/text on the keys, or the keys can be linked to VClock Lamps for more control.

Enable MIDI enables a MIDI connection. The Select button is used to select which MIDI device to connect to, when there are more than 1.   MIDI is used to interface to some consoles, such as the RØDECaster.

IP Settings Tab



VClock Settings

Game Port Activity
Advantech Activity
XKeys Activity
TCP IP Client Activity
TCP IP Server Activity
UDP Server Activity
Ember Plus Activity
Livewire Multicast GPIO Activity
HTTP Server Activity
SNMP Server Activity
Serial Activity
NTP Status
DMX Activity
StreamDeck Activity
Calendar Activity
MIDI Activity

GPIO and Serial Settings | IP Settings | DMX | NTP Settings | Email and Calendar Settings | Miscellaneous

TCP IP Clients [Reconnect All]

	Status	Enabled	Unique ID	Address	Timeout	Protocol	Port	Separator	Init String	Heartbeat String	Relay Cmds	Relay Salvos	Relay GPIs
*	☒	☐									☐	☐	☐

☐ Enable TCP IP Server
☐ Relay Cmds
☐ Relay Salvos
☐ Relay GPIs
TCP IP Server Port: 93 [Reconnect]
☐ Advertise as Livewire GPIO device? (IP Server Port must be 93)
☐ Enable Livewire Multicast GPIO
☑ Backward Compatibility (LWI: / LWO: Salvos)
☐ Enable HTTP Server
HTTP Server Port: 18513 [Reconnect]
☐ Enable SNMP Server
☐ Enable UDP Server
Port: 50001 [Reconnect]
☐ Enable Ember+
IP Address: 127.0.0.1 [Reconnect]
Port: 9000 [Show All Available Variables]

[Save Settings]

It is possible to connect several **TCP IP Clients** simultaneously, so the interface for adding these is slightly different to the others (Serial, IP Server etc). It is a table where you can add rows to define each IPClient connection.

TCP IP Clients connect to a TCP IP Server on a machine IP address and Port. This could be any other device, or it could be a Master clock broadcasting out its salvos to slave clocks.

The settings are similar to the serial port, but are entered separately for each TCP IP Client connection. Some entries unique to TCP IP Client connections are:

Enabled is a way to disable each row entirely. Check to enable, uncheck to disable.

UniqueID is a free-text string (defaults to the row number) which is a way to reference the specific TCP IP Client when checking Salvos against incoming commands or when sending data.

Address and **Port** specify which host to connect to. Default values for the Port will be set when a Protocol is selected, though these can then be overridden if required.

The **Timeout** setting (default 50) is the port timeout in milliseconds when trying to establish the connection.

Protocol specifies what protocol the device it is connecting to will speak to VClock with. By default it is "Any", which will check all protocols and also a literal string match, moving character by character through the incoming data. This can be time consuming for long strings, so specifying the protocol is best when possible. Selecting a Protocol will also set a default Port, Separator, Init String and Heartbeat String if relevant. Protocols are:

Any (check all protocols and a literal string match (time consuming))
String Match (check no protocols, just look for an exact string match)
AEQ (AEQ Forum Consoles)
BARIX (Barix Barionet 50 Network GPIO device)
BLACKMAGIC (Blackmagic Smart VideoHub range)
BT SRC-8 (Broadcast Tools SRC-8 GPIO device via IP to serial device)
BT SRC-16 (Broadcast Tools SRC-16 GPIO device via IP to serial device)
BT SRC-32 (Broadcast Tools SRC-32 GPIO device via IP to serial device)
BT ACS 8.2C (Broadcast Tools ACS 8.2C GPIO and Audio switcher device via IP to serial device)
CLOUDCAST (Cloudcast Broadcast Delay Service (BDS)). The Unique ID of the IPClient connection also has to be set to the ID of the delay unit you want to communicate with.
DHD (DHD's ECP Protocol – DHD also support Ember+)
EMM88 (Audionics EMM88 8x8 audio switcher)
LAWO (Lawo's RAS protocol – Lawo also support Ember+)
LWCP (Axia Control Protocol for Fusion / Quasar / QoR / IQX / IQS Consoles)
LWCP+TIMER (the same as LWCP but also sync's the Consoles UpDown timers with VClock's UpDown counters.
LWCPIQ (A special version of the Axia Control Protocol for QoR / IQX / IQS Consoles, based around console control (CR Mon PGM selection etc)
LWRP (Axia Routing Protocol for Audio / GPIO routing)
PDM (Telos 25-Seven PDM Control and tally)
SAS (SAS (Sierra Automated Systems) Console control and tallys)
VCLOCK (connecting as a slave to another Master clock)
VMIX (Controlling VMix Video Mixing software)
WHEATNET (Wheatstone's protocol for SLIO GPIO triggering)

Separator is an end of command symbol which separates each incoming command. This removes the need to parse the incoming data 1 character at a time and again is more efficient. \n or \r can be used, along with standard characters or hex-encoded characters in the format <0x0a>. Default values will be set when a Protocol is selected, though these can then be overridden or removed if required.

Init String is a string that is sent to the IP port whenever it successfully connects to a device, used for asking for initial status, logging in, etc. Default values will be set when a Protocol is selected, though these can then be overridden or removed if required. Special examples for TCP IP Client are:

AXIA:INIT (or XNODE:INIT, LWCP:INIT or LWRP:INIT) all do the same thing (previously they were handled separately, so they all remain for backward compatibility). These are for LWRP connection on port 93, LWCP on port 4010 or LWCPIQ on port 4040, and register for GPI updates, crosspoint mappings, audio destination assignments, VMix statuses etc.

LWCP and LWCPIQ protocols will use triggers in the Salvos table to build the initialization string, so that anything monitored will be correctly initialized at startup.

As of Version 5.02, VMIX:INIT has been removed for this purpose, as it conflicts with support for VMix Video Mixing software support (see later).

CLOUDCAST:INIT (for use with a Cloudcast Broadcast Delay Service to register for updates on the current delay value)

CTP:INIT (for use with a CTP Systems DIO8008R Dante Audio Switcher, to get a dump of the currently active audio crosspoints at startup / reconnection).

LAWO:INIT (for use with a Lawo interface to get a dump of the entire current status of the console – so we can see what state all lamps are in)

MIDI:INIT (triggered when the MIDI input interface is connected / reconnected – MIDI doesn't have a Status command to get the latest values of everything, but this allows you to set some default states)

PDM:INIT (for use with a Telos 25-Seven PDM, to enable the ability to query the delay value)

SAS:INIT (for use with an SAS System – VClock checks any Salvos using SAS triggers and queries their current state to ensure VClock is accurate at startup / after a connection loss)

SALVO: (will trigger any Salvo with the following name. For example SALVO:CONNECTEDTOWIDGET will trigger any Salvo rows with a trigger of CONNECTEDTOWIDGET)

VCLOCK:INIT or GPIDUMP (the latter for backward compatibility) - requests current status of all GPIs from another "master" VClock as well as all memory slot names and values.

VMIX:INIT (for use with VMix Video Mixing software, not to be confused with VMixes in an Axia Console – see LWCPVMIX above for this). – requests startup status of the main output, preview and overlay layers.

WHEAT:INIT (to register for any used SLIOs in the Salvos table for Wheatstone).

In addition to these, a Salvo with ***UNIQUEID:INIT*** will be triggered whenever an IPClient reconnects (where the UNIQUEID matches the one set for that IPClient connection).

Heartbeat String is a string that is sent once per second to the port once a connection is established. Some equipment (though not VClock) require this to maintain the connection. Default values will be set when a Protocol is selected, though these can then be overridden or removed if required,

Wheatstone requires "**<>**" to keep the connection alive

Telos 25-Seven PDM's require "**PDM:HEARTBEAT**" to request the current delay once per second.

Enable **TCP IP Server** tells the clock to start a TCP IP Server on a port, for other applications (or slave VClocks) to connect to. Salvo commands can be sent to it just as with a TCP IP Client, and it can also broadcast special commands to slave clocks. This relaying works in the same way as with IP Client, though it blocks relaying Salvos received by the IP Server port.

Advertise as Livewire GPIO Device enables a network broadcast that allows Livewire / Axia to detect VClock as a GPIO Device. It isn't required – Pathfinder can be manually told about VClock. But it makes configuration easier. The IP Server needs to be set to port 93 for this to work (for VClock to receive GPIO from Pathfinder). Only 1 VClock on a machine can listen to this port, and other applications such as the Axia Livewire Driver may already be using it.

Enable UDP Server tells the clock to start a UDP listener on a port, for other equipment to send Salvo commands to.

Enable Ember+ tells the clock to connect to an Ember Plus "Provider" on a specified IP address and Port. VClock will add watches for any Ember Plus entries in the Salvos and receive updates as they change. The **Show All Available Variables** button opens a new window and will start to request all variables from the Ember+ Provider it is connected to. Depending on the complexity of the equipment, this can take several minutes! Once it has found them all, you can save them to a text file for later reference.

Enable Livewire Multicast GPIO tells the clock to listen for multicast GPIO from Axia/Livewire. The clock will then receive all GPIO on the network and can trigger events such as lighting lamps when it sees specified GPI or GPO multicast broadcasts. See Appendix D for more details. With this option enabled, VClock can also send multicast GPIO. The **BackwardsCompatibility** option allows Salvos to continue to be created in the format LWI: and LWO: and is enabled by default. This method was deprecated in version 4.7.7 in favour of LWMC:.

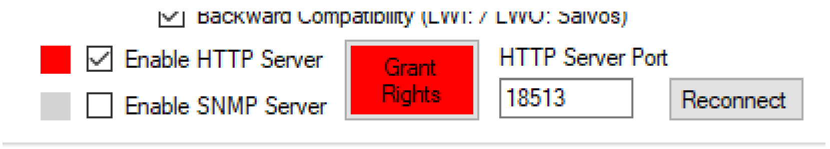
Enable HTTP Server tells the clock to start a HTTP listener on a port, for other applications to do http posts or http gets to. If you hit this port with a webbrowser, it will respond and tell you how to send commands to it. Relaying is not an option for this interface, as it is only an incoming connection. Commands received will however be relayed to the other interfaces if enabled.

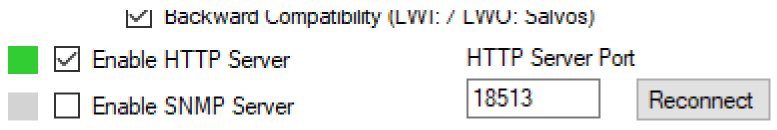
With Windows 10 comes some extra complication with the HTTP Server. By default Windows restricts the ability to open a listening port, so you have to authorize the user for the specific port you will be using. You do this by typing the following command in a command prompt which is "run as administrator":

```
netsh http add urlacl url=http://+:18513/ user=DOMAIN\USER
```

The 18513 is the port you want to use (and must match the port set in VClock). The DOMAIN\USER is the user you want to grant permission to, so the user that VClock will run as. If it is a local PC then you need to supply the "hostname" of the PC as the domain. If you type "whoami" at a command prompt, this will tell you the domain and user you are currently logged in as.

If HTTP Server is enabled, and VClock is unable to open the port, it will display a red button labelled "**Grant Rights**". This will run the above command as an elevated user (so will prompt for admin rights), which should then resolve the issue, and the connection should go green





Enable SNMP Server tells the clock to start an SNMP trap receiver, for other applications / devices to send an SNMP trap to the clock. This is again only an incoming connection, but received commands will be relayed as required.

Relaying

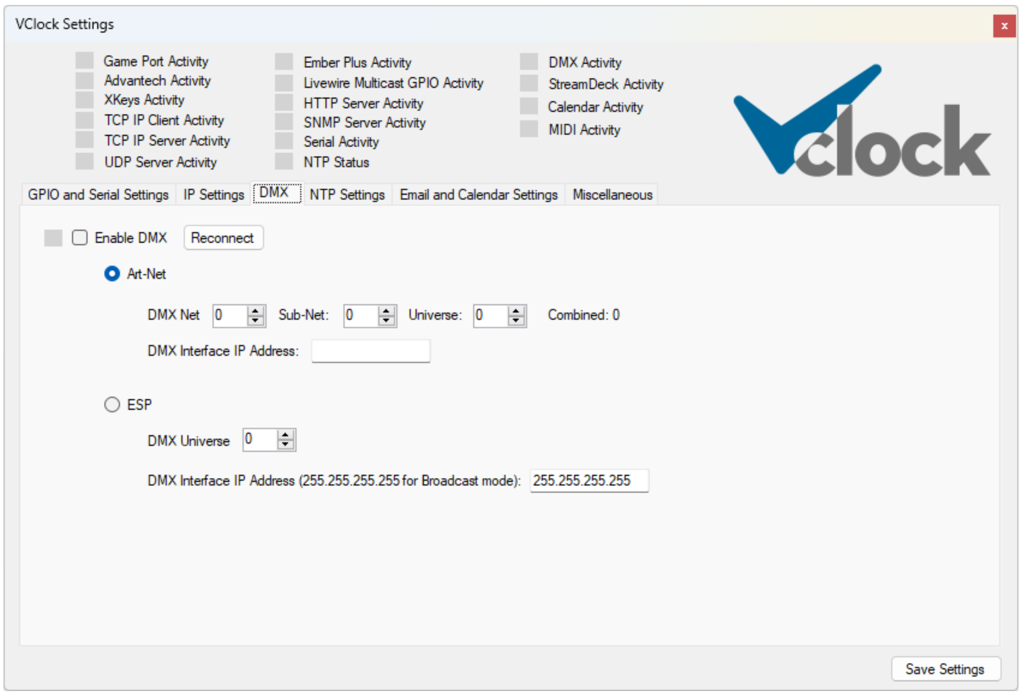
In each of the Serial, IPServer, and IPClient sections, it is possible to relay commands, salvos or GPIs that have been received from other ports. (VClock will not relay commands originated from a source back to that same source. This is to stop potential loops between 2 clocks, each relaying the incoming commands back to the other).

> **Relay Cmds** tells VClock to relay all of the commands that it processes to any other VClocks listening to the port (so this clock acts as a "Master" clock). The Slave VClock(s) mimic the behaviour of the Master clock.
>
> **Relay Salvos** tells VClock to relay the matching salvo names to other listening clocks, allowing the Slaves to have their own commands related to the salvo.

Relay GPIs tells VClock to relay GPI pin details (allowing a single GPI card to be used by several VClocks, each reacting to different GPIs for example).

For each of the above interfaces, if the license is invalid and all of the Grace Events are used up, then none of the received commands will be acted on.

DMX Tab


VClock Settings

Game Port Activity
Advantech Activity
XKeys Activity
TCP IP Client Activity
TCP IP Server Activity
UDP Server Activity
Ember Plus Activity
Livewire Multicast GPIO Activity
HTTP Server Activity
SNMP Server Activity
Serial Activity
NTP Status
DMX Activity
StreamDeck Activity
Calendar Activity
MIDI Activity

GPIO and Serial Settings | IP Settings | DMX | NTP Settings | Email and Calendar Settings | Miscellaneous

[ ] Enable DMX Reconnect

[● Art-Net]

DMX Net: 0 Sub-Net: 0 Universe: 0 Combined: 0

DMX Interface IP Address:

○ ESP

DMX Universe: 0

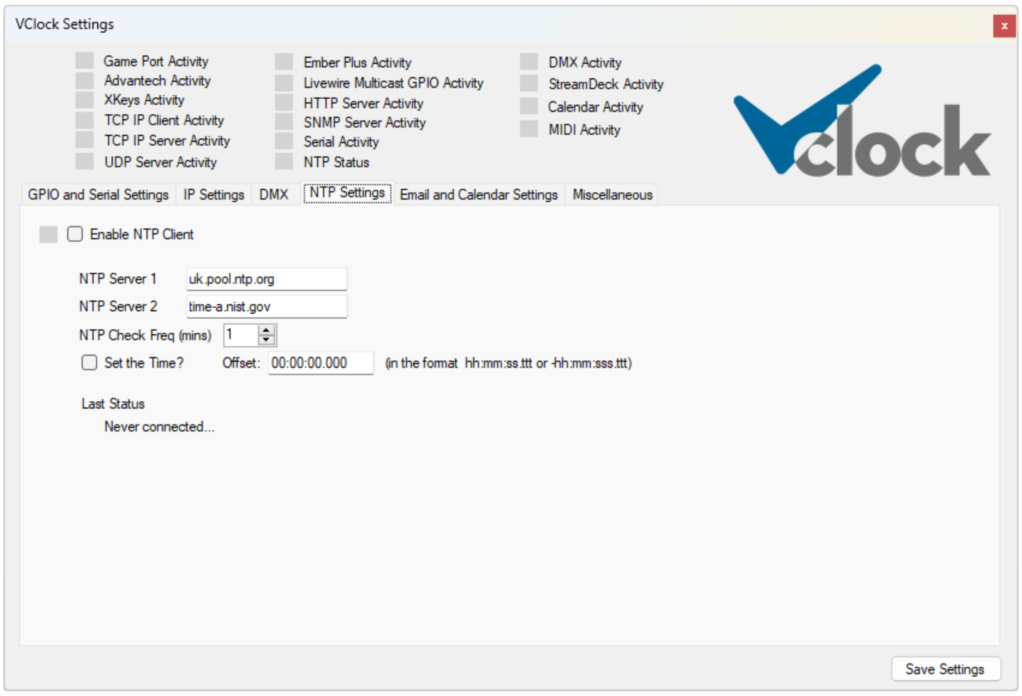
DMX Interface IP Address (255.255.255.255 for Broadcast mode): 255.255.255.255

Save Settings

VClock supports the Art-Net and ESP protocols for controlling DMX over IP. The DMX tab allows you to enable this support and select the protocol, along with setting the "universe" and endpoint IP addresses. With ESP, a broadcast address of 255.255.255.255 can be used, or you can specify the IP address of the unit you are controlling. With Art-Net it has the be the specific IP address.

A plus licence is required for the DMX functionality.

NTP Settings Tab


VClock Settings

Game Port Activity
Advantech Activity
XKeys Activity
TCP IP Client Activity
TCP IP Server Activity
UDP Server Activity
Ember Plus Activity
Livewire Multicast GPIO Activity
HTTP Server Activity
SNMP Server Activity
Serial Activity
NTP Status
DMX Activity
StreamDeck Activity
Calendar Activity
MIDI Activity

GPIO and Serial Settings | IP Settings | DMX | NTP Settings | Email and Calendar Settings | Miscellaneous

☐ Enable NTP Client

NTP Server 1: uk.pool.ntp.org
NTP Server 2: time-a.nist.gov
NTP Check Freq (mins): 1
☐ Set the Time? Offset: 00:00:00.000 (in the format hh:mm:ss.ttt or -hh:mm:sss.ttt)

Last Status
Never connected...

Save Settings

VClock has an NTP Client built in. It can be enabled to check-only, or to set the time. Use check-only if you have another time-sync solution, so that VClock can warn if this fails.

A big issue with most NTP Clients is that NTP Servers sometimes lie. They jump back 20 years or forward 3 hours. A common solution to this is to limit how much the time can be adjusted by – but this has its drawbacks if the date gets accidentally changed (lots of people use the clock settings as a calendar!).

VClock's solution is we think unique. We check 2 separate NTP Servers and only if they agree will we then accept the time as accurate.

Another unique option is to apply an offset to the time. VClock has other options to adjust the DISPLAYED time, but this option actually sets the PC clock to be offset from the NTP servers. Useful for example if the output of the PC will be on an IPTV system which adds a significant delay.

Not that an offset of "1" will be 1 day, not 1 second. Be sure to always use the format HH:MM:SS.ttt (or even D:HH:MM:SS.ttt).

If enabled, the NTP Client has 3 states. There are status indicators on the main VClock settings pages to show the state, and salvos also get triggered:

> **NTPERROR** (red indicator) means that we were unable to connect to one of the NTP Servers, or the 2 servers did not agree.
> **NTPSUCCESS** (green indicator) means that the time is in sync, or we successfully set the time to be in sync.
> **NTPWARNING** (orange indicator) means that the time is not in sync, but we are not configured to set the time.

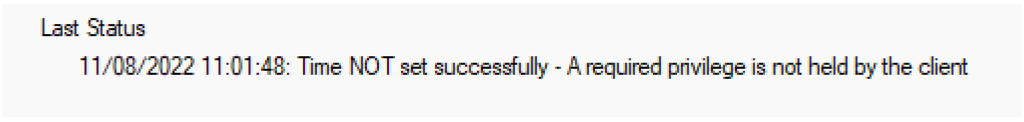
The salvos can be used to control a lamp or caption to indicate the status of time sync.

The variable **%NTPDIFFERENCE** can be used in any Caption to show the time difference between the local time and the NTP Server time, when VClock is set to check but not set the time.

VClock will show the latest status on the settings screen – including the error if it was unable to set the time.

User rights

One cause of not setting the time is user rights.

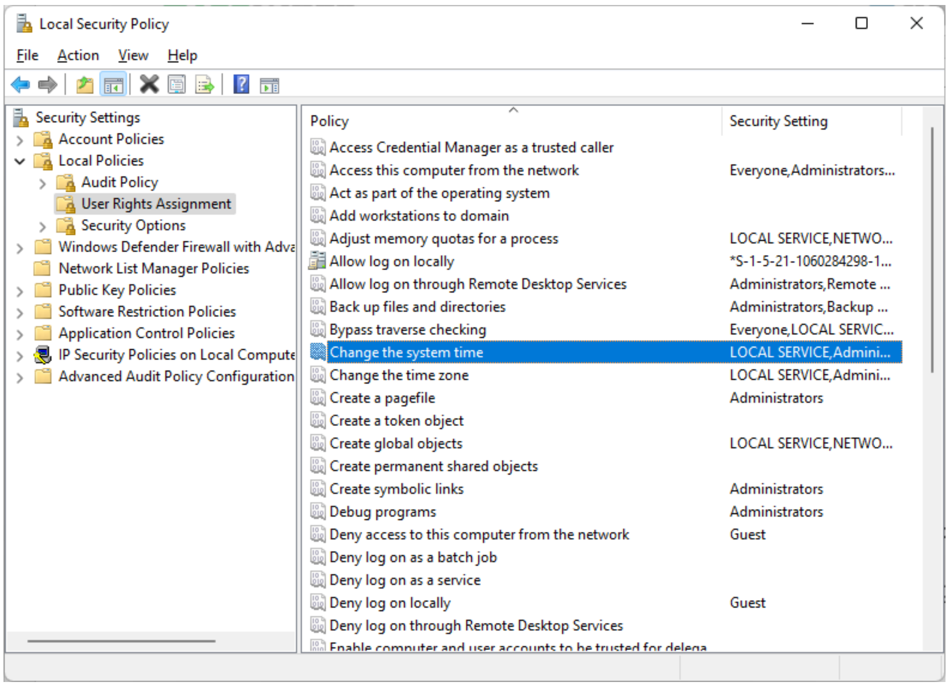


Last Status

11/08/2022 11:01:48: Time NOT set successfully - A required privilege is not held by the client

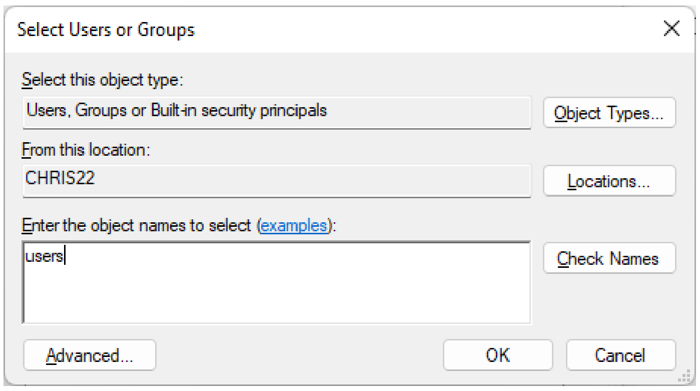
By default, standard users do not have the rights to set the time on a PC.   One solution is to run VClock "as administrator" by right clicking the shortcut and selecting this option.   But a better solution is to provide the user the correct rights.

Open "Local Security Policy" by clicking on the start menu and typing it. Then select "Local Policies / User Rights Assignment" on the left, and find "Change the system time" on the right:


Local Security Policy
File
Action
View
Help
Security Settings
Account Policies
Local Policies
Audit Policy
User Rights Assignment
Security Options
Windows Defender Firewall with Adva
Network List Manager Policies
Public Key Policies
Software Restriction Policies
Application Control Policies
IP Security Policies on Local Compute
Advanced Audit Policy Configuration

Policy	Security Setting
Access Credential Manager as a trusted caller	
Access this computer from the network	Everyone,Administrators...
Act as part of the operating system	
Add workstations to domain	
Adjust memory quotas for a process	LOCAL SERVICE,NETWO...
Allow log on locally	*S-1-5-21-1060284298-1...
Allow log on through Remote Desktop Services	Administrators,Remote ...
Back up files and directories	Administrators,Backup ...
Bypass traverse checking	Everyone,LOCAL SERVIC...
Change the system time	LOCAL SERVICE,Admini...
Change the time zone	LOCAL SERVICE,Admini...
Create a pagefile	Administrators
Create a token object	
Create global objects	LOCAL SERVICE,NETWO...
Create permanent shared objects	
Create symbolic links	Administrators
Debug programs	Administrators
Deny access to this computer from the network	Guest
Deny log on as a batch job	
Deny log on as a service	
Deny log on locally	Guest
Deny log on through Remote Desktop Services	
Enable computer and user accounts to be trusted for delega	

Double-click the entry, and add "users" to the allowed list. To do this, select "Add user or group", select "object types" at the top and ensure Groups are selected. Then simply enter USERS in the window and press OK



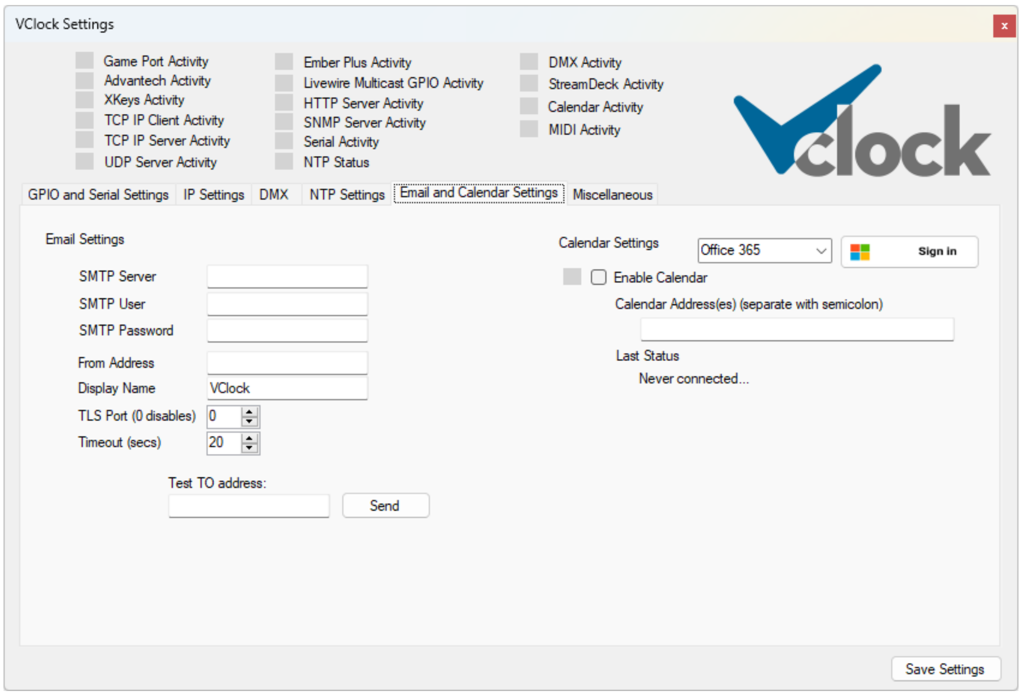
Select this object type:
Users, Groups or Built-in security principals

From this location:
CHRIS22

Enter the object names to select (examples):
users

The machine will need a reboot before this will take effect.

Email and Calendar Settings Tab



Email Settings

SMTP Server
SMTP User
SMTP Password
From Address
Display Name: VClock
TLS Port (0 disables): 0
Timeout (secs): 20
Test TO address:

Calendar Settings: Office 365
Enable Calendar
Calendar Address(es) (separate with semicolon)
Last Status
Never connected...

Email Settings are used to specify which SMTP server is used when VClock needs to send an email. **SMTPServer** and **FromAddress** are required fields. The others depend on the server. The **TLSPort** should be set to zero for "normal" servers and Timeout defaults to 20 seconds but can be increased for slow connections. Email settings are only available with a "Plus" license.

If you put an email address in the "**Test TO Address**" box and press "Send", the result will be displayed below it, so you can see if there are any errors.

Calendar Settings are used to connect to **Office365**, **Google Calendar** or a **VCalendar**, to collect calendar data to display room bookings.

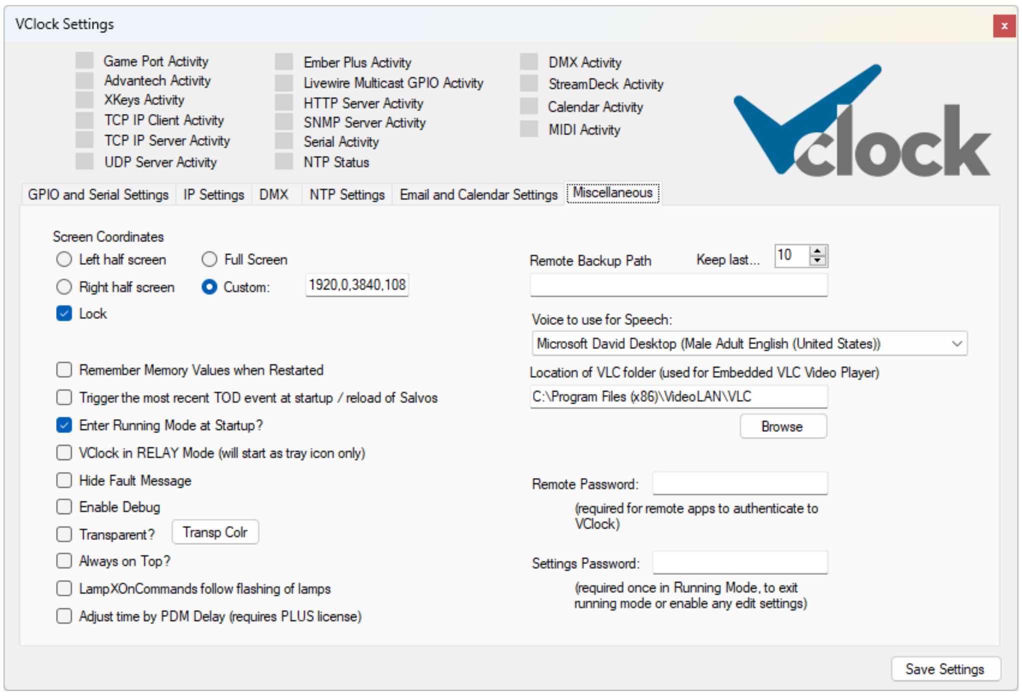
You will need to **authenticate** to generate a token from the settings pages, and specify which calendar address(es) you want to monitor (in a semicolon separated list).

IMPORTANT – Do not close the browser page when authenticating until prompted to – if you close it prematurely, VClock will never recover and you will have to End-Task the VClock process.

VClock will show the last status of the authentication attempt or calendar check. It will poll the calendar(s) once a minute.

A plus licence is required for the email and calendar functionality.

Miscellaneous Tab


VClock Settings

- [ ] Game Port Activity
- [ ] Advantech Activity
- [ ] XKeys Activity
- [ ] TCP IP Client Activity
- [ ] TCP IP Server Activity
- [ ] UDP Server Activity
- [ ] Ember Plus Activity
- [ ] Livewire Multicast GPIO Activity
- [ ] HTTP Server Activity
- [ ] SNMP Server Activity
- [ ] Serial Activity
- [ ] NTP Status
- [ ] DMX Activity
- [ ] StreamDeck Activity
- [ ] Calendar Activity
- [ ] MIDI Activity

Tabs: GPIO and Serial Settings | IP Settings | DMX | NTP Settings | Email and Calendar Settings | Miscellaneous

Screen Coordinates
- Left half screen
- Full Screen
- Right half screen
- [✓ Custom:] 1920,0,3840,108
- [✓] Lock

- [ ] Remember Memory Values when Restarted
- [ ] Trigger the most recent TOD event at startup / reload of Salvos
- [✓] Enter Running Mode at Startup?
- [ ] VClock in RELAY Mode (will start as tray icon only)
- [ ] Hide Fault Message
- [ ] Enable Debug
- [ ] Transparent? Transp Colr
- [ ] Always on Top?
- [ ] LampXOnCommands follow flashing of lamps
- [ ] Adjust time by PDM Delay (requires PLUS license)

Remote Backup Path   Keep last... 10

Voice to use for Speech: Microsoft David Desktop (Male Adult English (United States))

Location of VLC folder (used for Embedded VLC Video Player): C:\Program Files (x86)\VideoLAN\VLC   Browse

Remote Password:
(required for remote apps to authenticate to VClock)

Settings Password:
(required once in Running Mode, to exit running mode or enable any edit settings)

Save Settings

Screen Co-ordinates allows you to specify where you want the clock to appear. When first started, VClock will set the co-ordinates to be full-screen of the main monitor if they have not been defined.

There are 2 ways to easily locate the clock in the position you want.

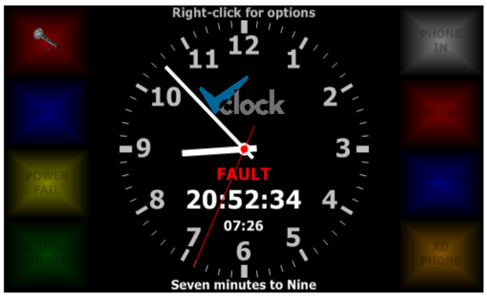
The first is to drag the application to the screen you want the clock to appear on (assuming there is more than 1), then select the "Full screen" option on the settings page. VClock will work out the co-ordinates for you. Similarly "Left half screen" and "Right half screen" will split the current screen down the middle and set VClock to use the left (or right) half. Once saved, it is the actual co-ordinates that are saved which are reloaded when VClock starts.

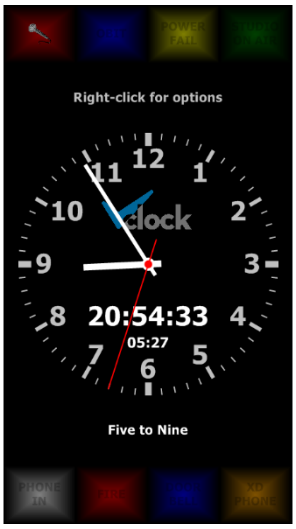
Another way to locate the clock in the position you want, if it isn't as "standard" as above is to enter "Running Mode", then drag the clock into position. If you right-click the clock, there is a "Lock position" option. You need to turn this off in order to position the clock. Simply drag the clock face with the left mouse key. To resize it, hold the CTRL key while dragging. Once in position, if you go back to the settings page then the screen co-ordinates will have been updated.

Tick the **Lock** tickbox to stop the clock from being movable by dragging, and **Save Settings** to make this the default when the clock is started. (You can also right-click the clock and save the co-ordinates or lock the clock. Using these right-click options automatically save these settings for future restarts, but does not save any other settings).

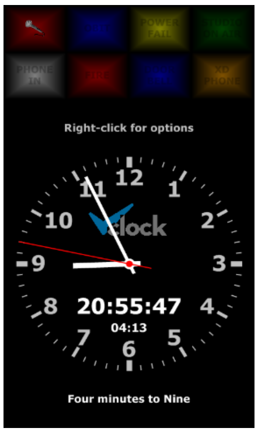
The format for Screen Co-ordinates is "LEFTPOS,TOPPOS,RIGHTPOS,BOTTOMPOS". (so 4 numbers separated by commas, no spaces). For example, if your desktop size is 1024x768 you would set the Co-ordinates to be "0,0,1024,768" to fill the screen. If you had a 2nd screen to the right with the same resolution and you wanted the clock to appear there, you would set it to "1024,0,2048,768".

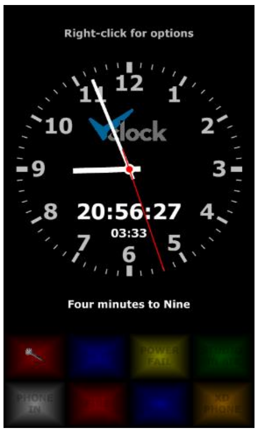
By default, if the screen resolution is wider than it is high, the lamps will appear down the side. Similarly, if the resolution is higher than it is wide (say a standard widescreen monitor mounted on its side), then the lamps will appear across the top and bottom of the screen (below is a 16x9 screen in its two orientations):

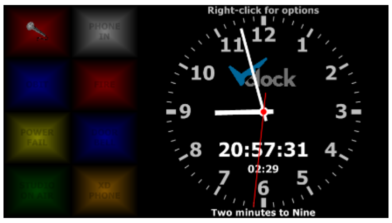


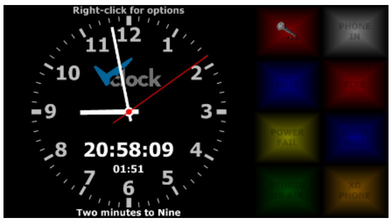


It is also possible to override these positions using the LampsPosition setting:









Remember Memory Values when Restarted will automatically write any memory value changes to a text file (MemorySlots.txt in the config folder), and automatically read these values back in when VClock is started. When unchecked, VClock will start with no memory slots each time it is restarted.

Trigger the most recent TOD event at startup / reload of Salvos scans the Salvos when reloaded, and finds the most recent one in the past for "today" that would have been triggered, and fires it again. Useful to ensure the clock is in the right state for that time of day when showing branding, for example.

Enter Running Mode at Startup will start the application in "running" mode (ie not Configuration mode). This is the default. If you start the clock in this mode, pressing **CTRL-F1** will get you back to the Configuration Mode (and again to go back to VClock mode), or you can right-click the clock to get a menu.

VClock in RELAY Mode will enable a special mode, which actually disables most of the clock. See next section for details.

Hide Fault Message will stop VClock from flashing "FAULT" in the center of the clock face if one of the input methods is in a fault state (IP, GPI, SNMP, HTTP, Serial).

Enable Debug will tell VClock to write useful information to a debug.log file, moving the file one per hour into a debug folder and keeping them for 10 days.

Transparent makes the background transparent in OS's that support it (Windows 7+). This works by nominating a colour that you are not using on the clock as the "transparent" colour. Anything set to this colour will become transparent. The "**Transp Colr**" button allows you to specify this colour (so as to avoid colours in your logo, lamps, etc).

Always On Top will keep VClock in front of other applications on the PC.

LampXOnCommands follow flashing of lamps will make associated commands for LampXOnCommand and LampXOffCommand repeatedly trigger whilst a lamp is flashing in any way, rather than a single on and off at the start and end of the state change.

Adjust Time by PDM Delay will adjust the overall time across the whole of VClock by the 25-Seven PDM or CloudCast BDS's current delay (or a value set with the PDMDelay=[ss.hhh] command). If checked, backtiming will be relative to post-delay time and will adjust if "dump" is pressed, etc. The second hand will show post-delay time and if enabled, the ClockRTCTrace will trail behind the second hand to show the real time. If Adjust Time by PDM Delay is not checked, the second hand will show real time and the ClockRTCTrace will lead the second hand, showing the post-delay time.

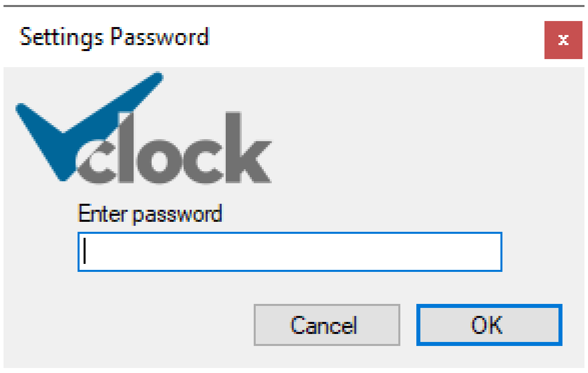
Setting a **Config Backup Path** will zip all images/ini files/licence files/text files in the program folder and save them to the backup path. **Keep Last** tells VClock how many previous backups to keep before deleting them. Backups are created each time the clock starts or the config or salvos are saved. A blank setting (default) disables the feature, though VClock will still make local backups in a "BACKUPS" subfolder of the config folder, unless the Keep Last is set to zero.

Voice to use for Speech allows you to select the voice that VClock uses when using the SPEAK= command. A "VClock Plus" license is required.

Location of VLC Folder is required to use the embedded VLC Players. This should auto-populate if VLC is installed in a standard folder. Type in the correct path or browse to it if not. If this is changed, you will need to reload the defaults, or restart VClock for it to take effect. VClock is tested against VLC v3.0.8. Only 32 bit versions are supported. A "VClock Plus" license is required.

Remote Password is required for some remote apps to connect to VClock, such as VClockSalvoEditor. Ask for further details.

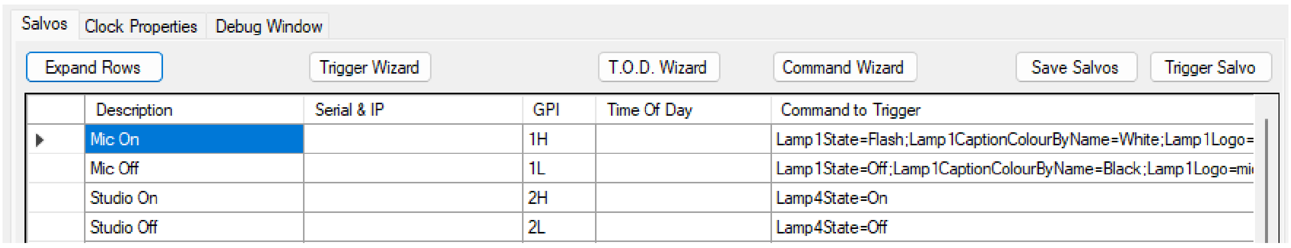
Settings Password if set, will request a password when a user tries to use most of the right-click menu options when in running mode (Edit Mode, Show Grid, Lock Position, Save Position and Exit). Once entered, the password is remembered until VClock is put back into Running Mode.



Passwords are saved in VClock.ini in an encrypted form. If you ever need to reset the password because you have forgotten it, you can delete the [Passwords] section entirely, or set a new password and change the Encrypted option to false (it will re-encrypt when you start VClock).

Save Settings will write all of the above configurations into an ini file, so that they are remembered for the next time you launch VClock.

Salvos

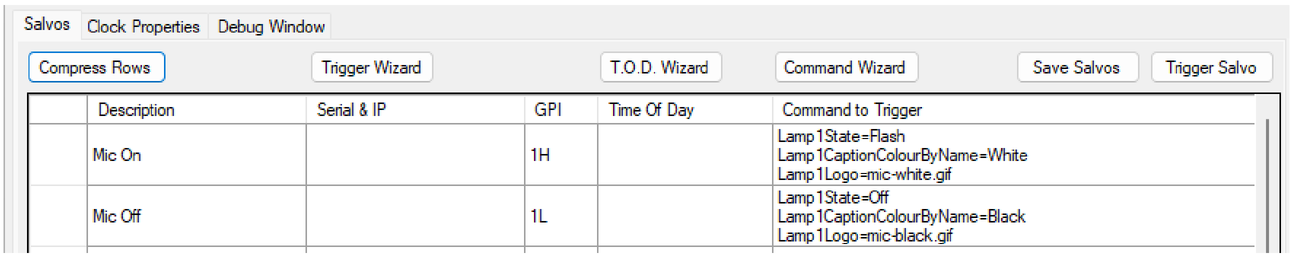


Description	Serial & IP	GPI	Time Of Day	Command to Trigger
Mic On		1H		Lamp1State=Flash;Lamp1CaptionColourByName=White;Lamp1Logo=
Mic Off		1L		Lamp1State=Off;Lamp1CaptionColourByName=Black;Lamp1Logo=mi
Studio On		2H		Lamp4State=On
Studio Off		2L		Lamp4State=Off

Salvos are the way to change the look of VClock when events are received from external sources (IP, Serial, GPI, http).

For each row in the table, the "Description" is just a reminder to you of what the row does. The "Serial & IP", "GPI" and "Time of Day" columns are triggers (see later) that will cause the command in the "Command to Trigger" column to happen. This can be 1 or more commands separated by semicolon (";").

The **Expand Rows** button will separate each command in the "Command to Trigger" column onto separate lines, to make it easier to edit. Ditto the "Time of Day" column. Adding additional entries to these columns can be done either with SHIFT-ENTER to add a new line, or by separating with a semicolon. When enter is pressed, the field will be modified to show in the selected format.



Description	Serial & IP	GPI	Time Of Day	Command to Trigger

Mic On		1H		Lamp1State=Flash
				Lamp1CaptionColourByName=White
				Lamp1Logo=mic-white.gif

Mic Off		1L		Lamp1State=Off
				Lamp1CaptionColourByName=Black
				Lamp1Logo=mic-black.gif

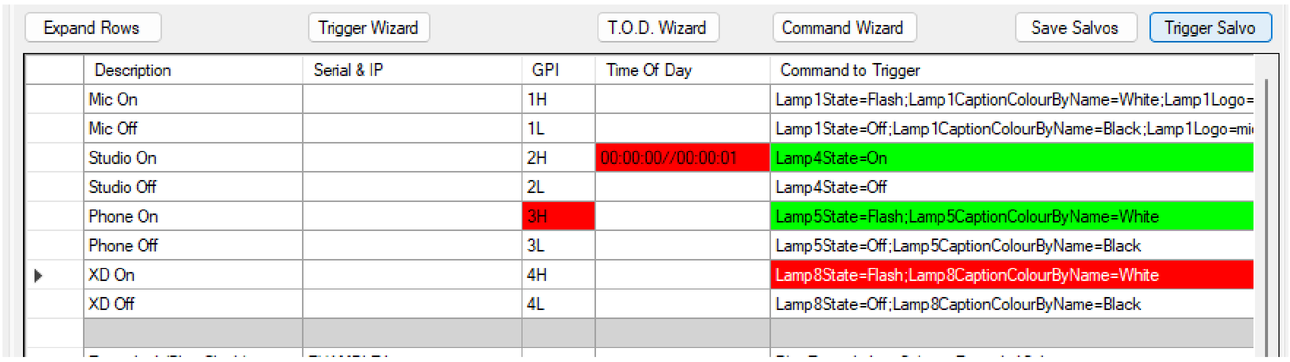
Save Salvos and **Trigger Salvo** on the Salvos tab are used when creating the salvos in the table below. "Save Salvos" writes the contents of the table to a text file (SALVOS.TXT). "Trigger Salvo" will run the Macros associated with the currently selected row in the table, so you can see what it does.

The Combo Box below the Salvos table (again on the Salvos tab) allows you to mimic sending a Salvo to VClock. Type in your own or select one of the example ones for more complex salvos such as an SNMP trap, then press **"Test Salvo"**.

Colouring of the Salvo Table

Blank rows, or rows with only a Description, are coloured gray. This is to help separate different sections of your configuration.

When a Salvo is triggered, the respective trigger field lights red briefly, and any command that is then subsequently triggered turns green. For example GPI 3 went high and the timer triggered below. Manually triggered commands (with the "Trigger Salvo" button) turn red (for example the "XD On" row below):



Description	Serial & IP	GPI	Time Of Day	Command to Trigger
Mic On		1H		Lamp1State=Flash;Lamp1CaptionColourByName=White;Lamp1Logo=
Mic Off		1L		Lamp1State=Off;Lamp1CaptionColourByName=Black;Lamp1Logo=mi
Studio On		2H	00:00:00//00:00:01	Lamp4State=On
Studio Off		2L		Lamp4State=Off
Phone On		3H		Lamp5State=Flash;Lamp5CaptionColourByName=White
Phone Off		3L		Lamp5State=Off;Lamp5CaptionColourByName=Black
XD On		4H		Lamp8State=Flash;Lamp8CaptionColourByName=White
XD Off		4L		Lamp8State=Off;Lamp8CaptionColourByName=Black

GPI

Games controller ports, XKeys USB Interfaces and Advantech cards are currently the only physically supported devices. You need to install DirectX 9.0 for the games port, and the DAQNavi driver in order to use an Advantech card with VClock. However it is also possible to set "virtual" GPIs within VClock, by using the command GPI=1H, etc.

When a GPI is seen as going "high", the GPI column of the salvos table is searched for any matches. A match is the GPI number on its own, or the GPI number followed by H (for "high"!). If a GPI is seen as going "low", it must match the GPI number followed by L (for "low"!).

There is some basic but powerful logic available for GPIs. It is possible to require more than 1 GPI to trigger a salvo (set the GPI column to "1H+2H" for example to require both GPI 1 and 2 to be high to trigger it). It is also possible to list several sets of GPIs that can trigger the Salvo, separated by comma. So for example: "1H+2H,3H" would mean that either GPI 1 AND 2 need to be on, or 3 on its own would also trigger the salvo.   To turn off an *and* of "1H+2H", you would need another salvo that was 1L *or* 2L ("1L,2L"). Another way to achieve this is to list all of the valid states but start the command with ! to invert the logic. So "!1H+2H,3H" would trigger if neither either 1 and 2 were not high, AND 3 was not high. If you have several GPIs involved then the number of potential combinations gets exponentially greater. But the inverse command allows you to catch all invalid states with 1 short(ish) command.

Often you need to check the state of lots of GPIs, which can result in a long string

```
1L+2L+3L+4L+5H+6L+7L+8L
```

This can be shortened by preceding a state with a comma separated range, in square brackets. So to achieve the same result as above you could use:

```
[1-4,6-8]L+5H
```

Serial / IP / HTTP / Ember+

A text string received by any of the other interfaces is checked against the "Serial & IP" column of the salvos table.

The HTTP Server can be hit with any standard web browser, by hitting http://ipaddress:port (for example http://localhost:18513). This will display basic information on how to use the port, but to trigger a salvo you would send something in the format of http://localhost:18513/VClock?Salvo=TOPLEFTFLASH . The clock would act on "Salvo=TOPLEFTFLASH". So basically everything after the ? would be sent to the clock.

There are some special cases where the incoming data is in a complicated format. VClock offers some shortcuts to process these.

LAWO: AEQ: LW: LWO: LWI:, LWA:, LWMIX:/XNODE:, DHD:, WHEAT:, LWCPBUTTON, LWCPMONBUTTON, LWCPIQ, LWCPPROFILE, LWCPCRMUTE, LWCPSTMUTE, LWCPVMIX:, EMM88:, EMBER+:, BARIX: CLOUDCAST:, SAS: and BMROUTE: are all examples of special cases for incoming data. Where they use the IPClient, it is also possible to specify which IPClient they are relevant to by adding ~UNIQUEID (where UNIQUEID is the one specified for that IPClient on the settings page). For example WHEAT~BLADE1:1,ON would translate to a more complex string being received by the specific IPClient with the Uniqueid "BLADE1". WHEAT:1,ON would trigger regardless of which IPClient it arrived on).

Optionally in the Salvos table, you can use **%BUFFER%** in the "Command to Trigger" column. This will get replaced with the remainder of the salvo received via the HTTP Server, IP Server or IPClient.

For example if the incoming Http Post is http://localhost:18513/VClock?xpwd=changeme&title=Billie Myers-Kiss the Rain, the Salvo to match is set to is "title=" and the command is set to "TOPCAPTION=%BUFFER%", then the resulting command will be "TOPCAPTION=Billie Myers-Kiss the Rain".

You can also use **%BUFFERTOx%** in the Command to Trigger (replacing x with any character), which will take the remaining buffer, but only up to the specified char.

Using the same example but with incoming IPClient data, if the incoming data is title=Billie Myers-Kiss the Rain, the Salvo to match is set to is "title=" but the command is set to "TOPCAPTION=%BUFFERTO-%", then the resulting command will be "TOPCAPTION=Billie Myers".

Time Of Day

A string in the format TIME/DAYS/REPEAT/UNTIL, where:

- TIME (in the format "HH:MM:SS") is the time of the trigger. Field is required.
- DAYS (in the format "MoTuWeThFrSaSu" or "All") is which day(s) the command is valid. Default is "All" if section is left blank.
- REPEAT (in the format "HH:MM:SS") is how often the trigger is valid after the initial TIME. For example "00:00:02" would trigger it every 2 seconds after the initial TIME (default is 00:00:00 – ie no repeat if left blank).
- UNTIL (in the format "HH:MM:SS") is when the trigger becomes invalid (default midnight if left blank).

For example: 09:00:00/MoTuWeThFr/00:00:02/17:00:00 would trigger (say a lamp coming on) every 2 seconds from 9am to 5pm Monday to Friday. A reciprocal command of 09:00:01/MoTuWeThFr/00:00:02/17:00:01 could turn it off again (1 second offset from the first), meaning the lamp flashes once per second.

Several time ranges can be set on a single Salvo row, by separating them with comma or semicolon. So for example:

```
10:00:00,11:00:00/MoTu,12:00:00
```

… would trigger at 10am every day, 11am only on Mondays and Tuesdays, and at 12 noon on all days.

Command To Trigger

In any case, when a match is found in the salvos table, the commands in the 4th column are sent to VClock and acted upon. Each command should be separated by a semicolon.

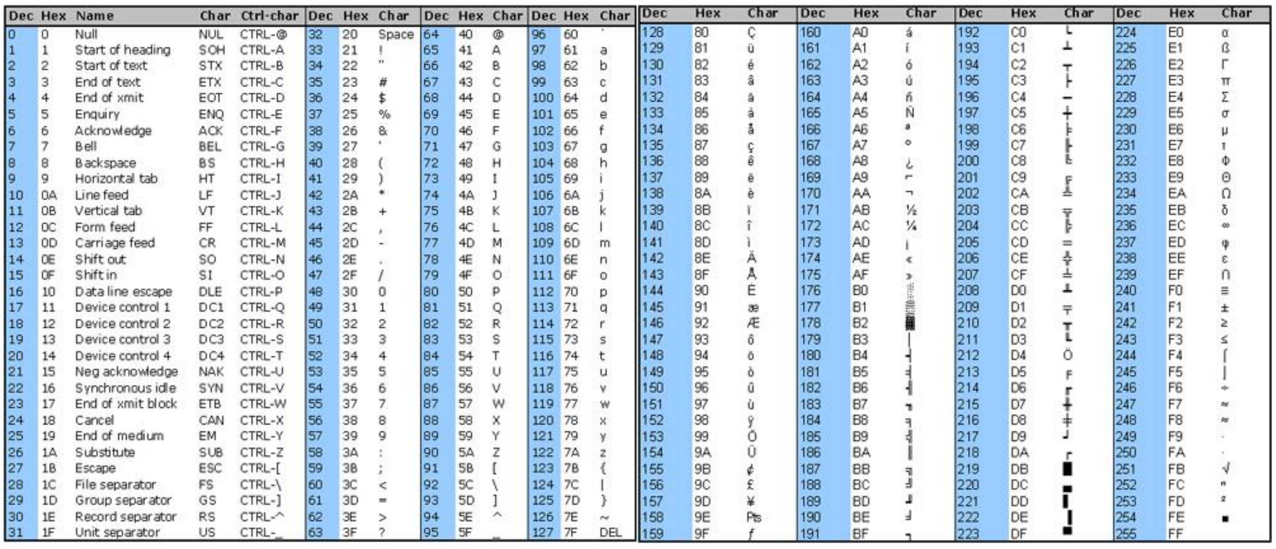
One powerful trick when using IP/HTTP/SNMP Salvos, is to set the command to be GPI=xH or GPI=xL (where x is any GPI number). This way, the incoming salvo can be converted to a virtual GPI closure, which can then be used in the GPI logic detailed above (to require more than 1 thing to happen before triggering a salvo, etc).

Another trick with GPIs is that you can toggle the value of a GPI using the command GPI=x! (for example GPI=21! Will invert the value of GPI 21). This is particularly useful for salvos triggered by the press of a "lamp". This can toggle the state each time, then the GPI value (high or low) can be used to send data and update the title of the lamp.

With the Serial / IP commands, data can be received that is not easily printable. For instance the ASCII character 0 is "null", or 128 is "Backspace". To solve this, for characters below 32 or above 127, VClock interprets the characters to be "<0xFF>", where FF is replaced with the Hex equivalent of the character code.

For example, "Hello<return>" is expressed as "Hello<0x0d>".

A full list of ASCII character codes is:



Dec	Hex	Name	Char	Ctrl-char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	0	Null	NUL	CTRL-@	32	20	Space	64	40	@	96	60	`
1	1	Start of heading	SOH	CTRL-A	33	21	!	65	41	A	97	61	a
2	2	Start of text	STX	CTRL-B	34	22	"	66	42	B	98	62	b
3	3	End of text	ETX	CTRL-C	35	23	#	67	43	C	99	63	c
4	4	End of xmit	EOT	CTRL-D	36	24	$	68	44	D	100	64	d
5	5	Enquiry	ENQ	CTRL-E	37	25	%	69	45	E	101	65	e
6	6	Acknowledge	ACK	CTRL-F	38	26	&	70	46	F	102	66	f
7	7	Bell	BEL	CTRL-G	39	27	'	71	47	G	103	67	g
8	8	Backspace	BS	CTRL-H	40	28	(	72	48	H	104	68	h
9	9	Horizontal tab	HT	CTRL-I	41	29	)	73	49	I	105	69	i
10	0A	Line feed	LF	CTRL-J	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	VT	CTRL-K	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	FF	CTRL-L	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage feed	CR	CTRL-M	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	SO	CTRL-N	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	SI	CTRL-O	47	2F	/	79	4F	O	111	6F	o
16	10	Data line escape	DLE	CTRL-P	48	30	0	80	50	P	112	70	p
17	11	Device control 1	DC1	CTRL-Q	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	DC2	CTRL-R	50	32	2	82	52	R	114	72	r
19	13	Device control 3	DC3	CTRL-S	51	33	3	83	53	S	115	73	s
20	14	Device control 4	DC4	CTRL-T	52	34	4	84	54	T	116	74	t
21	15	Neg acknowledge	NAK	CTRL-U	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	SYN	CTRL-V	54	36	6	86	56	V	118	76	v
23	17	End of xmit block	ETB	CTRL-W	55	37	7	87	57	W	119	77	w
24	18	Cancel	CAN	CTRL-X	56	38	8	88	58	X	120	78	x
25	19	End of medium	EM	CTRL-Y	57	39	9	89	59	Y	121	79	y
26	1A	Substitute	SUB	CTRL-Z	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	ESC	CTRL-[	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	FS	CTRL-\	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	GS	CTRL-]	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	RS	CTRL-^	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	US	CTRL-_	63	3F	?	95	5F	_	127	7F	DEL

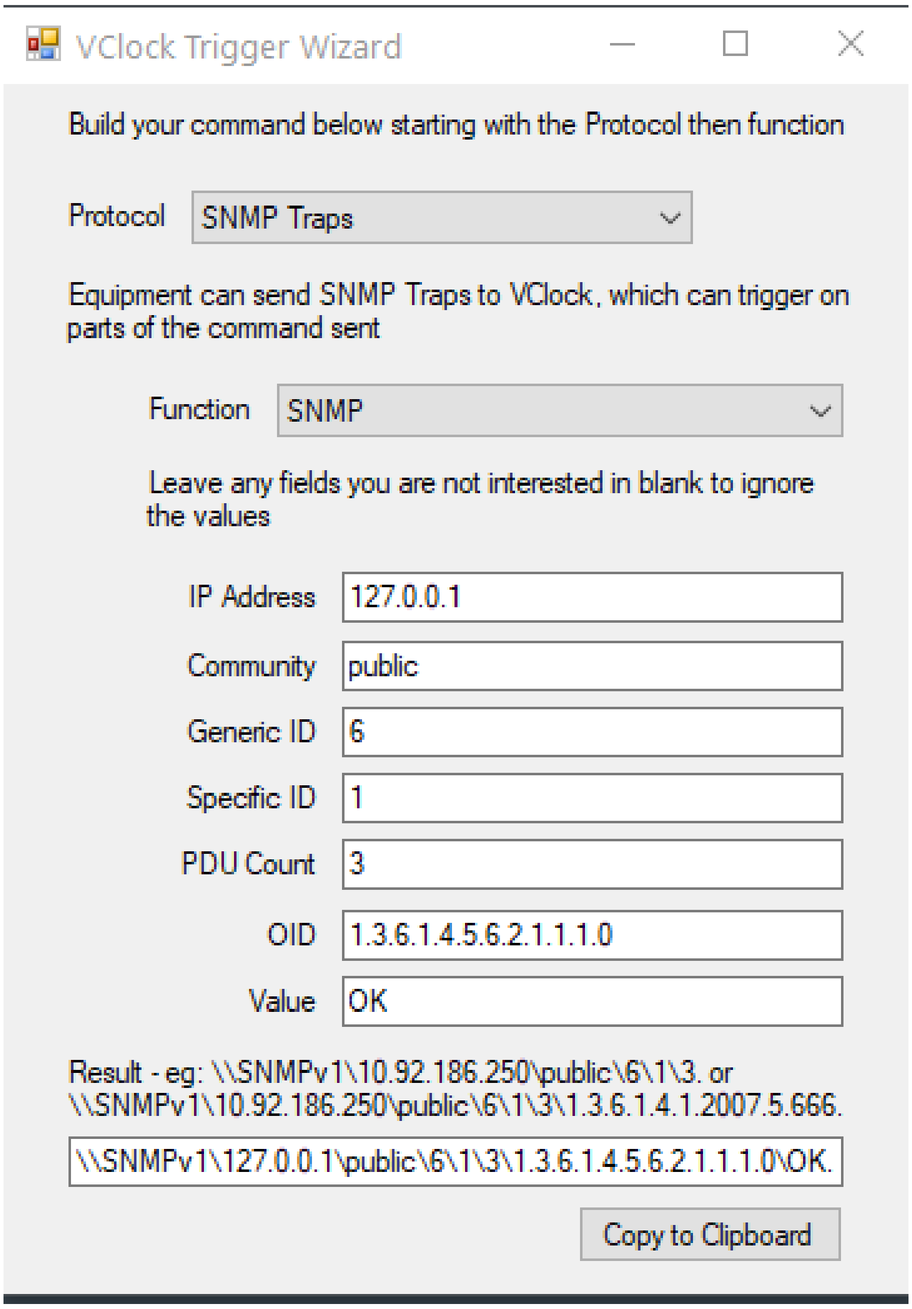
Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
128	80	Ç	160	A0	á	192	C0	└	224	E0	α
129	81	ü	161	A1	í	193	C1	┴	225	E1	ß
130	82	é	162	A2	ó	194	C2	┬	226	E2	Γ
131	83	â	163	A3	ú	195	C3	├	227	E3	π
132	84	ä	164	A4	ñ	196	C4	─	228	E4	Σ
133	85	à	165	A5	Ñ	197	C5	┼	229	E5	σ
134	86	å	166	A6	ª	198	C6	╞	230	E6	µ
135	87	ç	167	A7	º	199	C7	╟	231	E7	τ
136	88	ê	168	A8	¿	200	C8	╚	232	E8	Φ
137	89	ë	169	A9	⌐	201	C9	╔	233	E9	Θ
138	8A	è	170	AA	¬	202	CA	╩	234	EA	Ω
139	8B	ï	171	AB	½	203	CB	╦	235	EB	δ
140	8C	î	172	AC	¼	204	CC	╠	236	EC	∞
141	8D	ì	173	AD	¡	205	CD	═	237	ED	φ
142	8E	Ä	174	AE	«	206	CE	╬	238	EE	ε
143	8F	Å	175	AF	»	207	CF	╧	239	EF	∩
144	90	É	176	B0	░	208	D0	╨	240	F0	≡
145	91	æ	177	B1	▒	209	D1	╤	241	F1	±
146	92	Æ	178	B2	▓	210	D2	╥	242	F2	≥
147	93	ô	179	B3	│	211	D3	╙	243	F3	≤
148	94	ö	180	B4	┤	212	D4	Ö	244	F4	⌠
149	95	ò	181	B5	╡	213	D5	╒	245	F5	⌡
150	96	û	182	B6	╢	214	D6	╓	246	F6	÷
151	97	ù	183	B7	╖	215	D7	╫	247	F7	≈
152	98	ÿ	184	B8	╕	216	D8	╪	248	F8	≈
153	99	Ö	185	B9	╣	217	D9	┘	249	F9	·
154	9A	Ü	186	BA	║	218	DA	┌	250	FA	·
155	9B	¢	187	BB	╗	219	DB	█	251	FB	√
156	9C	£	188	BC	╝	220	DC	▄	252	FC	ⁿ
157	9D	¥	189	BD	╜	221	DD	▌	253	FD	²
158	9E	Pts	190	BE	╛	222	DE	▐	254	FE	■
159	9F	ƒ	191	BF	┐	223	DF	▀	255	FF	

Wizards

There are Wizards available to help you create Salvo Triggers and Commands. These are an aid to build correctly-formatted triggers/commands, but once you know the format, directly entering them into the Salvos table is also possible.

Trigger Wizard

There is a Wizard available to help create Triggers / remember the syntax. You can either click on the "Trigger Wizard" button above the "Serial & IP" column of the Salvos table (where you can then create a trigger and copy it to the clipboard, then paste into the relevant cell), or you can right click the cell you want to create a trigger in, and launch the Wizard from there. In the latter case, the existing cell value will update the contents of the Wizard (if formatted correctly!) and the cell will be auto-populated when you press save on the Wizard.



Build your command below starting with the Protocol then function

Protocol: SNMP Traps

Equipment can send SNMP Traps to VClock, which can trigger on parts of the command sent

Function: SNMP

Leave any fields you are not interested in blank to ignore the values

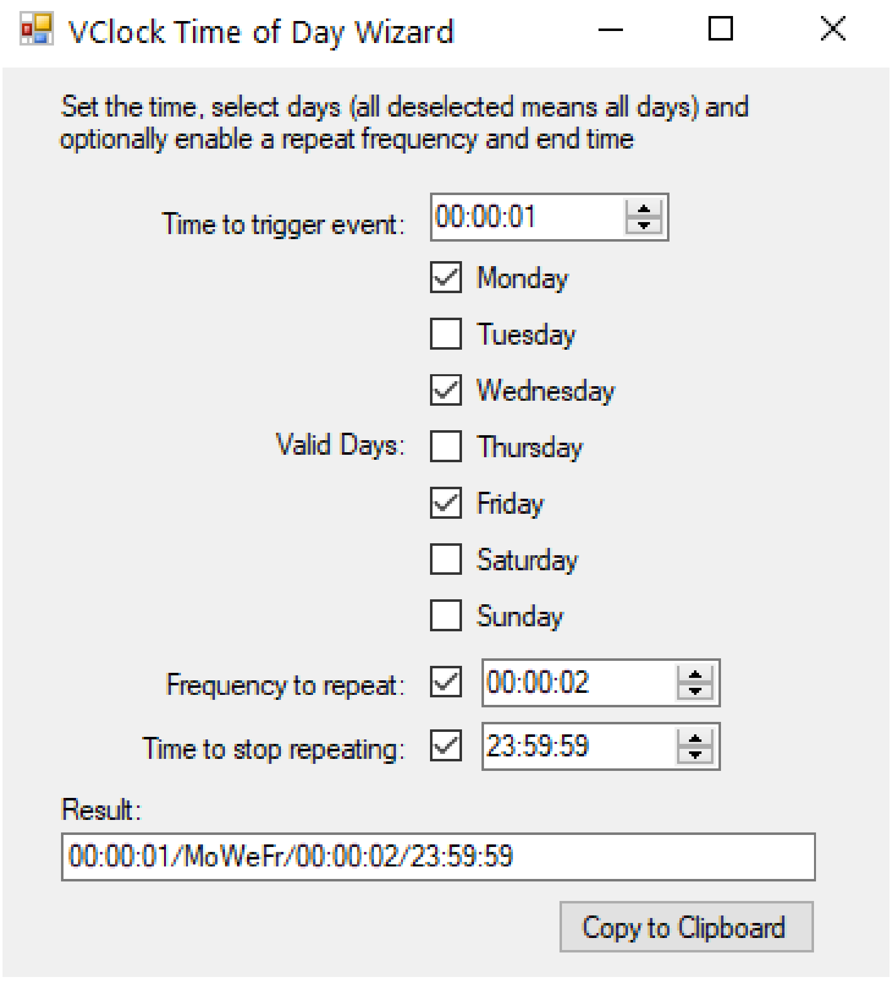
IP Address	127.0.0.1
Community	public
Generic ID	6
Specific ID	1
PDU Count	3
OID	1.3.6.1.4.5.6.2.1.1.1.0
Value	OK

Result - eg: \\SNMPv1\10.92.186.250\public\6\1\3. or \\SNMPv1\10.92.186.250\public\6\1\3\1.3.6.1.4.1.2007.5.666.

\\SNMPv1\127.0.0.1\public\6\1\3\1.3.6.1.4.5.6.2.1.1.1.0\OK.

Time Of Day Wizard

There is a Wizard available to help create Time Of Day triggers / remember the syntax. You can either click on the "T.O.D. Wizard" button above the "Time Of Day" column of the salvos table (where you can then create a trigger and copy it to the clipboard, then paste into the relevant cell), or you can right click the cell you want to create a trigger in, and launch the Wizard from there. In the latter case, the existing cell value will update the contents of the Wizard (if formatted correctly!) and the cell will be auto-populated when you press save on the Wizard.


VClock Time of Day Wizard

Set the time, select days (all deselected means all days) and optionally enable a repeat frequency and end time

Time to trigger event: 00:00:01

Valid Days: [✓ Monday] Tuesday [✓ Wednesday] Thursday [✓ Friday] Saturday Sunday

Frequency to repeat: [✓] 00:00:02

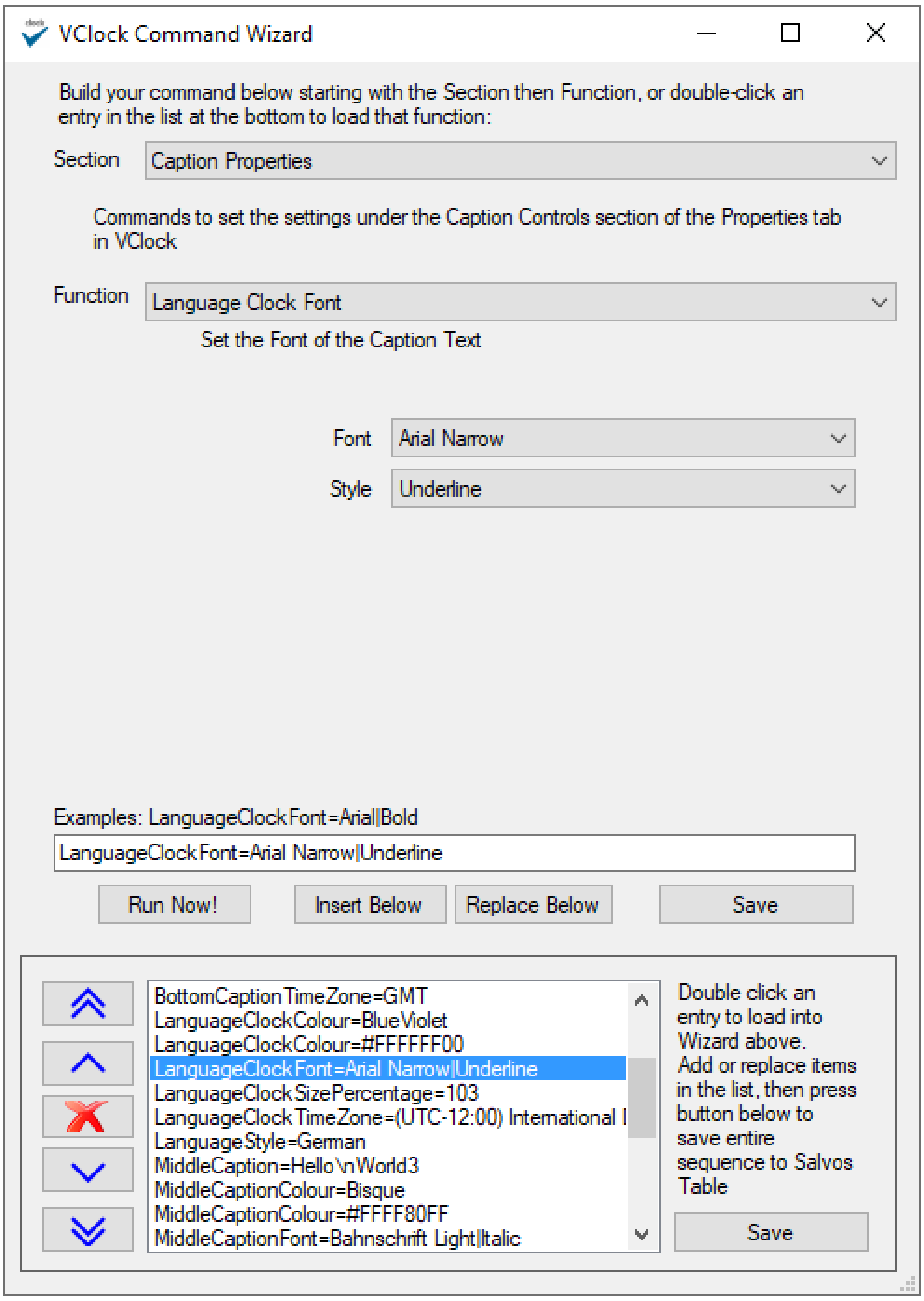
Time to stop repeating: [✓] 23:59:59

Result: 00:00:01/MoWeFr/00:00:02/23:59:59

Command Wizard

This is the most complex of the Wizards, as there can be many commands on a single Salvo.

You can either click on the "Command Wizard" button above the "Command To Trigger" column of the Salvos table (where you can then create one or more commands and copy them to the clipboard, then paste into the relevant cell), or you can right click the cell you want to create a trigger in, and launch the Wizard from there. In the latter case, the existing cell value will update the contents of the table at the bottom of the Wizard.



Build your command below starting with the Section then Function, or double-click an entry in the list at the bottom to load that function:

Section: Caption Properties

Commands to set the settings under the Caption Controls section of the Properties tab in VClock

Function: Language Clock Font

Set the Font of the Caption Text

Font: Arial Narrow

Style: Underline

Examples: LanguageClockFont=Arial|Bold

```
LanguageClockFont=Arial Narrow|Underline
```

```
BottomCaptionTimeZone=GMT
LanguageClockColour=BlueViolet
LanguageClockColour=#FFFFFF00
LanguageClockFont=Arial Narrow|Underline
LanguageClockSizePercentage=103
LanguageClockTimeZone=(UTC-12:00) International I
LanguageStyle=German
MiddleCaption=Hello\nWorld3
MiddleCaptionColour=Bisque
MiddleCaptionColour=#FFFF80FF
MiddleCaptionFont=Bahnschrift Light|Italic
```

Double click an entry to load into Wizard above.
Add or replace items in the list, then press button below to save entire sequence to Salvos Table

Select a section and function, fill in the details then you can save that single command back to the Salvos table (or copy to the clipboard). Or add the command to the list at the bottom of the page, build up a set of commands in that list, then save the entire list back to the Salvos table (or copy to the clipboard).

If you want to test a command you can click the "Run Now" button and see the results on the main clock screen. There are buttons to insert/replace items in the list, move items up and down or delete them.

For many text fields, you can right-click and insert from a list of standard VClock #variables.

There are limitations to the Wizard and some shortcuts that you can do manually, can't be created in the Wizard. For example:

- Using + in a DMX command (DMX=1:255,255,0+11:255,255,0)
- Using + in a LWMIX command (LWMIX=1,3,ON+2,4,ON)
- Using multiple ranges for multiple lamps (Lamps[1-3,5-7]Caption=Hello), GPIs (GPIs=[1-3,5-7]H) and GPOs (GPOs=[1-3,5-7]H)
- Using %GETMEM in some situations – currently only allows you to insert into text strings but the memory slots can be used in other situations.

Special Salvo Commands

There are many commands that can be sent, but that are not really properties of the clock (to control external equipment, etc). Commands in red require a "VClock Plus" license.

Internal VClock Commands

APPENDSALVOS=<filename> is the same as SALVOS= above but will not replace the existing salvos. Useful if you have several configurations which have lots of shared salvos but slight differences (such as which lamp to light for a particular GPI number). SALVOS= and APPENDSALVOS= can be used in the same command (but only that way around for obvious reasons).

CONFIG= <commaSeperatedSettings> This gives the ability to change settings on the config pages on the fly, until VClock is restarted. Only IPClient connections are available currently. 1 or more of the following parameters can be changed:

```
CONFIG=IPCLIENT1,IP=192.168.1.1,Port=1234,UniqueID=wibble,Enabled=True
```

IPCLIENT is the only supported setting currently. The "1" on the end of IPCLIENT tells it to edit the first one in the list. The rest are obvious. Note that the settings are not saved if VClock is restarted unless the user visits the Config page and presses save.

Audio=<filename.wav> will play a sound file. Useful as an alert when a warning lamp is lit for example. Only wav files are supported.

AudioLoop=<filename.wav> is similar to Audio= but will loop the file continually until the playback is cancelled with Audio= (ie no filename).

ClockCounterReset= will reset the counter on the clock caption, if used. This can also be done by updating the Clock Caption and including %CAPTION in the text.

File=<filename> loads the contents of a config file. For example File=Example1.txt will load one of the shipped example config files.

GPI=<number><state> This would usually be sent as a result of a Salvo received via IP or Serial, to convert to a GPI which can then be relayed to other clocks.

Format is GPI=<number><state>, where state is H (high), L (low), P (pulse), ! (invert), F (flash) or T (Telephone). For example GPI=1H would turn GPI 1 on. GPI=1! would invert whatever state GPI 1 is currently in. This can be particularly useful as the action of a "lamp" press (Lamp1MouseDown), to toggle the state of the GPI. The GPI value (high or low) can then be used to send commands and change the caption of the lamp to indicate the current state.

Flash and Telephone repeat patterns similar to OnOff and Phone on a lamp. Pulse will gererate an approximately 250ms pulse going high, then return to low.

GPIs=[<range>]<state> It is possible to change several GPIs at the same time by using a range. The range can actually be a comma separated list of several ranges, in the format:

GPIs=[1-3,5,7-8]H

This would turn on GPIs 1, 2, 3, 5, 7 and 8. States can be the same as the GPI= command (L H P ! F T).

LampXCounterReset= allows you to reset the counter for each lamp (replace X with lamp number). It gets reset automatically when the lamp changes state, but this is a way to reset it without changing the lamps state. (the = is important).

LogToFile=<string> The <string> will be written to the standard log file, as specified in the "Logging Settings" section of the Clock properties.

REPROCESSGPIS= will act as if any GPI that is on has just been triggered. Useful mainly for demos where we want to reload layouts but retain mic lives etc.

RESTARTVCLOCK= will close and re-launch VClock. Note if debug is enabled, this will fail. All current states of GPIs etc will be lost.

RUNNINGMODE=True|False will switch between running mode and config mode screens – useful if you want to ensure it is back in running mode before a show.

SALVOS=<filename> allows you to load a different set of Salvos from the default SALVOS.TXT. It isn't saved after a restart of the app - it reverts to SALVOS.txt.

SPEAK=Words to speak will convert text to speech and play it out of the default sound device. Can also use variables. See "Text to Speech" section for more detail.

StopwatchSet=hh:mm:ss allows you to set the Stopwatch to a specific value – useful for a programme timer where you want to record "as-live" and show the time the show will air. Similarly **UpcounterSet=** and **DownCounterSet=** will set the value of those counters.

StopwatchInc=hh:mm:ss increments the Stopwatch by a specified value. Similarly **UpcounterInc=** and **DowncounterInc=** will increment the value of those counters. **StopwatchDec=**, **UpcounterDec=** and **DownCounterDec=** will decrement them.

StopwatchState=START|STOP|FREEZE|RESET control the main Stopwatch counter (if enabled). You can use the same commands for the **UpCounterState=** and **DownCounterState=** (which will link to the Fusion timers if connected via the LWCP+TIMER protocol).

SWAPLAMPS=<n>[-m],<i> will swap all of the properties of the first lamp number with the 2nd. Editing the file directly to do this can be laborious. Similarly, **COPYLAMPS=** will copy the contents of 1 to the other, but leaving the original as it was. You can specify a range of lamps, eg SWAPLAMPS=1-4,5 will swap lamp 1 with 5, 2 with 6, 3 with 7 and 4 with 8.

TopCounterReset=, MiddleCounterReset= and **BottomCounterReset=** reset the counters on the Top/Middle/Bottom captions. Changing the caption also resets these.

TopMost=TRUE|FALSE will override the "Always on top" setting on the settings page. Useful to bring the clock in focus if a Salvo is triggered for instance to ensure it is seen.

VLCxCONTROL=[PLAY|PAUSE|STOP] controls the VLC Players (x is replaced with 1-3).

WEBBROWSERREFRESH= will tell the page in the browser to refresh. The = is important for the command to be processed.

External Commands

BARIX=<output>,<state> (or **BARIX~UNIQUEID=<output>,<state>**) This turns on/off GPO outputs on the Barix Barionet 50 Network device. Output is a pin number (1-4). State is 1 (or H), or 0 (or L).

BTA=[<deviceid>,]<output>,<input>,<state>
(or **BTA~UNIQUEID=[<deviceid>,]<output>,<input>,<state>**) This turns on/off Audio crosspoints on the Broadcast Tools ACS 8.2C device. This can either be connected to the serial port, or as an IPClient using an IP to serial device. Device ID is 0-3 (it can be omitted and defaults to 0, the "main" unit). Output is the output number (1-2). Input is 1-8 and State ON or OFF. When turned on, the audio from that input is MIXED into the output specified. UniqueID can be of an IPClient connection, or SERIAL for the Serial Port.

BTAS=<deviceid>,<output>,<input>,<state>
(or **BTAS~UNIQUEID=[<deviceid>,]<output>,<input>,<state>**) Similar to BTA= and the same command format, but this forces OFF all of the other inputs to the specified output. So used if you want to select only 1 of the inputs to an output at a time. UniqueID can be of an IPClient connection, or SERIAL for the Serial Port.

BTO=[<deviceid>,]<output>,<state>
(or **BTO~UNIQUEID=[<deviceid>,]<output>,<state>**) This turns on/off GPO outputs on the Broadcast Tools SRC-8/16/32 and ACS 8.2C devices. These can either be connected to the serial port, or as an IPClient using an IP to serial device. Device ID is 0-3 (it can be omitted and defaults to 0, the "main" unit). Output is a pin number (1-32). State is H (High), L (Low), or P (300ms Pulse). UniqueID can be of an IPClient connection, or SERIAL for the Serial Port.

BMROUTE=<output>,<input> This Is support for the Blackmagic Smart Videohub range of video routers. This will set the output to switch to the selected input (starts at input/output 1).

CLOUDCAST=<function>   This is to control the Cloudcast Broadcast Delay Service. Function can be: BUILD, EXIT, DUMP, DUMPALL, COUGHSTART, COUGHEND. The first 4 are just triggers, the cough function is active whilst in the start state, until the end state (so you would assign START to LampXMouseDown and END to LampXMouseUp for example). The IPClient connection to the delay unit must have CLOUDCAST set as the protocol, and the Unique ID must match the Delay unit's ID.

CommandLine=<command> allows you to run an external command when a trigger is received. For example "CommandLine=Notepad defaults.txt" would open defaults.txt in Notepad, or "Commandline=cmd /c copy c:\file1.txt c:\file2.txt" would do a file copy. Useful if you want to trigger an external app to send an email, or fire off some other external alert.

CSVEntry=<filePATH>, <contents> Very similar to WriteToFile, but the filename is always assumed to be YYYYMMDD.LOG, so only the path is specified. For example:

    CSVEntry=c:\,%LAMP1START,%NOW,%LAMP1DUR,CSVtest

… would write a file C:\20210101.LOG (for 1st Jan 2021) and the file would contain a comma separated list of the Lamp1start value, the time now, the duration Lamp1 has been in that state, and the string CSVtest.

CTP=A1,D3,ON (or **CTP~UniqueID=A1,D3,ON**) sets the crosspoints on a CTP Systems DIO800R Dante Audio Switcher.

(A1 is the output, D3 is the input, ON is the state. The UniqueID will send it to a single IPClient otherwise it will be sent to ALL IPClients).

Valid inputs/outputs are A1-A8 (analogue) or D1-D16 (Dante).

Valid states are ON, OFF or special states are:

> ONS/OFFS (These options automatically turn on/off the stereo pair of the crosspoint that you specify. So A1,D3,ONS would turn on A1,D3 AND A2,D4)
>
> ONM/OFFM (turn on/off a mono source to a stereo output. So A1,D3,ONM would turn on A1,D3 and A2,D3).

You can combine multiple commands with +, for example:

    CTP=A1,D3,ON+A2,D4,ON+D11,D13,OFF+D12,D14,OFF

Note that due to a limitation of the switcher, sending CTP=A1,D3,ON;CTP=A2,D4,ON on a single Salvo is likely to lead to unreliable results. Using the + option above to send many crosspoints in a single command is preferred.

DMX

DMX has been tested against the following device, but the Art-Net protocol in particular is supported on a wide range of hardware.

https://www.enttec.com/products/controls/dmx-ethernet-lighting-control/ode-mk2-open-dmx-ethernet/

DMX=<startChannel>:<value>[,<value2>…<value*n*>][/nnn] When DMX is enabled on the DMX tab of the settings page, this allows you to set 1 or more DMX values. Multiple commands can be separated with a **+**. Examples are

    DMX=1:255 (to set a single byte at position 1 to 255)
    DMX=10:255,255,255 (to set an RGB lamp at address 10 to all on (white))
    DMX=1:255+10:255,255,255 (to set both of the above)

The value after the / is optional on the end of the command, and signifies over what time period (in approximate mS) to make the transition. Default is 0 (immediate). So:

> DMX=1:255/1000 (to set a single byte at position 1 to 255, fading up over 1 Second)
> DMX=10:255,255,255/500 (to set an RGB lamp at address 10 to all on (white) over ½ a second)
> DMX=1:255/1000+10:255,255,255/500 (to do both of the above)

DMX=<startChannel>:Colour=<ColourName>[/nnn] Allows you to specify a standard name of a colour, which will be converted to its RGB values and written to the start channel address and the next 2. Multiple commands can be separated with a **+**. For example:

> DMX=10:Red (would be the equivalent of DMX=10:255,0,0)
> DMX=10:Red/500 (the same, fading to that value over ½ a second)
> DMX=10:Red/500+20:Blue (the above plus setting channel 20/21/22 to Blue)

The value after the / is optional on the end of the command, and signifies over what time period (in approximate mS) to make the transition. Default is 0 (immediate).

DMX=ALLON[/nnn], DMX=ALLOFF[/nnn] Turn all 512 DMX values either on or off. The value after the / is the time in mS to make the transition. When not specified the default is 0 (immediate).

DMX=%GETMEM(S1Default) DMX can use Memory Slots – an example would be similar to DMX=Idle but when you have more than 1 studio to remember a default for. It can either contain the entire DMX command, or a part of it. All of these would do the same thing:

SETMEM=S1Default,Blue	…	DMX=1:Colour=%GETMEM(S1Default)/500
SETMEM=S1Default,Blue/500	…	DMX=1:Colour=%GETMEM(S1Default)
SETMEM=S1Default,1:Blue/500	…	DMX=%GETMEM(S1Default)

DMX=Idle Resets the lights to the current "idle" values as defined by the DMXIdleState property. This does NOT accept a value after a / for the transition period, this would be done on the individual commands in DMXIdleState. Not really required now memory slots are supported.

EMAIL=to=<recipientslistCommaSeparated>&cc=<recipientslistCommaSeparated>&bcc=<recipientslistCommaSeparated>&urgent&subject=<emailsubject>&message=<emailmessage>

This allows VClock to send an email, using the account details specified on the settings page. At least 1 of *to*, *cc* or *bcc* needs to be specified. The rest are all optional (though a subject and message are usually preferred!). The *urgent* setting can be omitted to send a standard priority message. For example:

EMAIL=to=example@voceware.co.uk&subject=Silence Detected&message=Silence is detected on FM

EMBER+=//path/to/ember+/parameter:<value> will set the value for an Ember+ parameter, using the Ember+ connection defined in the Tools/Settings / IP Settings page.

EMBERPLUS= does the same thing, as does **EMBER =** (space instead of +). This is because sending a + as a http post into VClock (Command=Ember+=//path....) will result in VClock receiving a space not a +.

GPO=<number><state> This turns physical GPO (Outputs) on/off on an Advantech or XKeys card (it does not work with Game ports). For example to turn an external lamp on when VClock receives a command to tell it the mic is live or that the studio is live. Format is GPO=<number><state>, where state is H (high) or L (low). For example

GPO=1H

…would turn GP Output 1 on.

GPOs=[<range>]<state> It is possible to change several GPOs at the same time by using a range. The range can actually be a comma separated list of several ranges, in the format:

GPOs=[1-3,5,7-8]H

This would turn on GPOs 1, 2, 3, 5, 7 and 8. States can be the same as the GPO= command (L or H).

HTTP=<url> will do a HTTP "GET" to an address (like hitting it with a web browser). For example :

HTTP=http://localhost:18513/VClock?Command=Lamp1State=On

Any information returned will be processed for salvo matches. For example an RDS Json file may be polled using a Time Of Day event in VClock. The resulting JSON can be parsed to find a text match:

```
{
        "pi":"663C",
        "call":"WXYZ",
        "rt":"More Music Soon",
        "rtp1":"",
}
```

A salvo of

"rt":"

… would find the start of the radiotext. Then a command of

TopCaption=%BUFFERTO"%

… would use all of the data up to the next quotation mark, resulting in

TopCaption=More Music Soon

IPSERVER=, **IPCLIENT=** (or **IPCLIENT~UNIQUEID=**) and **SERIAL=** will send whatever data follows to the respective connections. If specified the UniqueID tells VClock which IPClient to send the command to. If omitted, we send to all IPClients. IPSERVER will also look up variables, so you can send commands such as IPSERVER=SETMEM=FRED,%GETVAR(FRED) to relay a memory slot name to other clocks.

LWA=5,21604 (or **LWA~UNIQUEID=5,21604**) is an extension of IPCLIENT and will send an LWRP command to an Axia Node or endpoint, telling it to change the Livewire channel. The first parameter is the DST number, the 2nd is the Livewire Channel (21604 is the FromSource direction, -21604 is the ToSource return direction). It will also accept a multicast address (LWA=5,239.192.84.100) You can also specify a new name for the Destination as a 3rd parameter.

LWCPBUTTON= or LWCPBUTTON~UNIQUEID=<module>,<button>,<command> sends the command to the button panel on an Axia Mix Engine surface (Fusion). Valid commands are:

IND=OFF | ON | FLASH | FLASH_ULTRA | FLASH_FAST | FLASH_MEDIUM | FLASH_SLOW | WINK | FLASH_1000 | FLASH_500 | FLASH_250 | FLASH_125 | FLASH_375_125 | DBL_WINK_125
TEXT="Wibble\nWobble"
BACKCOLORON=[255,0,0] | [0xFF,0x00,0x00] | Colour
BACKCOLOROFF=[255,0,0] | [0xFF,0x00,0x00] | Colour

The BACKCOLOR commands for Fusion can take some basic colour names (Red, Green, Blue, etc). Not all colours are supported, so best to stick with the RGB arrays, either in decimal or Hex as above).

LWCPBUT= or LWCPBUT~UNIQUEID does the same thing as LWCPBUTTON
For example LWCPBUT=7,10,IND=ON (turn on the lamp on button 10 of module 7).

LWCPIQ= or LWCPIQ~UNIQUEID=<object>,<property>,<command> sends the command to the Axia IQ, IQX or IQX. Valid commands include (but any command can be sent using the above format):

LWCPIQ=crmo,ssel,1
LWCPIQ~St1=stmo,ssel,2

(Control Room or Studio Monitoring, Source Selection, 1-4 = PGM1-PGM4, 7=EXT1, 8=EXT2)

LWCPMONBUTTON= or LWCPMONBUTTON~UNIQUEID=<button>,<command> sends the command to one of the 4 buttons on the monitor panel on an Axia Mix Engine surface (Fusion or Quasar). Valid commands are:

> IND=OFF | ON | FLASH | FLASH_ULTRA | FLASH_FAST | FLASH_MEDIUM | FLASH_SLOW | WINK | FLASH_1000 | FLASH_500 | FLASH_250 | FLASH_125 | FLASH_375_125 | DBL_WINK_125
> TEXT="Wibble\nWobble"
> BACKCOLORON=[255,0,0] | [0xFF,0x00,0x00] | Colour (not Quasar)
> BACKCOLOROFF=[255,0,0] | [0xFF,0x00,0x00] | Colour (BackColorOff not supported by Quasar)
>
> The BACKCOLOR commands for Fusion can take some basic colour names (Red, Green, Blue, etc). The Quasar does not support this. So best to stick with the RGB arrays, either in decimal or Hex as above). Quasar also doesn't support the BACKCOLOROFF command, off is always unlit.

LWCPMONBUT= or LWCPMONBUT~UNIQUEID does the same thing as LWCPMONBUTTON. For example:

```
LWCPMONBUT=4,IND=ON
```

(turn on the lamp on button 4).

LWCPVMIX= or LWCPVMIX~UNIQUEID=<output>,<input>,<state> will set/clear an individual VMix input on an Axia Mix Engine. VMIX= and VMIX~UNIQUEID= also work for backwards compatibility. For example

```
VMIX=1,3,ON
LWCPVMIX~UniqueID=1,3,ON
```

LWI= does the same as LWO= (see later), but controls the Axia GPI. This isn't possible on hardware ports but it is on the Axia PC Driver. If you set the GPIO Livewire address on a port of the Axia Driver to match that of the VMix or Record button on a QoR for example, you can control it by sending the LWI= command to the Axia PC Driver. This causes the Driver to send a multicast GPI event to the Qor. But VClock can send Multicast Livewire directly see….

LWMC=[C|G,]LwCh,I|O,Pin,State,[PulseDuration] will send Livewire Multicast GPIO to devices. Multicast GPIO can be sent as a "Console", or as a "GPIO Device". Only consoles can send commands to GPIO devices (for example XNodes) and vice-versa. An example is:

```
LWMC=C,12345,O,1,L,250
```

C/G is Console/GPIO device (that VClock is emulating). Console is assumed if it is omitted.

LwCh is the multicast channel number

I/O is Input (GPI) or Output (GPO). Output is assumed if it is omitted.

Pin is the pin number (1 to 5)

State is L – Low (on) / H – High (off) / PH – Pulse High / P or PL – Pulse Low

If set to a pulse, the default duration is 250mS. This can be changed with the PulseDuration value, which can be 10-630mS in 10mS increments, or 250-7750mS in 250mS increments. Setting other values will round to the nearest lower value. Setting more than 7.75 seconds will result in a 7.75 second pulse.

LWMIX=1,3,ON (or **LWMIX~UniqueID=1,3,ON**) sets the crosspoints on the mixer section of an Axia XNode, using the LWRP protocol.

(1 is the output, 3 is the input, ON is the state. Again the UniqueID will send it to a single IPClient otherwise it will be sent to ALL IPClients).

Valid states are ON, OFF or a db value in 1/10th of a db. Other special states are:

> ONS/OFFS (These options automatically turn on/off the stereo pair of the crosspoint that you specify. So 1,3,ONS would turn on 1,3 AND 2,4)
>
> ONM/OFFM (turn on/off a mono source to a stereo output. So 1,3,ONM would turn on 1,3 and 2,3).

You can combine multiple commands with +, for example:

> LWMIX=1,3,ON+2,4,ON+11,13,OFF+12,14,OFF

LWO=1,xxhxx,<dur> (or LWO~UNIQUEID=1,xxhxx,[PulseDuration]) is an extension of IPCLIENT and will send a GPO command to an Axia Xnode connected to the IPClient on port 93, or to the IPServer if the port is set to 93 and "Advertise as Livewire GPIO Device" is checked.

The number following is the GPO number, then the mask (x is ignore, l is low (on) and h is high (off). Finally, a duration (in mS) can optionally be provided, which will generate a pulse for that duration on the XNode pin.

UniqueID can be that of an IPClient connection, or IPSERVER to send to the IPServer only.

MIDI=ControlChange,<Channel>,<Controller>,<Value> will send a control command out to a Midi device.

Channel can be 1-16

Controller may be a known enum such as BankSelect,or can be the numeric value (0-127)

Value can be 0-127, OFF (equates to 0) or ON (equates to 127).

See also RODE= for a simplified list of MIDI commands for the RØDECaster console.

MIDI=NoteOn|NoteOff,<Channel>,<Note>,<Velocity> will play (or stop) a note on a Midi device.

Channel can be 1-16.

Notes equate from 0 (C0) to 127 (G10). VClock can accept either format and can use the usual flat notes (Ab9) or the sharp notes that Midi uses by default (G#9).

Velocity can be 0-127.

PDM=<command>,<state> will send a command to a Telos 25-Seven PDM unit. These emulate the buttons on the front of the unit.

Valid commands are BUILD EXIT COUGH DUMP and BYPASS

Valid states are TRIGGER DOWN and UP

TRIGGER is a momentary press of the button. Sometimes you want to hold the button though, for example whilst coughing. In this case, you would assign

Lamp1MouseDown → PDM=COUGH,DOWN
Lamp1MouseUp → COUGH,UP

RODE= Is a simplified set of Midi commands, specifically for the RØDECaster console. See the RØDECaster section in Appendix D for more details.

SAS=<command> will send a raw command to an SAS console

SASMODULE=<consoleID>,<source>,<state> will set the sate of the source on the console specified to (ON | OFF | CUEON | CUEOFF).

SASOPTO=<number>,<state> will set the specified Opto to the state (ON | OFF). It can also be set to **?** to request the current state of the Opto.

SASRELAY=<number>,<state> will set the specified relay to the state (ON | OFF | PULSE). It can also be set to **?** to request the current state of the relay.

SASSALVO=<number> will trigger the specified Salvo

SASXPOINT=<destination> or SASXPOINT=<destination>,? will query the current source assigned to the specified destination. You would then trigger on this response using the SASXPOINT: trigger.

SASXPOINT=<dest>,<source> will assign the specified source to the destination.

SAVESTREAMDECKIMAGES= is a helper function that will generate a BMP file for any lamps set up to be displayed on a StreamDeck. They are saved in the VClock ProgramData folder as STREAMDECKKEY-xx.BMP where xx is the key number.

STREAMDECK=<buttonNo>,<listofCommandsSeparatedWith&> where…

> BackColour=<colour> (or BC=)
> Image=<filepath> (or I=)
> ImagePosition=<TOP|MIDDLE|BOTTOM> (or IP=T|M|B)
> Text=<message> (or T=)
> TextColour=<colour> (or TC=)
> TextPosition=<TOP|MIDDLE|BOTTOM> (or TP=T|M|B)
> TextSize=<size> (or TS=)
> TextFont=<font name> (or TF=)
> TextStyle=<NORMAL|BOLD|ITALIC> (or TS=N|B|I)
> Clear (clears button - overrides any of the above)
> ClearAll (clears all buttons, ButtonNo is not used but a valid one needs to be specified - overrides any of the above)

For example:

> STREAMDECK=1,Image=vclock.png&ImagePosition=Top&Text=VCLOCK&TextPosition=Bottom

Or, in short form:

> STREAMDECK=1,I=vclock.png&IP=T&T=VCLOCK&TP=B

This allows VClock to control the look of a StreamDeck button manually, sending it a combination of background colour, image and text.

Another, more powerful way to control a StreamDeck is to use the LampXStreamDeckKey= setting to mirror an on-screen Lamp with a StreamDeck key. Then you can disable the on-screen version with LampXStreamDeckMode=StreamDeckOnly.

UDP=<ipaddress>,<port>,<message> will send UDP to the specified IP Address and Port. It will also look up variables for the message.

VMIX=<inputNumber>,<command> will control VMix Video Mixing software. Valid commands are LIVE|PREVIEW|OVERLAYxON|OVERLAYxOFF (x can be 1-4). We can also trigger VMix Scripts with STARTSCRIPT|STOPSCRIPT. See VMix section in Appendix D for more details.

WHEAT=<slio><state> or **WHEAT~UNIQUEID**=**<slio><state>** This triggers SLIOs on a Wheatstone blade connected via the IP Client. Slio is the SLIO number, state is H (high) or L (low). If specified the UniqueID tells VClock which IPClient to send the command to. If omitted, we send to all IPClients.

WriteToFile=<filename>,<contents> The filename and contents section can each contain "VClock variables". This appends to the end of the file if the file already exists. For example:

```
WriteToFile=c:\%Y%M%D.LOG,%LAMP1START,%NOW,%LAMP1DUR,Wibble
```

XKEYSLAMPFLASHRATE=<rate> will set the frequency of the flash when a lamp is set to flash. It is a global setting. Values are 1 (very fast) to 255 (approx once every 4 seconds).

XKEYSLAMPINTENSITY=<bank1Intensity>[,<bank2Intensity>] will set the intensity (brightness) of all lamps. There are 2 banks of lamps on some XKeys devices, blue and red. Values are 0 (off) to 255 (brightest). If only 1 value is set, it will be used for both banks.

XKEYSLAMPSTATE=<lampNumber>,<state> will set the state of a lamp on an XKeys device such as an XK-24. LampNumber starts at 1.

For single colour devices, valid states are ON, OFF, FLASH.

There are 2 colours (blue and red) on some devices. Blue is the first colour and will be controlled by the above states. Better states are:

BLUE (turns on blue / off red)
RED (turns on red / off blue)
BLUEFLASH (flashes blue / off red)
REDFLASH (flashes red / off blue)
BLUEFLASHREDON (flashes blue, with red solid on)
REDFLASHBLUEON (flashed red, with blue solid on)
BOTH (turns blue and red on)
BOTHFLASH (flashes blue and red)
OFF (blue and red both off)

ON and FLASH will control the BLUE lamp on devices with 2 colours.

See the XKeys section of Appendix D for more detail.

XKEYSLAMPSTATEs=[<range>],<state> It is possible to change several XKeys Lamps at the same time by using a range. The range can actually be a comma separated list of several ranges, in the format:

```
XKEYSLAMPSTATEs=[1-3,5,7-8]H
```

This would turn on Lamps 1, 2, 3, 5, 7 and 8. States can be the same as the XKEYSLAMPSTATE= command.

Special Salvo Triggers

There are potentially lots of complicated triggers coming in from equipment, and lots of internal VClock triggers to be remembered.

The Trigger Wizard helps you select the correct triggers and build up the correctly formatted command from the component parts. Or you can type them into the Salvos table directly. Case and spacing etc are all important, so I would encourage the use of the Trigger Wizard.

Below are the various triggers available, both for internally generated VClock triggers, and for external equipment.

Internal VClock Triggers

There are some salvos that get triggered internally by VClock. These are:

LampXMouseDown and **LampXMouseUp** are triggered if the user clicks on a lamp. You could, for example, turn the lamp on when the mouse is down and off when the mouse is up. A GPO could also be turned on/off to trigger a lamp on a slave clock to show the same indication as a tally.

Similarly, **ClockMouseDown/ClockMouseUp**, **TopCaptionMouseDown/TopCaptionMouseUp** and **BottomCaptionMouseDown/BottomCaptionMouseUp** can be used.

LampXMultiSalvoValue:y will be triggered whenever the MultiSalvo value changes to the value y. For example if Lamp1MultiSalvoMax was set to 3, and there were Salvos for:

Lamp1MultiSalvoValue:1 … Lamp1Caption=%date
Lamp1MultiSalvoValue:2 … Lamp1Caption = %english
Lamp1MultiSalvoValue:3 … Lamp1Caption=%time

…then the Lamp1MouseDown Salvo could be set to Lamp1MultiSalvoValue=INC and each time you clicked the lamp the caption of the lamp could be toggled between the different defined views. Or the Timezone could be changed for GMT and BST, etc.

LampXTimeout (where X is the lamp number from 1 to 32). This gets triggered if the LampTimeout value is set, after the time has elapsed when a lamp is turned on. It is used, for example, to extend a short pulse to light a lamp for a longer period of time (a doorbell for example).

NTPERROR is triggered if the NTP Server was unable to connect to one of the NTP Servers, or the 2 servers did not agree.

NTPSUCCESS is triggered when the time is in sync, or we successfully set the time to be in sync.

NTPWARNING is triggered when the time is not in sync, but we are not configured to set the time.

StopwatchValue=00:00:00 (00:00:00 can be any value) will get triggered when the stopwatch is running (or frozen) and the specified time is reached. This can be used to stop the stopwatch for example, or to flash a lamp with 10 seconds to go and with a 2nd salvo it could flash the lamp faster with 5 seconds to go, then turn it off and stop the stopwatch at 00:00:00.

Similarly **UpCounterValue**= and **DownCounterValue**= can be used.

UNIQUEID:INIT is triggered when an IPClient reconnects. The UNIQUEID is that of the IPClient connection in the Settings pages.

XKEYS:INIT is triggered when an XKeys device reconnects. Useful to set the XKEYSLAMPINTENSITY or XKEYSFLASHRATE values.

External Triggers

VClock can receive commands from just about anything and you can match in the Salvos table based on the raw text. But there are also lots of useful triggers that can be used to simplify this, or to match complicated incoming data.

For any IPClient connection, a Unique ID can also be specified to match against a specific connection. The protocol will also play a part, and so long as only 1 connection exists of a particular protocol (and there aren't any set to "Any"), the unique ID is not required.

For example:

```
 AEQ:MIC 1,ON
```

...would match against any IPClient connection with a protocol of "AEQ" or "ANY".

```
AEQ~STUDIO1:MIC 1,ON
```

...would only match the specific connection with the Unique ID of STUDIO1 set (and with a protocol of either AEQ or Any).

AEQ:<Channel>,<State> will trigger when an AEQ Console's specified channel (by source name) changes to the specified state (ON or OFF). For example:

```
AEQ:MIC 1,ON
AEQ:MIC 1,OFF
```

BARIX:<pin>,<state> will trigger when a Barix Barionet 50 GPI is changed. State can be ON or OFF. For example:

```
BARIX:3,ON
BARIX:3,OFF
```

BTI:[<deviceid>,]<pin>,<state> (or **BTI~UNIQUEID:[<deviceid>,]<pin>,<state>**) will trigger when a Broadcast Tools SRC-8/16/32 or ACS 8.2C GPI is changed. DeviceID is 0-3 (it can be omitted and defaults to 0 – the "main" device. Pin is 1-32, State can be H or L. For example:

```
BTI:0,32,H
BTI:32,L
```

UniqueID can be of an IPClient connection, or SERIAL for the Serial Port.

BMROUTE:<output>,<>input> will trigger when the specified output is switched to the specified input. These are both number-based, starting at 1. For example:

```
BMROUTE:1,12
```

… would trigger when output 1 is switched to input 12

You can also trigger when a route does NOT match, by prefixing with !. For example:

    BMROUTE:1,!12

… would trigger when output 1 is switched to any input other than 12.

CALENDARx:<state> will trigger when VClock is connected successfully to a provider. **BOOKED** will trigger when there is a booking at the current time, **AVAILABLE** will trigger when free. **ERROR** will trigger if you are successfully authenticated, but there is an issue retrieving the Calendar info. X is the calendar number (there can be more than 1 defined in a semicolon separated list in VClock's settings pages).

CALENDAR:ERROR (note no X) will trigger when VClock is attempting to connect to an Office 365 or Google Calendar and there is an authentication error.

CLOUDCAST:<state> triggers when a Cloudcast BDM enters the specified state. Valid states are:

    IDLE|BUILDING|SAFE|IN DELAY|EXITING | NOTSAFE

For example:

    CLOUDCAST:SAFE

… could be used to light a lamp to tell the user it is now safe to take a caller to air.

CTP:<output>,<input>,<state> will trigger when a CTP Systems DIO8008R Dante Audio Switcher crosspoint changes state. Inputs can be A1-A8 for analogue inputs, D1-D17 for Dante Inputs. Outputs A1-A8/D1-D16. State can be ON or OFF. For example:

    CTP:A1,D1,ON
    CTP:A1,D1,OFF

DHD:<LogicID>,<State> triggers when a DHD's Logic port changes to the specified state (ON or OFF). This uses the DHD External Control Protocol.

EMBER+://full/path/to/variable/name:<value> (or **EMBERPLUS:**) will trigger when an Ember+ endpoint changes to the specified value. It is also possible to use this value – see the Ember+ section in Appendix D for more info.

EMM88:<output>,<input>,<state> will trigger when an Audionics EMM88 crosspoint changes (output and input are numerical 1-8, state can be ON or OFF). For example:

    EMM88:1,4,ON

… would trigger when output 4 input 1 was turned on.

LAWO:<slot>,<lamp>,<state> will trigger when a specific slot/lamp changes to the specified state. The lamp can be OFF, or can be one of RED | GREEN | YELLOW | WHITE. For example:

```
LAWO:1,18,RED
LAWO:1,18,OFF
```

LWMC:<LWID>,<direction>,<pin>,<state> triggers when Livewire Multicast GPIO is picked up from the network and changes to the specified state (L – Low or H – High). Low is deemed as "on" in the Livewire world. For example:

```
LWMC:12345,O,1,L
```

… would trigger when Livewire ID 12345. Output put 1 went Low.

LWCPBUTTON:<moduleNumber>,<buttonNumber>,<state> (or **LWCPBUT:**) is triggered when a button is pressed on an Axia Element or Fusion console button panel. State can be DOWN/ON/TRUE or UP/OFF/FALSE. For example:

```
LWCPBUTTON=1,DOWN
LWCPBUTTON=1,UP
```

LWCPCRMUTE:<state> where state can be ON|MUTE|MUTED|TRUE or OFF|NORMAL|FALSE. This triggers when the Control Room logic of an Axia Console signals that the mics are / are not live. For example:

```
LWCPCRMUTE:MUTED
LWCPCRMUTE:NORMAL
```

LWCPIQ:<object>,<property>,<command> will trigger when an IQ/IQX/IQS console sends a matching command using its LWCP interface on port 4040 (a special version of LWCP aimed at console surface control). For example:

```
LWCPIQ:crmo,ssel,1
LWCPIQ:stmo,ssel,!1
```

... Control Room or Studio Monitoring, Source Selection, value = 1-4 (PGM1-4, 7 (EXT1) or 8 (EXT2). Preceding the value with ! will trigger if NOT set to that value.

Any valid LQCP IQ command can be sent using the above format, not just the common ones that are in the Wizards.

LWCPMONBUTTON:<buttonNumber>,<state> (or **LWCPMONBUT:**) will trigger when a user button is pressed on the monitoring panel of an Axia Element/Fusion console. State can be DOWN/ON/TRUE or UP/OFF/FALSE. For example:

LWCPMONBUTTON:1,DOWN
LWCPMONBUTTON:1,UP

LWCPPROFILE:<ProfileName> will trigger whenever a profile is changed on an Axia Console. If the name matches, the salvo is triggered. For example:

LWCPPROFILE:Default Show

LWCPSTMUTE:<state> where state can be ON | MUTE | MUTED | TRUE or OFF | NORMAL | FALSE. This triggers when the Studio logic of an Axia Console signals that the mics are / are not live. Note that the "Control Room" logic is usually more relevant for the room where the console is (see LWCPCRMUTE above). An example of the Studio command is:

LWCPSTMUTE:MUTED
LWCPSTMUTE:NORMAL

LWCPVMIX:<output>,<input>,<state> is triggered when an Axia Mix Engine's VMix input is turned on/off. An engine typically has 16 VMixes, each with 5 inputs. So to trigger on VMix 8, Input 3 being toggled:

LWCPVMIX:8,3,ON
LWCPVMIX:8,3,OFF

LWA:<destination>,<LWID> is triggered when an audio destination (on say an XNode) changes to the Livewire ID or multicast address specified. It is also possible to precede the Livewire ID with an ! to match when NOT that Livewire ID. For example:

LWA:5,12345 (would trigger when the Livewire ID is 12345)
LWA:5,239.192.48.57 (as above but using the multicast address)
LWA:5,!12345 (would trigger when the Livewire ID is anything BUT 12345)
LWA:5,-12345 (would trigger when the Livewire ID is the ToSource (backfeed) of 12345)

LWI:<GPIPort>,<mask> is triggered when a Livewire Input on an Axia GPIO XNode matches a specified mask. There are more details about the masks in Appendix D, but each GPI port is 5 pins, and each pin can be Off/High (H), On/Low (L) or ignored (x). So a mask for pin 3 Low would be "xxLxx". For example, for Port 5 pin 4 going on and off, the trigger would be:

LWI:5,xxxLx
LWI:5,xxxHx

LWMIX:<output>,<input>,<state> triggers when the specified crosspoint on an Axia XNode changes to the specified state. States can be ON or OFF. If the crosspoint is active at all, it is deemed "ON" (even if at -60 dbFS). For example:

LWMIX:9,1,ON
LWMIX:9,1,OFF

LWO:<GPIPort>,<mask> is triggered when a Livewire Output on an Axia GPIO XNode matches a specified mask. There are more details about the masks in Appendix D, but each GPI port is 5 pins, and each pin can be Off/High (H), On/Low (L) or ignored (x). So a mask for pin 3 Low would be "xxLxx". For example, for Port 5 pin 4 going on and off, the trigger would be:

LWO:5,xxxLx
LWO:5,xxxHx

MIDI:<MessageType>,<Channel>,<Controller|Note>,<Value> is triggered when a Midi command is received.

Channel can be 1-16.

Value can be 0-127, OFF (equates to 0) or ON (equates to !0). The ! preceding the value is triggered if the current value is NOT the one stated (so for example for a fader on, 0 = off, anything else = on).

See also RODE: for a simplified list of MIDI commands for the RØDECaster console.

Valid MessageType values are ControlChange, NoteOn and NoteOff. Any others will be passed through as raw data.

Notes equate from 0 (C0) to 127 (G10). VClock can match in either format and can use the usual flat notes (Ab9) or the sharp notes that Midi uses by default (G#9) in Salvos.

PDMLAMP:<lamp>,<state> is triggered when a Telos 25-Seven PDM unit's front panel lamps change state.

lamp = BUILD|EXIT|COUGH|DUMP
state = ON|OFF

PDMSTATUS:<function>,<state> is triggered when a Telos 25-Seven PDM unit's status changes. This offers similar functionality to PDMLAMP but has extra states that may be useful alongside the lamp tallys for the buttons.

function=BYPASS|BUILDING|DELAYSAFE|DELAYFULL|MUTED|EXITING|DELAYEMPTY
state = TRUE|FALSE

Must functions are self-explanatory. DELAYSAFE signals when > 4 seconds of delay is available. MUTED is triggered when you hold the COUGH button down until there is no delay left… the unit outputs silence whilst the button continues to be held.

RODE:<MessageType>,<Channel>,<Value> is a simplified list of MIDI commands for the RØDECaster console, and gets translated to the full MIDI command internally. See the RØDECaster section in Appendix D for more details.

SAS:<command> looks for an exact match for the incoming data from an SAS Systems console, etc. Some of these are simple strings such as M00050907 when a Module changes state for example. So:

SAS:M00050907

… would trigger for this event.

Other events have special control characters so we have created some triggers to simplify the process. See the next few triggers.

SASMODULE:<ConsoleID>,<Source>,<State> (State can be ON | OFF | CUEON | CUEOFF). For example:

SASMODULE:5,907,OFF (which would translate to M00050907 as in the raw example above).

SASOPTO:<OptoNumber>,<State> (State can be ON | OFF). For example:

SASOPTO:22,ON (which would translate to Z00:0022:1)

SASRELAY:<RelayNumber>,<State> (State can be ON | OFF). For example:

SASRELAY:22,ON (which would translate to Z01:0022:1)

SASXPOINT:<Dest>,<Source> will trigger when the destination is changed to the specified source. You can also precede the source with an ! to invert the match. For example:

SASXPOINT:1234,5678 (matches when the source is 5678)
SASXPOINT:1234,!5678 (matches when the source is anything BUT 5678)

\\SNMPv1\ SNMP is the odd one out in terms of trigger format. Rather than being something like SNMP:\\path\to\value, it is \\SNMPv1\path\to\value.

SNMP traps are received by the clock as follows:

```
** SNMPv1 TRAP from 10.92.186.250:1048
*** community public generic id: 6 specific id: 1
*** PDU count: 3
**** Vb oid: 1.3.6.1.4.1.2007.5.666.2.1.1.1.0 type: Integer32 value: 9
**** Vb oid: 1.3.6.1.4.1.2007.5.666.2.1.1.4.0 type: OctetString value: No input
level detected in 0 seconds
```

```
**** Vb oid: 1.3.6.1.4.1.2007.5.666.2.1.1.2.0 type: OctetString value: Wed Nov
28 14:01:01 2012
** End of SNMPv1 TRAP
```

(this is a ProntoNet codec telling us that silence is detected).

VClock translates the above into the following:

```
\\SNMPv1\10.92.186.250\public\6\1\3.
\\SNMPv1\10.92.186.250\public\6\1\3\1.3.6.1.4.1.2007.5.666.2.1.1.1.0\9.
\\SNMPv1\10.92.186.250\public\6\1\3\1.3.6.1.4.1.2007.5.666.2.1.1.4.0\No
input level detected in 0 seconds.
\\SNMPv1\10.92.186.250\public\6\1\3\1.3.6.1.4.1.2007.5.666.2.1.1.2.0\Wed
Nov 28 14:01:01 2012.
```

This string can then be entered as the Salvo in VClock in the "Serial & IP" column to be used as a trigger.

Each section of the string can be specified, or if you don't care about that section, it can be missed out. So, for example, taking the second line above as the trigger, both of the following strings in the Salvo would be recognised:

```
\\SNMPv1\10.92.186.250\public\6\1\3\1.3.6.1.4.1.2007.5.666.2.1.1.1.0\9.
\\SNMPv1\10.92.186.250\\\1\\\9.
```

(the values that tend to change are the PDU data (the last section), the specific ID (1 or 2 fail or ok on a ProntoNet), and the PDU count (success has an extra line of data, so the PDU count is 4 rather than 3 for a ProntoNet).

STREAMDECK:<buttonNo>,<state> will trigger when the specified button is pressed or released on the StreamDeck device. Different models of Streamdeck from 6 to 32 buttons are available. State is DOWN or UP.

STREAMDECK:INIT will trigger when a StreamDeck device is first connected (or reconnected). It can be used to set default text on the StreamDeck buttons, etc.

VMIX:<inputNumber>,<state> will trigger when Vmix Video Mixing software output states change, such as an input/camera being routed to the output/preview, or an overlay being enabled. Valid states are ON | OFF | PREVIEWON | PREVIEWOFF | OVERLAYxON | OVERLAYxOFF (where x can be 1-4). See the VMix section in Appendix D for more details.

WHEAT:<SLIO>,<state> will trigger when the specified SLIO number on a Wheatstone Blade changes to the specified state (High (H) or Low (L)). For example (with SLIO 12):

```
WHEAT:12,H
WHEAT:12,L
```

Clock Properties

Properties are divided into 7 sections. Caption Controls, Clock Controls, GPI Controls, Lamp Controls, Logging Settings, Miscellaneous Controls and WebBrowser Controls. Highlighting each setting in the Properties Panel gives a brief description of that property. Below are each in detail.

Calendar Controls (These are hidden for VClock Lite and Standard licenses)

CalendarFormatNow – Text that gets used when there is a current booking for the room/studio. **%FROM%** gets replaced with the time the booking was from, **%TO%** with the time the booking is to and **%TITLE%** with the title of the booking.

CalendarFormatNext – Text that gets used when there is a booking in the future for the room/studio. **%FROM%** gets replaced with the time the booking was from, **%TO%** with the time the booking is to and **%TITLE%** with the title of the booking.

CalendarFormatAllDay – Text that gets used when there is an all day booking for the room/studio. **%TITLE%** gets replaced with the title of the booking.

CalendarFormatSummaryAvailable – Summary Text that gets used when there is no current booking. **%NEXT%** gets replaced by the time of the next booking.

CalendarFormatSummaryBooked – Summary Text that gets used when there is a current booking. **%NEXT%** gets replaced by the time of the next availability.

CalendarFormatSummaryNoBookingsFound – Text that gets used when there are no bookings found. There are no relevant variables for this one.

CalendarIncludeAllDay – If disabled (false), All Day Bookings will be ignored in the calculation for the next availability, and will not be included in the CalendarFormats.

Caption Controls

TopCaption allows you to display messages across the top of the screen. It can contain several messages (separated by the " | " (pipe) symbol), and will rotate them every X seconds, where X is set using the **TopCaptionInterval** setting. **TopCaptionColour, TopCaptionFont** and **TopCaptionSizePercentage** can also be set. The caption strings can also contain several "VClock Variables". **TopCaptionSizePercentage** is a percentage of the automatically calculated size – so if you want it larger than "normal" set to greater than 100. Less than 100 makes it smaller. **TopCaptionTimeZone** sets which TimeZone to use when displaying a clock using "VClock Variables"

MiddleCaption is similar, but the position is approx two thirds of the way down the clock face. It is intended really when the clock face is disabled using ClockStyle=None.

BottomCaption is the same, and will display below the clock face. Defining this will DISABLE the LanguageClock.

LanguageStyle sets the style of the display across the bottom of the screen. It has several options, including a "Custom" one which allows complete control over the text displayed by editing the CUSTOMCLOCK.TXT file. See Appendix B for info. **LanguageClockColour**, **LanguageClockFont**, **LanguageClockSizePercentage** and **LanguageClockTimeZone** also affect this. This is also where you would select non-English languages. These actually work like the CUSTOMCLOCK.TXT file – a file called CUSTOMCLOCK_<Language>.txt is created, which allows you to tweak the configuration should you want to say "midnight" differently, etc. English also uses one of these files in case you want to say the time slightly differently to the default. See Appendix B.

Clock Controls

ClockBackgroundImage is an image (ideally square) centered on the clock face. It can be used to add a bespoke background to the clock face and could include tick marks, numbers and so on. **ClockBackgroundImageSizePercentage** zooms in/out this image.

ClockPosition is an advanced setting, usually set to Fill. Once the Lamps have been assigned their locations, the clock auto-fits into the remaining space. This setting can offset the clock to one direction (up/down/left/right), to make extra space for any other app you may want to display on top of the VClock application.

ClockPositionMode chooses whether the clock is automatically positioned (based on various factors such as number of lamps per row/column, number of rows/columns and positioning of the rows/columns), or whether you want to manually specify the coordinate of the center of the clock face. Default is Automatic.

ClockPositionCenter is automatically updated if the ClockPositionMode is set to Automatic. When changed to Manual the ClockPositionCenter specifies the coordinates of the center of the clock face. This is in the format "LeftPosition TopPosition" and each value is a percentage of the overall display area (so 50 50 would make the clock in the center of the application).

ClockRadiusSizePercentage sets the percentage to adjust the automatically calculated size of the radius of the clock face. So 200 would make it twice the size, 50 would make it half. This only has an effect when ClockPositionMode is set to Manual.

ClockCaption is the caption displayed on the clock face. "\n" means new line. If the clock is unlicensed then this will not be displayed. Also, if a logo is assigned (using the ClockLogo setting) then the logo takes precedence. **ClockCaptionColour**, **ClockCaptionFont** and **ClockCaptionSizePercentage** are also used to control this caption.

ClockHandColour sets the colour of the hour and minute hands on the analogue clock face (if enabled using the ClockStyle setting). **ClockSecondHandColour** sets the second hand colour. **ClockSecondHandStyle** adjusts the length / thickness / shape of the second hands (or disables them). **ClockHandStyle** adjusts the hour and minute hands style, and can adjust the length and thickness of the hands or change the style to a pointed hand.

ClockRTCTraceBrightness and **ClockRTCTraceMaxSeconds** control a trace that surrounds the standard second hand, when there is a non-zero PDMDelay set. The trace shows the difference between the offset time and the real time clock. Brightness is a value between 0 (transparent) and 255 (solid). MaxSeconds is between 0 (disabled) and 60. The trace is only shown if the offset is less than the MaxSeconds defined (and not zero). The "Adjust Time by PDM Delay" option under settings/miscellaneous tab decides whether overall time (and backtime counters etc) are adjusted by the PDM Delay. If it is selected, the trail will be behind the second hand (and the second hand will show the adjusted time). If not, the second hand will show real time, with the trail following the second hand. (PDMDelay requires a PLUS license).

ClockInnerColour and **ClockOuterColour** set the center and outer colours of the clock face. The clock will create a gradient of colour between these two points. It is perfectly acceptable to set them both to be the same colour for a flatter clock-face. These colours are also not in use if the **EnableClockBackground** setting is set to false as the clock face in effect becomes transparent.

ClockLogo can override the ClockCaption with a logo of your choice. Simply point it to a standard JPG/BMP/GIF/PNG file. This will not be displayed if the clock is unlicensed. You can then tweak the size of your logo with **ClockLogoSizePercentage**. The larger the number, the larger the logo.

ClockNumbersColour sets the colour of the numbers on the clock face (if displayed). **ClockNumbersFont** sets the font and **ClockNumbersSizePercentage** sets the font size (relative to the auto-calculated size). **ClockNumbersStyle** selects what numbers are displayed (all 12, only 4 quadrics, seconds (rather than minutes), or none). **ClockTickColour** sets the colour of the marks ("ticks") denoting the minutes around the edge of the clock face. **ClockTickStyle** sets how many ticks/dots are displayed, and whether they are lines, dots or disabled entirely. They can also be set to change colour based on the second or minute of the current hour.

AnalogueClockTimeZone sets which TimeZone to use for the analogue clock.

BacktimeValues. By default the clock backtimes to the top of the hour. This setting is a comma-seperated list of different points within the hour to backtime to, in the format MM:SS (for example 00:00,15:00,30:00,45:00 to backtime to each quarter hour). If the default of 00:00 is not used, the clock will display the time that it is backtiming to in square brackets. You can also specify specific times of day in the same way (in the format hh:mm:ss). The overall display can be disabled by using the **EnableBacktimeCountdown** setting, though the value can still be used with the %BACKTIME and %BACKTIMETARGET VClock Variables.

DateStyle selects whether a date or day and data is shown in the number 3 position, or whether this feature is disabled. If ClockStyle is set to Digital this setting has no effect.

ClockStyle selects between an Analogue clock face (with optional Digital clock within it), a Digital clock face (with logo), a larger digital clock face (seconds on the 2nd line, no logo). no clock face (but still captions and logos and so on), or a "Wall of Lamps" (which disables every part of the clock and only shows lamps in a fixed layout).

DigitalClockColour, **DigitalClockFont**, **DigitalClockSizePercentage** and **DigitalClockTimeZone** set the colour, font / font size and timezone of the Digital clock displayed in the lower half of the analogue clock face, provided

EnableDigitalClock is enabled. These settings are also used for the main digital clock when ClockStyle is set to Digital or LargeDigital (assuming that EnableStopwatch is not enabled, which replaces the Digital Clock!).

DigitalClockFormat allows you to specify a time format different to your machine's regional settings, using standard DateTime formatting variables. For example "DigitalClockFormat=hh:mm:ss" will show a 12 hour clock, without am/pm. HH:mm:ss would be 24 hour clock.

DotStyle sets the style of 60 dots around the circumference of the clock. They can represent seconds or minutes, or can represent the Stopwatch value rather than real time (in minutes). Or they can be disabled. They can also be solid colours, or a gradient to add a more 3D effect.

DotColour1, **DotColour2**, **DotColour3** and **DotColour4** set what colour the dots are. The first is used for 1-10 seconds, the 2nd for 11-20, the 3rd for 21-35 and the 4th for 36-60.

DotZeroAlwaysOn sets the top dot on the clock face to always be on (true) or off (false). Traditionally VClock lights the dots starting at 1 and counting to 59, then all off for 0. Other hardware clocks have the top lamp on at all times, so all dots are lit at 60. The default is false.

EnableStopwatch overrides the Digital Clock, replacing it with a stopwatch style counter. **StopwatchColour, StopwatchFont** and **StopwatchSizePercentage** control the look of the stopwatch. **StopwatchMode** can be Up or Down (the direction in which it counts!).

DMX Controls (These are hidden for VClock Lite and Standard licenses)

DMXIdleState allows you to set commands that can be called by the command DMX=Idle. This is useful if you want, for example, buttons to select the normal colour of a studio, but have Salvos that turn the room red when the mics are live. The buttons can set the DMXIdleState, and when the mic goes off it can revert to the current Idle State, rather than a predefined colour.

The command takes several DMX command separated by +. For example:

```
DMX=1:Colour=Red/500+11:255,0,0
```

GPI Controls

The settings under this section are about the Analogue (voltage) inputs available on some Advantech GPI cards. These are up to 8 (depending on how many the card supports) voltages that are read as a value between 0 and 65535.

AnalogueInputScalingMaths allows you to scale this 0-65535 value to something more usable (maybe a temperature between 0 and 100). The default is "X/655.35". X is the original value.

AnalogueInputDecimalPlaces specifies how many decimal places to include in the "scaled" value.

AnalogueInputLogEventValueChange allows you to specify a string to log each time the *scaled* value changes (the original 0-65535 value will change constantly by a small amount, the scaled value should be more static). The string can contain "VClock Variables". The most useful one is probably the analogue value (%ANALOGUEINx and %ANALOGUEINxSCALED where x is the input number).

Lamp Controls (These are all hidden for VClock Lite)

LampBacktimeValues is similar to BacktimeValues in the Clock Settings above, but is an independent backtimer for each lamp. VClock Variables **%LAMPxBACKTIME** and **%LAMPxBACKTIMETARGET** can be used on any caption for these.

LampBorderSizePercentage creates a border around the lamp, following the shape of the lamp. Setting to 0 disables the border. The colour is set with **LampBorderColour**. **LampBorderFollowsLampBrightness** decides whether the border is dimmed when lamps are dimmed, etc.

LampColour sets the overall colour of the lamp.

LampCaption sets the text on the lamp. "\n" means new line. **LampCaptionColour** and **LampCaptionFont** set the colour and font of this. **LampCaptionSizePercentage** sets the size of the font, relative to the auto-calculated size (so a value greater than 100 makes it larger than "normal", a value less than 100 makes it smaller). If a **LampLogo** is set, this overrides the caption, and **LampLogoSize** makes the logo larger/smaller. **LampLogoFollowsLampStatus**, if true, turns the logo off when the lamp is off (or flash on/off as the lamp flashes, etc).

LampShape sets the shape of the lamp (Rectangular, Circular, Square).

LampCurvePercentage is used when the shape is not circular, to round the corners of the square/rectangle. 0 is the default and leaves sharp corners. 100 would actually draw a circle if the shape were a square. 20% to 40% are good values to use.

LampState sets the state of the lamp (On, Off, Solid, SolidDim, OnOff, OnOffSolid, OnOffFast, OnOffFastSolid, OnDisabled, OnDisabledSolid, Phone, PhoneSolid, Flash). If set to Disabled, this hides the lamp completely. If set to Off or SolidDim, **LampOffStateBrightness** sets how dim or bright the lamp looks.

LampTimeout sets how long the lamp stays in the current state when the state changes. After the timeout (in seconds), a Salvo of "LampXTimeout" (X being the lamp number) gets sent to the clock, which can be used in the Salvo list to undo the state change (so for example turn it off 10 seconds after it is turned on by a pulse from a door bell). Default is 0 (disabled).

LampErrorText is a text string that can be displayed on the Top/Middle/Bottom Captions and will form part of the RSS feed from the clock. It can be several strings separated by the | symbol as with other parts of the software.

LampGPO is an Advantech or XKeys GPO number linked to the lamp, which follows the lamp status (so if the lamp is on the GPO is on, if the lamp is off the GPO is off and if the lamp is flashing then so is the GPO). Only available in VClock Plus.

LampMultiStateValue is a value that can be incremented or decremented with a Salvo command (**LampMultiStateValue=**INC or **LampMultiStateValue=DEC**), or set to a specific value. Most useful when incremented when a lamp is clicked on with the LampMouseDown event. Each time the lamp is clicked, the MultiStateValue increments, which can trigger a different Salvo, setting the display of the lamp (for example toggling between date, time, day, time in different timezone, etc). When **LampMultiStateMax** is reached, LampMultiStateValue returns to 1 (and vice-versa if decremented).

LampOnCommand and **LampOffCommands** are commands that can be acted upon by VClock when the lamp changes state to an "on" state (or any of the flashing states) or an "off" state. For example Lamp1OnCommand could be set to Lamp2State=Off and Lamp1OffCommand could be set to Lamp2State=On. Thereby creating an opposite lamp.

LampLogEventOn and LampLogEventOff allow you to specify a string to log each time the lamp is turned on (or off). The string can contain "VClock Variables".

LampPositionMode chooses whether the clock automatically positions the lamps (based on various factors such as number of lamps per row/column, number of rows/columns and positioning of the rows/columns), or whether you want to manually specify the coordinate of the individual lamps. Default is Automatic.

LampPosition is automatically updated if the LampPositionMode is set to Automatic. When changed to Manual the LampPosition specifies the coordinates of the lamp. This is in the format "LeftPosition TopPosition Width Height" and each value is a percentage of the overall display area (so 0 0 20 20 would make a lamp in the top left corner, 20% of the width of the screen and 20% of the height). You can also use the letter H in the first/third parameter and W in the second/fourth, as a calculation based on the height/width of the lamp. So for example 0 0 H*5/4 20 would make a lamp in the top left corner, 20% of the height of the screen, with a ratio of 5x4 (so a rectangle).

LampStreamDeckKey sets which StreamDeck key to mirror with the on-screen Lamp. 0 (default) disables the feature.

LampStreamDeckMode is used to determine whether the on-screen lamp is displayed when LampStreamDeckKey is set. By default, both lamps are shown (MirrorLamp). Optionally the on-screen lamp can be disabled by setting to StreamDeckOnly.

LampTimeZone sets which TimeZone to use when displaying a clock using the "VClock Variables".

Carrying out commands on a range of Lamps

It is possible to carry out all of the above commands on multiple lamps. The most common use case would be to interlock some lamps for a transmission switcher, source selector, etc.

    Lamps[a,c,e-g,i-k,l]Function=Value

For example for the lamp status, the following formats are all valid:

```
Lamps[1-3]Status=On
Lamps[5,7]Status=On
Lamps[1-3,5,7,9-10]Status=On
```

Logging Settings

The settings under this section specify what the Logfile will be called, and what format it will be written in, when using functions such as LampXLogEventOn, LampXLogEventOff, AnalogueInputXLogEventValueChange or send a command of LogToFile=<string to write>.

LogFile is the full path to the file to be written, including the filename (which can contain "VClock variables" to specify the month, day, year, etc). For example C:\Logs\%M-%Y.LOG

LogEntryFormat is the format that the entry will be written in. This string must contain %ENTRY which is the text passed to the Log function. This overall setting LogEntryFormat allows you to prepend all entries with a standardly-formatted date/time, etc again using "VClock Variables".

Miscellaneous Controls

BackgroundColour sets the overall background of the application (ie anywhere that is not displaying a clock, lamp or text, and if the EnableClockBackground setting is disabled then the background of the clock face as well.

BlankScreen draws the background colour but nothing else (ie no clock, no lamps). It is intended to be used as a screensaver. It could be triggered by a "Time of Day" event, or even by a closure driven by a detector in the studio that tells the clock when there is no-one in the room(!). Default is false.

GlobalTimeOffset applies an offset to the time globally across the application. As a command you can also use GlobalTimeOffsetInSeconds= to set the offset (in the format 12.345 – ss.ttt). This can be used to adjust if an NTP server is giving out an offset time, if you wanted to run the time at a facility a few seconds ahead for some reason, etc. This setting would rarely be used.

InstantAlarmFile is the audio file which is played when the instant alarm feature is used (by right-clicking the clock face).

LampsPosition sets the location of the lamps. The default setting is Fill, which will position the lamps to each side of, or above/below the clock, depending on the orientation of the application (if it is wider than it is tall the lamps go to the side, if taller then they go above/below). Setting to None disables the lamps entirely.

NoOfLampsPerSide sets the number of lamps in the first column (or row, depending on the lamp positions). A setting of 4 to 8 is valid.

NoOfRowsOfLamps sets how many rows of lamps there are (1 or 2, default 2). If set to 1, the clock face acquires the extra space that the 2nd row (or column!) of lamps would have taken.

Opacity sets how transparent the clock is on-screen. Values are from 0 (the clock totally disappears) to 1 (the clock is totally visible). A value of say 0.5 would allow you to see other applications or the desktop that are behind the clock. Could be useful as an alternative screensaver to BlankScreen above.

OverallBackgroundImage prints a background across the entire canvas of VClock. **OverallBackgroundImageSizePercentage** allows you to zoom in/out of the image.

VLC Controls (these are hidden for Lite and Standard versions)

Note that you must have VLC installed on the VClock machine in order to use the embedded VLC Players. You must also have the path to VLC set in the Miscellaneous tab of VClock's settings pages.

VLCxAutoStart decides if the video player will start to play automatically when a file or stream is assigned to it, or at startup of VClock.

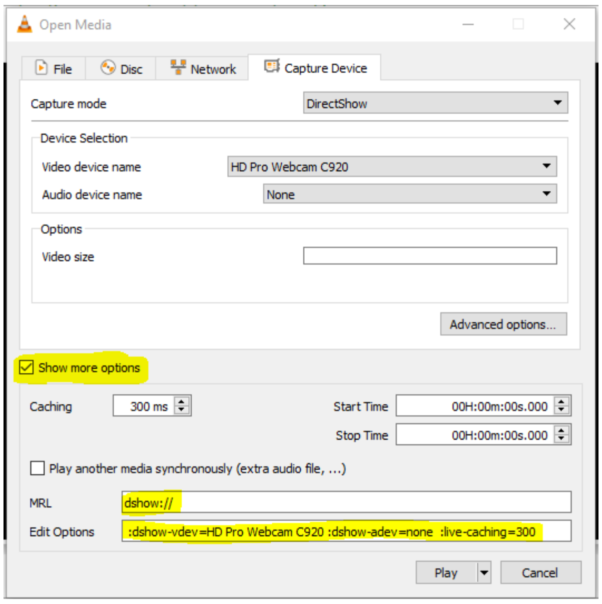
VLCxEnabled enables/disables the player. When disabled it is removed from the screen.

VLCxLoop decides if the video will start again when it reaches the end.

VLCxMRL sets the path of the video file or stream (the Media Resource Locator). This can be in formats such as c:\test.mp4, http://127.0.0.1/stream, rtsp://127.0.0.1/stream etc.

VLCxMRLOptions sets any options required for the playback. For example, if the MRL is "dshow://", the Options may be the name of a webcam (":dshow-vdev=NewTek NDI Video").

The easiest way to find the MRL and MRLOptions is to open VLC itself. Open a capture device or network screen but before playing, click on the "Show More Options" checkbox, then copy the "MRL" and "Edit Options" from there:



Capture mode: DirectShow
Video device name: HD Pro Webcam C920
Audio device name: None
Video size:
☑ Show more options
Caching: 300 ms
Start Time: 00H:00m:00s.000
Stop Time: 00H:00m:00s.000
☐ Play another media synchronously (extra audio file, ...)
MRL: dshow://
Edit Options: :dshow-vdev=HD Pro Webcam C920 :dshow-adev=none :live-caching=300

VLCxMute mutes the incoming audio of any stream/video.

VLCxTimeout sets the number of seconds a video can not progress its position, before we reconnect. A value of -1 disables this feature. HTTP Streams in particular are not detected by VLC when they fail and then re-appear.

VLCxPosition sets the position of the VLC Player on-screen. As usual this is as a percentage of the overall VClock size, in the format "LeftPosition TopPosition Width

Height" and each value is a percentage of the overall display area (so 0 0 20 20 would make a Player in the top left corner, 20% of the width of the screen and 20% of the height).

Voice Controls (these are hidden for Lite and Standard versions)

Note that you can select the Voice to use in the Miscellaneous tab of VClock's settings pages.

VoiceSpeed sets the speed of the voice. Valid settings are -10 to 10.

VoiceVolume sets the volume of the voice. Valid settings are 0 to 100.

WebBrowser Controls

WebBrowserURL sets the webpage to be displayed. If blank, this disables the WebBrowser.

WebBrowserPosition sets the location of the browser, relative to the clock face, using Top|Bottom|Left|Right|Fill|None (or T|B|L|R|F|N). Alternatively, a co-ordinate can be used in the format "LEFT TOP WIDTH HEIGHT". For example

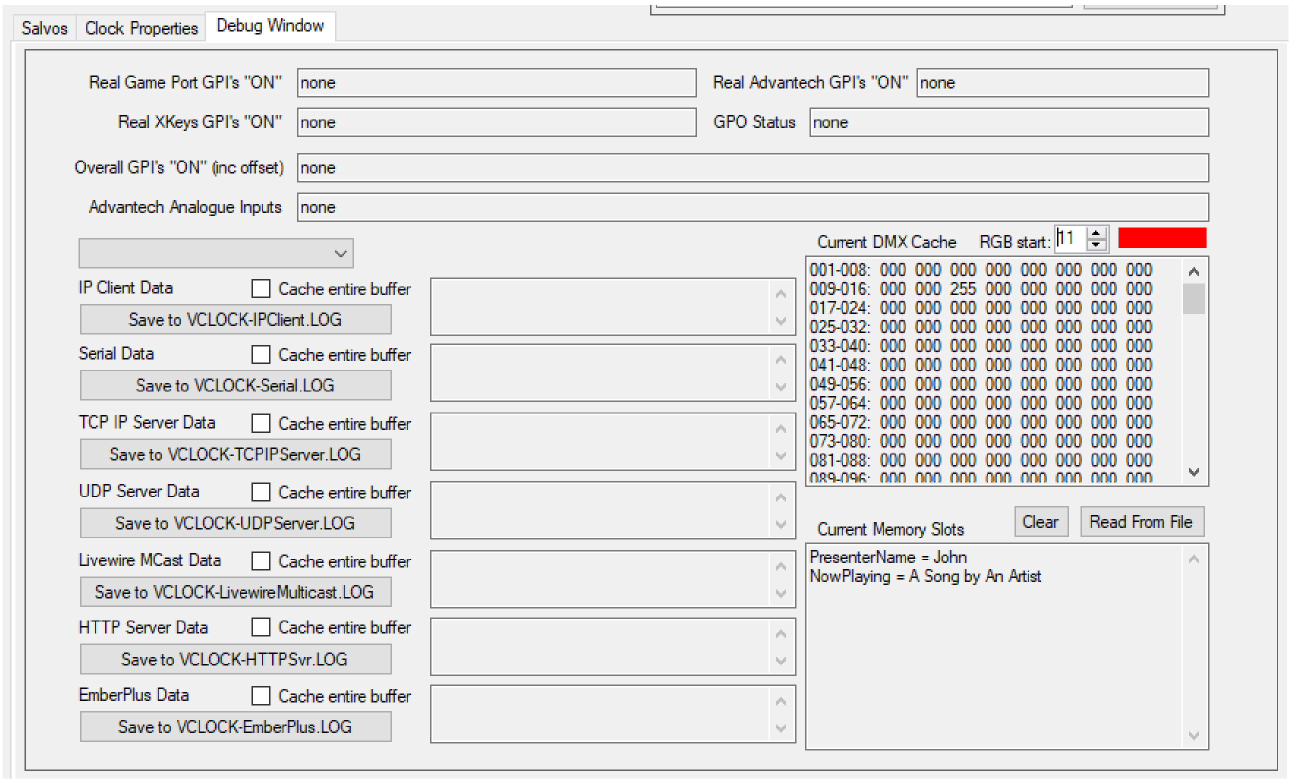
```
        WebBrowserPosition=10 10 80 80
```

WebBrowserFullScreen=True|False overrides the WebBrowserPosition and if enabled replaces the entire VClock screen area with the WebBrowser. Useful for Branding screens – Vclock can load different graphics into the WebBrowser via triggers from buttons on the console, etc.

The webpage should automatically zoom as the clock is resized, but to allow tweaking of this the **WebBrowserZoomPercentage** can be used.

Debug Window

The Debug Window tab gives real-time info about GPIs and Serial / IP / HTTP data received, the current state of the DMX channels and any memory values in use:


Salvos
Clock Properties
Debug Window
Clear
Read From File
Save to VCLOCK-IPClient.LOG
Save to VCLOCK-Serial.LOG
Save to VCLOCK-TCPIPServer.LOG
Save to VCLOCK-UDPServer.LOG
Save to VCLOCK-LivewireMulticast.LOG
Save to VCLOCK-HTTPSvr.LOG
Save to VCLOCK-EmberPlus.LOG

Real Game Port GPI's "ON" none
Real XKeys GPI's "ON" none
Real Advantech GPI's "ON" none
GPO Status none
Overall GPI's "ON" (inc offset) none
Advantech Analogue Inputs none

IP Client Data ☐ Cache entire buffer
Serial Data ☐ Cache entire buffer
TCP IP Server Data ☐ Cache entire buffer
UDP Server Data ☐ Cache entire buffer
Livewire MCast Data ☐ Cache entire buffer
HTTP Server Data ☐ Cache entire buffer
EmberPlus Data ☐ Cache entire buffer

Current DMX Cache RGB start: 11 🟥

```
001-008:  000  000  000  000  000  000  000  000
009-016:  000  000  255  000  000  000  000  000
017-024:  000  000  000  000  000  000  000  000
025-032:  000  000  000  000  000  000  000  000
033-040:  000  000  000  000  000  000  000  000
041-048:  000  000  000  000  000  000  000  000
049-056:  000  000  000  000  000  000  000  000
057-064:  000  000  000  000  000  000  000  000
065-072:  000  000  000  000  000  000  000  000
073-080:  000  000  000  000  000  000  000  000
081-088:  000  000  000  000  000  000  000  000
089-096:  000  000  000  000  000  000  000  000
```

Current Memory Slots

```
PresenterName = John
NowPlaying = A Song by An Artist
```

This is quite a busy screen, and on lower resolutions you may get a scroll bar to the right to see some of the lower options.

The **Game Port, XKeys and Advantech GPI's boxes** will show the number of all "active" GPIs (so GPIs that are "high", or that are "low" if the "Invert Signals" option is used). This will show the "real" GPI number. The "Overall GPIs" box will show which GPI numbers are currently active, including the "offset" setting (see settings section below). GPO Status shows any active GPO OUTPUTS from VClock.

Advantech Analogue Inputs will show the analogue values read from any Advantech card that supports it. It will be a value from 0-65535 for each input (up to 8), and in brackets will show the adjusted ("scaled") value using settings that you provide (see later).

The **TCP IP, UDP, Serial, Livewire Multicast GPIO, HTTP and EmberPlus Data Buffers** show the data as it comes in to the ports (for IP Client, you can select which IP Client you want to watch). This is useful if you are looking for the string that is sent, so that you can copy/paste it to a Salvo. Pressing the corresponding "Save" button writes the current contents of the box to a LOG file (overwriting each time), and opens the file in Notepad.

Data coming in is added to the end of the existing data buffer each time, then written to the debug screen, then parsed for any matches in the Salvos. When a match is found, or if the buffer is longer than the longest salvo, all of the buffer up to that point is deleted, then the parsing continues. The screen however does not refresh until the next lump of data is received, meaning that the previous "match" remains on-screen for a time. If you check the "Cache Entire Buffer" box, the textbox will always append the new data and never delete anything – so useful for debugging (it will let you save the whole session to a text file), but do not leave it in this state long-term as it will eventually turn into a massive memory leak for the application.

The top grid ("**Current DMX Cache**") on the right shows the current DMX values for each of the 512 DMX channels. DMX is only available with a "plus" licence.

Above this is an RGB Start value with a coloured box to the right of it. This will show the RGB colour set by the start position of the DMX (and the following 2 values).

The lower window on the right shows any **Current Memory Slots** set, along with their values. There are buttons to clear the values, and to re-read them in from disk (stored in a text file called MemorySlots.txt in the config folder, if you needed to edit them externally).

The **Read from Disk** option is only available if the "Remember Memory Values when Restarted" option (under Miscellaneous settings) is selected.

VClock Variables

Several settings within VClock allow you to specify Variables as part of the string. Specifically LampXCaption, TopCaption, MiddleCaption, BottomCaption, WriteToFile, LogFile, LogEntryFormat, LampXLogEventOn, LampXLogEventOff, AnalogueInputXLogEventValueChange and LogToFile.

For any variables referencing a date or time, the relevant TimeZone setting is used to adjust the time (**LampXTimeZone**, **TopCaptionTimeZone** and so on).

%hour %12hour %min %sec %ampm %dow %day %month %year allow you to format your own date/time (%dow is returned in the OS language, if you want to specify your own, use the CustomClock.txt file and use **%customdow**).

%ordinalday returns the day of the month in ordinal format (1st, 2nd, etc). Note this only works in English.

%moy returns the month of the year (in words). This will be returned in the OS language.

%time %date get replaced by pre-formatted time and date strings.

%h %m %s %t %D %M %Y are another way to get the time/date, and **%NOW** is the same as **%time**.
There are also "rtc" versions of these which ignore the **PDMDelay** if set, and always tell the actual real-time clock (taking the **GlobalTimeOffset** into account). Prepend any of the above time variables with rtc **(%rtchour, %rtcNOW etc**).

%TIMEOFFSET will display the current GlobalTimeOffset applied to VClock.

%english %englishnumbers get replaced by the internal formatted string versions of the time (the same format as the "Language Clock", which this method supersedes), and **%custom** enables the CustomClock format (see Appendix B for details of the file format). If **LanguageStyle** is set to a language (English, French, German, etc) then **%custom** will display the time in this language.

%COUNTER will add a stopwatch style counter to the Top/Middle/Bottom and Clock Captions. This is reset when the caption is changed, or with the commands **TopCounterReset=**, **MiddleCounterReset= BottomCounterReset=** or **ClockCounterReset=** (the = is needed in the command… anything can follow it, or indeed nothing at all!). This can also be used on any Lamp caption (**LampXCounterReset=** will reset these).
%STOPWATCH will add the stopwatch counter to any caption.
%STOPWATCHMINIMUM will add the minimum detail required (so only seconds if less than 60 seconds, then minutes and seconds, then hours and minutes and seconds and so on).

%UPCOUNTER and **%UPCOUNTERMINIMUM** will display the "Up Counter" on any caption, and similarly **%DOWNCOUNTER** and **%DOWNCOUNTERMINIMUM** will display the "Down Counter". These can be used in a similar way to the Stopwatch, but are designed more specifically for integration with Axia (see Appendix D). They can be controlled via Salvos in the same way as Stopwatch though, so can be used as extra counters.

%BACKTIME will show the current backtime counter of the main backtime display, if you want to show it on a lamp or caption. **%BACKTIMETARGET** will tell you the target we are currently trying to hit. Similarly **%LAMPXBACKTIME** and **%LAMPXBACKTIMETARGET** will do the same for each of the 32 backtime counters associated with the lamps.

%ANALOGUEINx (x = 1 to 8) displays the relevant analogue input value read from an Advantech card. It will display "Invalid" if the Advantech card does not support it / is not installed. **%ANALOGUEINxSCALED** displays the scaled value, using the AnalogueInputXScalingMaths and AnalogueInputXDecimalPlaces settings.

Finally, lamp statuses can be included (useful if a lamp is turned on as an alarm and you wish to display the Lamps ErrorText string for useful contact details, etc).

%LAMPXSTART is the time the lamp turned on, **%LAMPXDUR** is how long it has been on for and **%LAMPXERRORTEXT** is the text saved under the Lamp's ErrorText property.

%LAMPXMULTISTATEVALUE shows the current value of the MultiState. **%LAMPXMULTISTATEMAX** shows the maximum value set (before it reverts to 1).

%LAMPXTIMEZONE, %TOPCAPTIONTIMEZONE, **%MIDDLECAPTIONTIMEZONE**, **%BOTTOMCAPTIONTIMEZONE**, **%ANALOGUECLOCKTIMEZONE**, **%DIGITALCLOCKTIMEZONE** and **%LANGUAGECLOCKTIMEZONE** variables display the standard Windows TimeZone ID of the repective timezone setting.

%GETMEM(NAME) will be replaced with the current value of the specified memory slot.

%NTPDIFFERENCE shows the difference between the local PC's time and the NTP Server's time, when the NTP function is set to check but not set the PC clock.

%PDMDELAYn can be used in a caption to show the current delay of a Telos PDM or Cloudcast BDS (n = 0 – 3 and denotes the decimal places to round to, for example %PDMDELAY0 could display "3" whereas %PDMDELAY1 could be 3.4 and %PDMDELAY3 could be 3.396").

Memory Slots

VClock has user definable memory slots, which can be used to store strings, store numbers, do basic maths and trigger salvos when they change.

There are several commands:

SETMEM=NAME,VALUE will set (or update) a memory slot. The name defines which slot, the value can be any string or number.

ADDMEM(NAME,VALUE) will mathematically add (or subtract if negative) the specified value to the specified memory slot, provided the memory slot contains a number.

INCMEM(NAME) and **DECMEM(NAME)** will add or subtract 1 to/from the specified memory slot, so long as it is numeric.

%GETMEM(NAME) will be replaced with the current value of the specified memory slot, on LampCaptions and so on.

When memory slots are set or are changed, they cause internal triggers:

SETMEM:NAME will be triggered whenever the specified memory slot is created or updated.

SETMEM:NAME,VALUE will be triggered whenever the specified memory slot is created or updated to this specific value.

SETMEM:NAME,%VAR% will be triggered whenever the specified memory slot is created or updated. The %VAR% variable can be used in the command that it triggers.

SETMEM>:NAME,VALUE will be triggered whenever the specified memory slot is created or updated to a numeric value greater than the one specified.

SETMEM<:NAME,VALUE will be triggered whenever the specified memory slot is created or updated to a numeric value less than the one specified.

SETMEM><:NAME,VALUE will be triggered whenever the specified memory slot is created or updated to a numeric value between the ones specified.

The current memory values can be seen on the debug page. There are also buttons to clear them or to re-load them from disk.

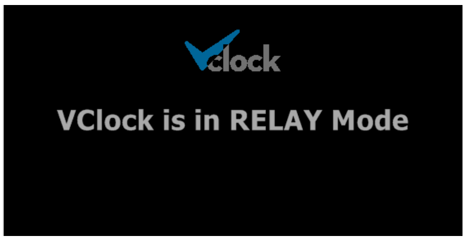
There is a setting in the VClock settings pages / Miscellaneous tab, to set whether memory values are remembered after a restart, or reset.

Relay Mode

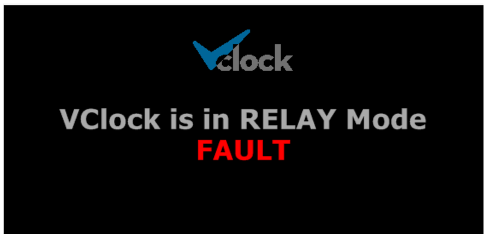
This is a special mode which actually disables the clock and lamps! It is designed for stations that need a central application to take in GPIs, Serial Commands, IP Commands, etc, and "relay" them out to other VClocks – perhaps at an OB across a Serial or IP link, which wants to know the status of GPIs back at the station.

Disabling the clock should make the application use less system resources, and VClock does not require a license to work in this mode. However VClock will relay the commands in standard mode as well, and will not stop relaying the commands even when the 10 "grace" events have ran out and stopped the local clock from acting on the commands.

When in Relay Mode, the clock face is replaced by a simple message:


VClock is in RELAY Mode

or if there is a GPI error fault will flash:


VClock is in RELAY Mode
FAULT

Text To Speech ("Speaking Clock") in detail

VClock can convert text to speech using the standard C# Speech.Synthesis libraries.

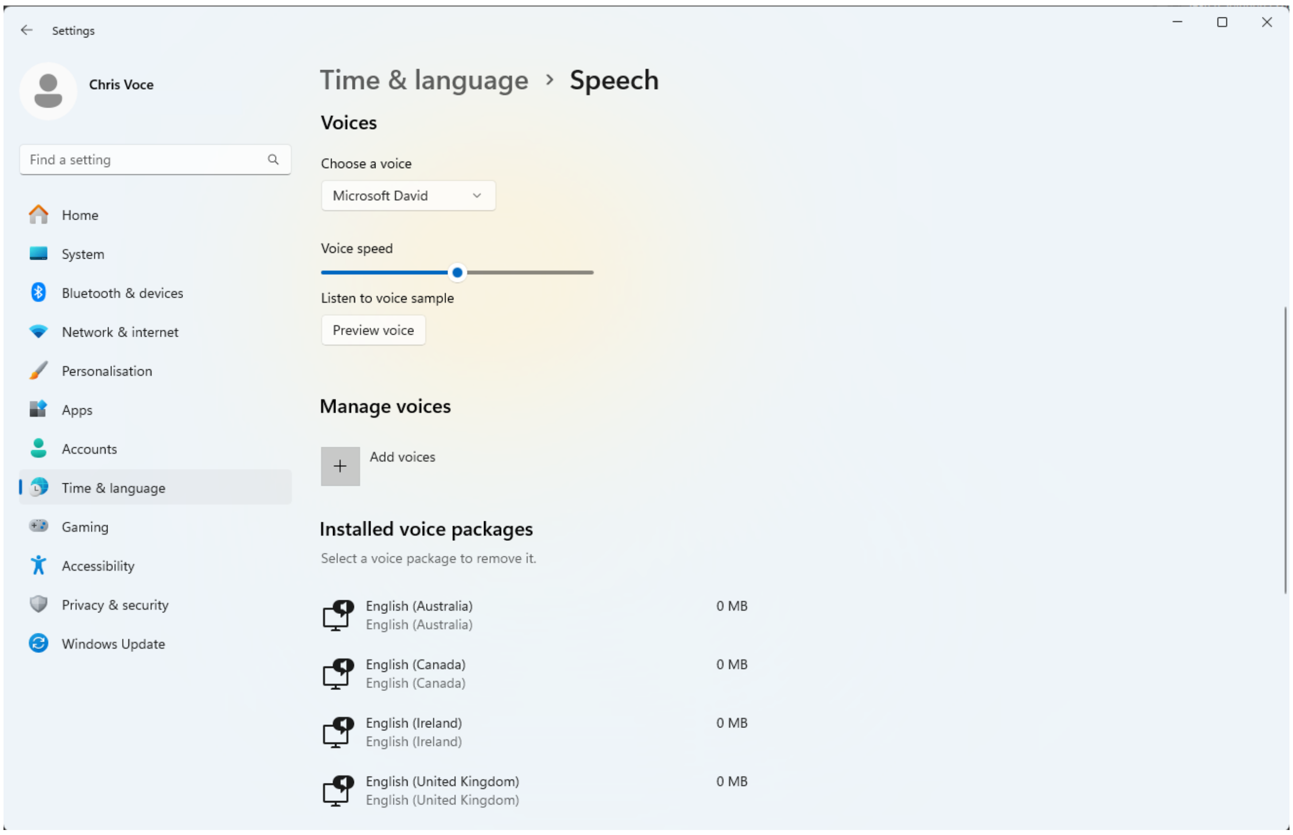
There are a couple of VClock properties to control the speed and volume of the speech, and a dropdown in Settings (Miscellaneous tab) to select the Voice to use.

You can make VClock speak with the Command SPEAK=

SPEAK can also use variables, and memory slots. So for example

```
SPEAK=Hello, world
SPEAK=The time is %english
SPEAK=%GETMEM(Message)
```

You can manage the voices that Windows has (in Windows 11), in Settings / Time & Language / Speech:


Settings

Chris Voce

Find a setting

Home
System
Bluetooth & devices
Network & internet
Personalisation
Apps
Accounts
Time & language
Gaming
Accessibility
Privacy & security
Windows Update

Time & language › Speech

Voices

Choose a voice: Microsoft David

Voice speed

Listen to voice sample: Preview voice

Manage voices

Add voices

Installed voice packages

Select a voice package to remove it.

English (Australia) English (Australia)	0 MB
English (Canada) English (Canada)	0 MB
English (Ireland) English (Ireland)	0 MB
English (United Kingdom) English (United Kingdom)	0 MB

From Windows 10 I believe – certainly in Windows 11, there is a disconnect between the voices you download in Windows settings, and the ones available to 3rd party applications. However, Googling "speech vs speech_onecore registry", there is a relative simple workaround to make these voices available.

The issue appears to be that there are now "Desktop" and "Mobile" TTS voices. The "Mobile" ones can only be used by the system itself, not 3rd party apps.

There are 2 registry locations for the voices. Desktop:

Computer\HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\Microsoft\SPEECH\Voices\Tokens

And Mobile:

Computer\HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\Microsoft\Speech_OneCore\Voices\Tokens

Under Tokens in each case is a key (and subkeys) for each Voice.

Making mistakes in the Registry can cause big problems, so be careful, back it up first, and if you don't feel confident then find someone who does!

That said, this is what I did (in Windows 11):

- Run Regedit (as Administrator) and browse to the Mobile tokens path above
- Right click the voice key that you want to use and choose Export
- Save it to a file
- Edit the file in Notepad to remove "_OneCore" (in 2 places in the file) – thereby setting it to the "Desktop" tokens path above
- Double click the file to re-import it to the new location.

If you restart VClock, that new voice should now be available.

Do this at your own risk! – though it worked fine for me, and Google suggests it is a widely used workaround.

TimeZones

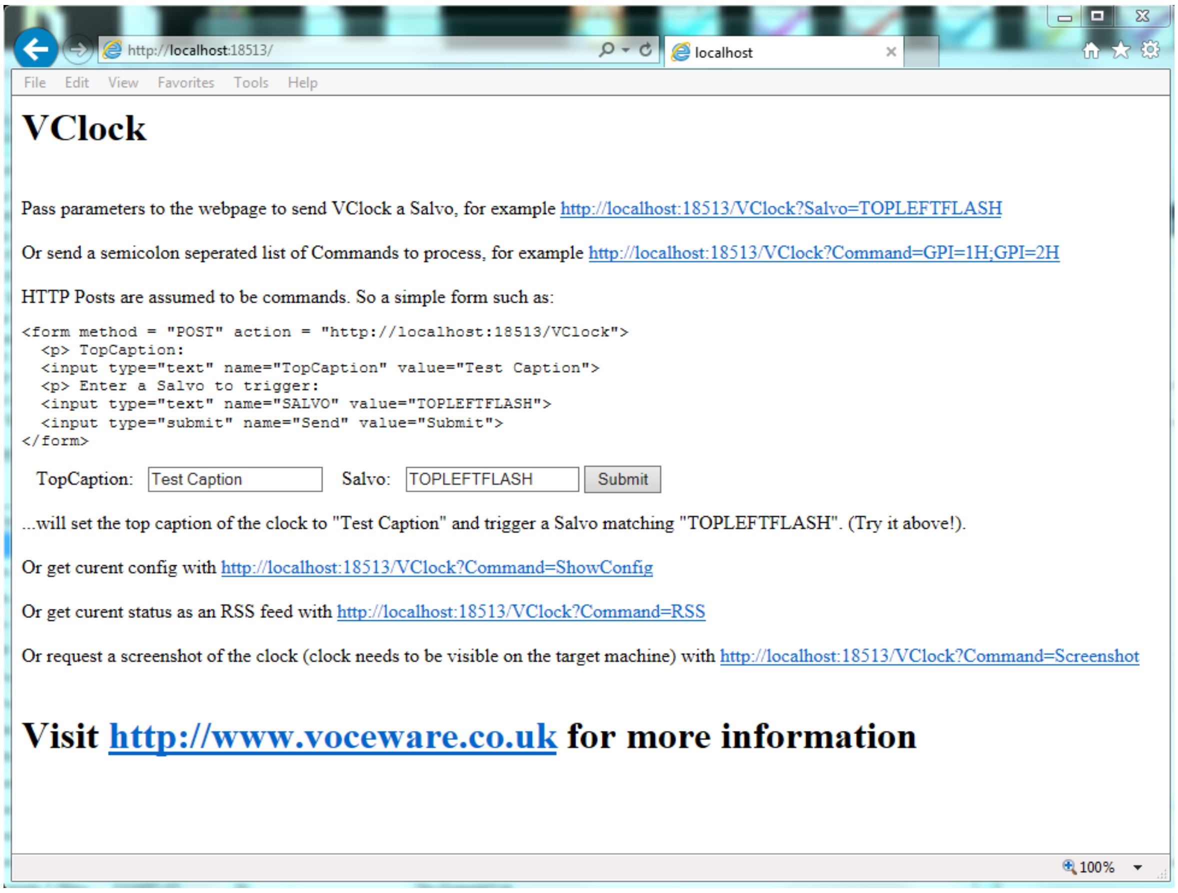
TimeZones can be independently set for each lamp, each caption, the analogue, digital and language clocks. These Timezone settings are used whenever a clock or a VClock Variable referring to a time are displayed.

The TimeZone setting is looked up in the standard timezones that the OS knows about. You can also set the TimeZone to “LOCAL” (local time), “GMT” or “UTC”. Additionally you can apply an offset to these in the form “LOCAL+01:30:00” or “GMT-01:00:00”. Finally, for all except the AnalogueClockTimeZone you can set them to “FOLLOW ANALOGUE CLOCK” (the default).

When a TimeZone is looked up, a partial match will suffice. So **AnalogueClockTimeZone=**Paris will set the TimeZone to “(UTC+01:00) Brussels, Copenhagen, Madrid, Paris” If you use the **%ANALOGUECLOCKTIMEZONE** VClock Variable, the shorter TimeZone ID will be displayed (“Romance Standard Time” in this case).

HTTP Interface in Detail

If you hit the clock with a webbrowser on the http port (default 18513), you get the following screen:



VClock

Pass parameters to the webpage to send VClock a Salvo, for example http://localhost:18513/VClock?Salvo=TOPLEFTFLASH

Or send a semicolon seperated list of Commands to process, for example http://localhost:18513/VClock?Command=GPI=1H;GPI=2H

HTTP Posts are assumed to be commands. So a simple form such as:

```
<form method = "POST" action = "http://localhost:18513/VClock">
  <p> TopCaption:
  <input type="text" name="TopCaption" value="Test Caption">
  <p> Enter a Salvo to trigger:
  <input type="text" name="SALVO" value="TOPLEFTFLASH">
  <input type="submit" name="Send" value="Submit">
</form>
```

TopCaption: Test Caption Salvo: TOPLEFTFLASH Submit

...will set the top caption of the clock to "Test Caption" and trigger a Salvo matching "TOPLEFTFLASH". (Try it above!).

Or get curent config with http://localhost:18513/VClock?Command=ShowConfig

Or get curent status as an RSS feed with http://localhost:18513/VClock?Command=RSS

Or request a screenshot of the clock (clock needs to be visible on the target machine) with http://localhost:18513/VClock?Command=Screenshot

Visit http://www.voceware.co.uk for more information

The links are self-explanatory really:

If you do a HTTP Get of Salvo=<anything> then you can set an entry in the salvo table to match "Salvo=<anything>" to trigger a set of commands. (in fact any incoming data will be treated as a salvo unless it specifies a command (Command=xxxxx)).

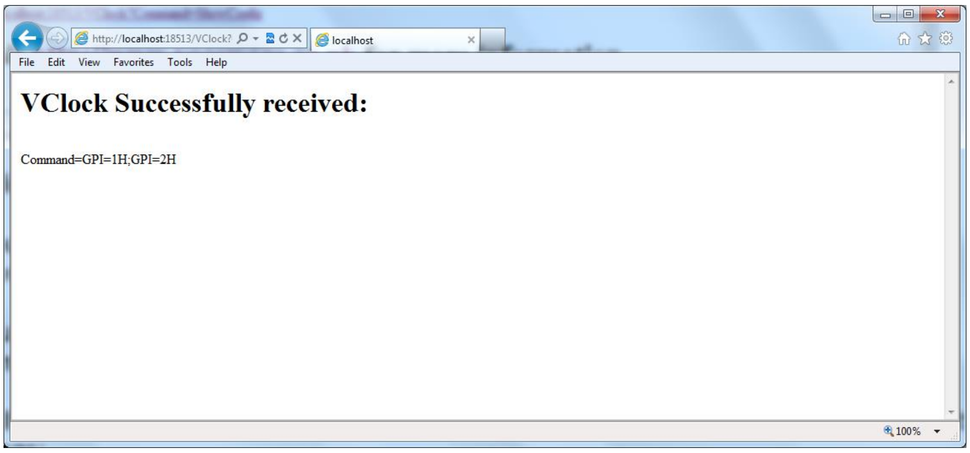
If you pass Command=<setofcommands>, you can bypass the salvo table and send commands directly to the clock.

With HTTP Posts, it is the reverse. The data is assumed to be a command unless the parameter is called SALVO. For example the simple form below shows an example of each:

```
<form method="POST" action="http://localhost:18513/VClock">
  <p>TopCaption: &nbspp;
  <input type="text" name="TopCaption" value="Test Caption">
  <p>Enter a Salvo to trigger: &nbsp;
  <input type="text" name="SALVO" value="TOPLEFTFLASH">
  <input type="submit" name="Send" value="Submit">
```

</form>

In each case the webpage will return a success page:

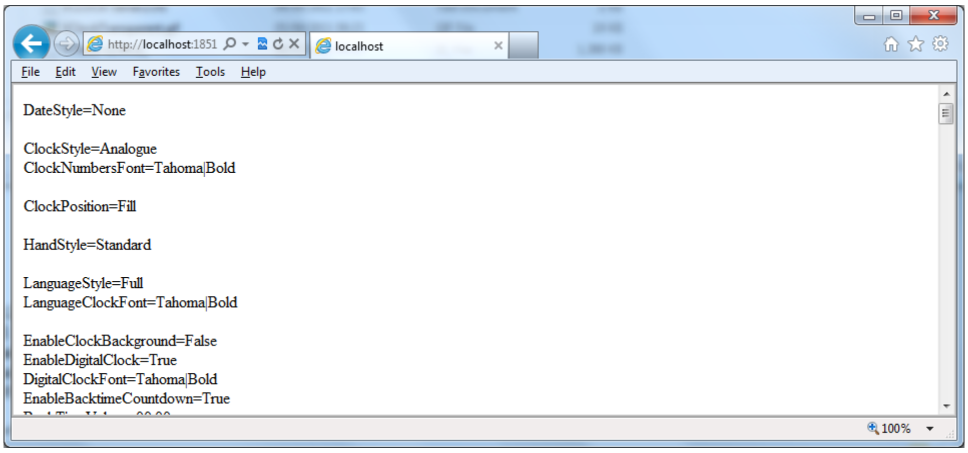


http://localhost:18513/VClock?
localhost
File Edit View Favorites Tools Help
100%

VClock Successfully received:

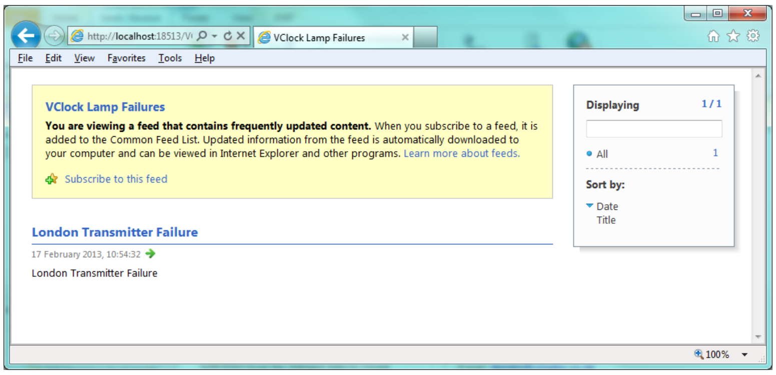
```
Command=GPI=1H;GPI=2H
```

If you pass Command=ShowConfig, the webpage will return a full list of the current settings:



```
DateStyle=None

ClockStyle=Analogue
ClockNumbersFont=Tahoma|Bold

ClockPosition=Fill

HandStyle=Standard

LanguageStyle=Full
LanguageClockFont=Tahoma|Bold

EnableClockBackground=False
EnableDigitalClock=True
DigitalClockFont=Tahoma|Bold
EnableBacktimeCountdown=True
```

If you pass Command=RSS, the webpage will return an RSS feed of any lamps that are turned on with LampErrorText set:



VClock Lamp Failures

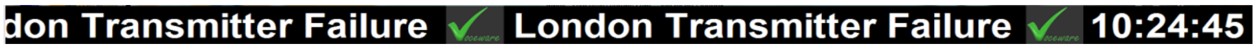
You are viewing a feed that contains frequently updated content. When you subscribe to a feed, it is added to the Common Feed List. Updated information from the feed is automatically downloaded to your computer and can be viewed in Internet Explorer and other programs. Learn more about feeds.

London Transmitter Failure

17 February 2013, 10:54:32

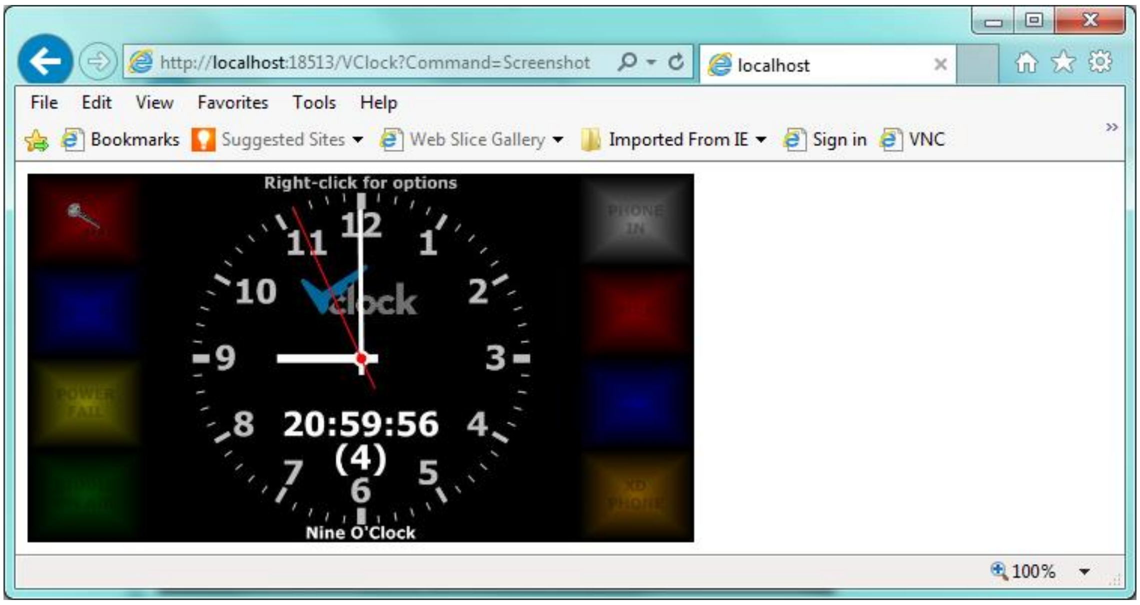
London Transmitter Failure

This is useful to feed data to VClocks sister application, VScroll, which scrolls messages across a screen:



don Transmitter Failure London Transmitter Failure 10:24:45

And finally (gimmicky), if you pass Command=Screenshot then the app will take a screenshot of its screen position and return it in the browser:


http://localhost:18513/VClock?Command=Screenshot
localhost
File Edit View Favorites Tools Help
Bookmarks Suggested Sites Web Slice Gallery Imported From IE Sign in VNC
Right-click for options
PHONE IN
POWER FAIL
XD PHONE
20:59:56
(4)
Nine O'Clock
100%

Appendix A – Full Macro List

Macros can be separated by semicolons, or by putting them on a separate line.

ClockBackgroundImage=[FILENAME.GIF | \\PATH\TO\FILENAME.GIF] (ideally square graphic, centered on clock face. Blank means none,
ClockBackgroundImageSizePercentage=100 filename alone assumes it is in the config folder)

DateStyle=[Full | DayOfMonth | None] (this is the display at the 3 o'clock position of the clock face)

ClockStyle=[Analogue | Digital | LargeDigital | None | WallOfLamps] (analogue still allows a digital display as well, but setting style to digital makes it larger and removes the analogue components. LargeDigital makes it even larger and removes the logo. None does not draw the clock face at all!, WallOfLamps is a fixed layout of lamps.
ClockNumbersFont=FontName|[Bold, Italic, Strikeout, Underline] (For example "Tahoma|Bold")

ClockNumbersSizePercentage=100 (percentage of "normal" size as auto calculated by VClock. > 100 makes larger, <100 smaller.
ClockNumbersStyle=[All | Quadric | Seconds| None] (All is default. Quadric displays only 12,3,6,9)

ClockPosition=[Fill | Left | Right | Top | Bottom] (Position of clock within the free space. Fill will center it)
ClockPositionMode=[Automatic | Manual]
ClockPositionCenter=X Y (Position of center of clock face as percentage of screen if Mode is Manual)
ClockRadiusSizePercentage=X (Radius of clock face as percentage of calculated size, if Mode is Manual)

ClockHandStyle=[Standard | Long | StandardThick | LongThick | Sharp | None] (style of the hour and minute hands. Standard is the default)

ClockSecondHandStyle=[Standard | Long | StandardThick | LongThick | Sharp | None] (Standard is default – meaning it shows a second hand)

ClockRTCTraceMaxSeconds=59 (Trace behind the second hand when PDMDelay is non-zero will not show if offset is greater than this value)

ClockRTCTraceBrightness=100 (How bright the trace should be. 0=transparent 255=solid)

DMXIdleState=DMX=1:Colour=Red/500+10:255,0,0 (Set of commands split by + to send when DMX=Idle is triggered – Useful to fall back to a variable "standard" colour for example when Mics are turned off, after changing the colour when they were turned on). Can also use a memory Slot - DMX=%GETMEM(S1Default)

DotStyle= [Seconds | SolidSeconds | Minutes | SolidMinutes | StopwatchMinutes | SolidStopwatchMinutes | None]
(None is default. Solid does not show a dot for unlit seconds)

DotZeroAlwaysOn=[True|False] When dots are shown around the clock face, this setting turns the very top one always on or always off (VClock traditionally was always off so showed the correct number of dots for the seconds. Older hardware clocks are always on. Default is False.

AnalogueClockTimeZone=[LOCAL | LOCAL+01:00:00 | GMT | GMT-01:00:00 | UTC | StandardTimeZone] (default is LOCAL)

LanguageStyle=[Full | Numbers | Custom | Digital | Date | None |
English | French | German | Italian | Spanish | Dutch | Danish] (digital shows a digital clock which can be preferred to the standard location within the clock face. Date shows the date in the format "Day, DD/MM/YY". Custom can be used to use either VClock Variables such as day/month/year/hour/minute /second/dayofweek, or to translate the verbose clock to other languages or dialects. Default is English.
LanguageClockFont=FontName|[Bold, Italic, Strikeout, Underline]
LanguageClockSizePercentage=100 (percentage of "normal" size as auto calculated by VClock. > 100 makes larger, <100 smaller.
LanguageClockTimeZone=[LOCAL | LOCAL+01:00:00 | GMT | GMT-01:00:00 | UTC | StandardTimeZone] (default is LOCAL)

EnableClockBackground=[True | False] (disables the inner/outer colours and the surrounding circle)

EnableDigitalClock=[True | False] (shows a digital clock within the analogue clock face)

DigitalClockFormat=HH:mm:ss (sets format of Digital Clock face, using standard DateTilme formatting variables)
DigitalClockFont=FontName|[Bold, Italic, Strikeout, Underline]
DigitalClockSizePercentage=100 (percentage of "normal" size as auto calculated by VClock. > 100 makes larger, <100 smaller.
DigitalClockTimeZone=[LOCAL | LOCAL+01:00:00 | GMT | GMT-01:00:00 | UTC | StandardTimeZone] (default is LOCAL)

EnableStopwatch=[True | False] (shows a stopwatch (in place of the digital clock if you are using an analogue clock face)
StopwatchFont=FontName|[Bold, Italic, Strikeout, Underline]
StopwatchSizePercentage=100 (percentage of "normal" size as auto calculated by VClock. > 100 makes larger, <100 smaller.

StopwatchMode=Up | Down (direction the stopwatch counts in)

```
EnableLanguageClock=[True | False]                       (deprecated.  Use LanguageStyle=None to achieve the same)

EnableBacktimeCountdown=[True | False]                   (backtimes to set junctions, defaults to top of hour)
BackTimeValues=00:00,15:00,30:00,45:00                   (mm:ss or hh:mm:ss separated by comma, default "00:00")

NoOfLampsPerSide=[2-8]        (default is 4.  This is the number down the left side (or across the top).  there are 32 lamps max, but only
NoOfLampsPerSide x NoOfRowsOfLamps are shown)
NoOfRowsOfLamps=-[1-4]        (default is 2.  This is the total number of rows (or columns!) of lamps.  If set to 1 the area used by the clock face
                                                         acquires the extra space that the 2nd row of lamps would have taken.
LampsPosition=[Fill | Left | Right | Top | Bottom | None]          (Fill will put half on each side (or above/below) the clock face, depending on
screen res.  None will disable lamps)

LampXState=[Off | On | Solid | SolidDim | Flash | OnOff |OnOffSolid | OnOffFast | OnOffFastSolid |
OnDisabled | OnDisabledSolid | Phone | PhoneSolid | Disabled                    (X is replaced by 1 to 32 !)
LampXTimeout=0                                           (0 is disabled, or set to no of seconds before triggering the
LampXDisabled                                            Salvo)
LampXShape=[Circle | Square | Rectangle]
LampXCurvePercentage=[0…100]                             (Sets the curve of the corners of non-circular lamps.  0 disables)
LampXOffStateBrightness=[1..255]                         (0 is off, 255 is full brightness)
Lamp3BorderColour=White                            (creates a border around the lamp.  Colour can be set by name or
Lamp3BorderFollowsLampBrightness=True                    number, size percentage defines how thick the border is (0 disables).
Lamp3BorderSizePercentage=5                              FollowsLampBrightness dims the border when the lamp is dimmed).
LampXBackTimeValues=00:00,15:00,30:00,45:00              (mm:ss or hh:mm:ss separated by comma, default "00:00")
LampXCaption=TITLE\nHERE;                                (\n means new line, %COUNTER gets replaced with MM:SS counter)
LampXCaptionColour=Black;                          (can also use Number instead of Name)
LampXCaptionFont=FontName|[Bold, Italic, Strikeout, Underline]
LampXCaptionSizePercentage=100            (percentage of "normal" size as auto calculated by VClock. > 100 makes larger, <100 smaller.
LampXColour=Red;                                   (can also use Number instead of Name)
LampXErrorText=List|Of|Messages                          (this is the message sent to the RSS feed when the lamp is on)
LampXLogo=[FILENAME.GIF | \\PATH\TO\FILENAME.GIF]        (If filename is specified, it disables the LampCaption.  Filename
on its own assumes file is in application programdata folder otherwise specify full path.  overrides Caption UNLESS %COUNTER is specified in the
caption)
LampXLogoSizePercentage=[Y]    (Y can be any percentage of the lamp size.  50% isDefault.  Can go > 100.  Negative values invert the image.
LampXLogoFollowsLampStatus=[true|false] (if true, the logo turns off when the lamp is off, and flashes on/off when the lamp is flashing on/off)
LampXGPO=[n]                                             (0 disables.  If set, a GPO follows the status of the lamp)
LampXLogEventOn=Message To Log                           (Writes a log event whenever the lamp is turned "on")
LampXLogEventOff=Another Message                         (… and "off"!)
LampXOnCommand=<VClockCommand>                           (sends a command to VClock whenever the lamp transitions to "on")
LampXOffCommand=<VClockCommand>                          (sends a command to VClock whenever the lamp transitions to "off")
LampXPositionMode=[Automatic | Manual]
LampXPosition=X Y W H                                    (if PositionMode is manual this sets the position and size of the lamp
as a percentage of the overall screen)
LampXTimeZone=[LOCAL | LOCAL+01:00:00 | GMT | GMT-01:00:00 | UTC | StandardTimeZone]          (default is LOCAL)

VoiceSpeed=[-10..10]                                     {sets the speed and volume of text to speech using the SPEAK=
VoiceVolume=[0..100]                                     command)

VLCxAutoStart=[True | False]                             (autostart playing when content is changed, and at VClock start)
VLCxEnabled=[True | False]                               (if disabled, player is not shown on-screen)
VLCxLoop==[True | False]                                 (loop at the end of the video, back to the start)
VLCxMRL=[C:\test.mp4 | http://127.0.0.1/stream | rtsp://127.0.0.1/stream ] (URL or file path to a video or stream to play)
VLCxMRLOptions=                                          (extra options required such as Webcam name when using dshow://)
VLCxPosition= X Y W H                                    (position of Player on-screen as percentages of overall Clock size)

AnalogueInputXScalingMaths=X/655.35                      (maths to scale read value (X) – X is 0-65535)       (up to 8 inputs)
AnalogueInputXDecimalPlaces=2                            (number of decimal points to use for the result of the Scaled value)
AnalogueInputXLogEventChangeValue=A Message to Log       (Logged when the SCALED value changes)

//All colours can be defined by Name ("ByName") or by Argb Number ("ByNumber")
BackgroundColour=Black
ClockInnerColour=SkyBlue
ClockOuterColour=SteelBlue
ClockCaptionColour=Black
ClockHandColour=Black
ClockNumbersColour=Black
ClockSecondHandColour=Red
ClockTickColour=Black
DotColour1ByName=Red                                     (DotColour1 is used for dots at 1-10 seconds.  DotColour2 is 11-20,
DotColour2ByName=Red                                      DotColour3 is 21-35 and DotColour4 is 36-60 seconds).
DotColour3ByName=Red
DotColour4ByName=Red
DigitalClockColour=White
StopwatchColour=White
LanguageClockColour=White
TopCaptionColour=Yellow
MiddleCaptionColour=Yellow
```

```
BottomCaptionColour=Yellow
Opacity=1  Sets the transparency of the clock.  Values are 0.0 (hidden) to 1.0 (fully visible).
Blankscreen=[True | False]          If set to true draws only the background colour (no clock or lamps). Intended as a screensaver.

//TopCaption, MiddleCaption and BottomCaption can each be a string or a collection of strings separated by "|" (shift and \).
//If the latter it will rotate based on CaptionInterval
//Each can also contain VClock Variables such as %dow, %customdow %day, %month, %year, %hour, %12hour, %min, %sec , %ampm, %date %time
%english %englishnumbers %custom %COUNTER

TopCaption=Visit us at www.voceware.co.uk|Call us on 0843 289 1443|%hour:%min:%sec - %dow %day/%month/%year
TopCaptionFont=FontName|[Bold, Italic, Strikeout, Underline]
TopCaptionSizePercentage=100   (percentage of "normal" size as auto calculated by VClock. > 100 makes larger, <100 smaller.
TopCaptionInterval=5
TopCaptionTimeZone=[LOCAL | LOCAL+01:00:00 | GMT | GMT-01:00:00 | UTC | StandardTimeZone]                    (default is LOCAL)

MiddleCaption=
MiddleCaptionFont=FontName|[Bold, Italic, Strikeout, Underline]
MiddleCaptionSizePercentage=100            (percentage of "normal" size as auto calculated by VClock. > 100 makes larger, <100 smaller.
MiddleCaptionInterval=5
MiddleCaptionTimeZone=[LOCAL | LOCAL+01:00:00 | GMT | GMT-01:00:00 | UTC | StandardTimeZone]                 (default is LOCAL)

BottomCaption=
BottomCaptionFont=FontName|[Bold, Italic, Strikeout, Underline]
BottomCaptionSizePercentage=100            (percentage of "normal" size as auto calculated by VClock. > 100 makes larger, <100 smaller.
BottomCaptionInterval=5
BottomCaptionTimeZone=[LOCAL | LOCAL+01:00:00 | GMT | GMT-01:00:00 | UTC | StandardTimeZone]                 (default is LOCAL)

ClockTickStyle=[All | Twelve | Quadric | TwelveDots | TwelveSolidDots | QuadricDots | QuadricSolidDots | TwelveDotsOutside |
TwelveSolidDotsOutside | QuadricDotsOutside | QuadricSolidDotsOutside |AllInside | TwelveInside | QuadricInside |
AllWithSecondColours | TwelveWithSecondColours | QuadricWithSecondColours | AllInsideWithSecondColours |
TwelveInsideWithSecondColours,|QuadricInsideWithSecondColours | AllWithMinuteColours | TwelveWithMinuteColours |
QuadricWithMinuteColours | AllInsideWithMinuteColours | TwelveInsideWithMinuteColours,|QuadricInsideWithMinuteColours
| None]
(style of the "ticks" (or dots) around edge of clock.  All is Default.)

//ClockLogo must point to an image file (in programdata folder, or supply full path).    If specified, the ClockCaption is not used.
ClockCaption=VClock                                           (\n means new line, %COUNTER gets replaced with MM:SS counter)
ClockCaptionFont=FontName|[Bold, Italic, Strikeout, Underline]
ClockLogo=                                                    (overrides Caption UNLESS %COUNTER is specified in the caption)
ClockLogoSizePercentage=Y                                     (Y can be any number – a percentage of 50% of the clock face
size.  50% is Default.  Can go > 100.  Negative values invert the
image.
(ClockCaption and ClockLogo are disabled when the license is invalid).

//OverallBackgroundImage will draw an image (stretched to the full size of the clock and lamp area).  Leave blank to use BackgroundColour
OverallBackgroundImage=[FILENAME.GIF | \\PATH\TO\FILENAME.GIF]
OverallBackgroundImageSizePercentage=100

InstantAlarmFile=alarm.wav

WebBrowserURL=http://www.voceware.co.uk/VClockWebBrowserExample.html
WebBrowserPosition=Right        (options are Left/Right/Top/Bottom/Fill/None.  Default is Right, None disables Browser)
WebBrowserFullScreen=[True | False]                           (Overrides WebBrowserPosition and fills VClock scrren area with the
                                                                 Browser)
WebBrowserZoomPercentage=100                                  (default is 100 (percent))

LogFile=C:\ Logs\%M-%Y.LOG                 (path to write Log File to when using LogToFile or various Event entries.  Can use VClock Variables)
LogEntryFormat=%day/%month/%year %hour:%min:%sec: %ENTRY      (format of each line.  %ENTRY is replaced with the message passed)

GlobalTimeOffset=[TimeSpan format]                            (Sets the overall time offset across entire application.  Can also use
                                                              GlobalTimeOffsetInSeconds=[ss.ttt] )
```

The following commands can be used in a Salvo, but should NOT be used in a defaults.txt file:

```
PDMDelay=[ss.ttt]                                             (Sets the PDM Delay for use with %PDMDelay VClock Variable and a
                                                                      25-Seven  PDM / Cloudcast BDS.  PLUS license required).

//An Audio file can be played via a Salvo.  This is NOT available in the Property Panel.
Audio=alarm.wav                                               (set "Audio=" to cancel mid-play)
AudioLoop=alarm.wav                                           (set "AudioLoop=" to cancel playback)

//File: allows you to load several of the above settings from a text file (in the same way that defaults.txt works) with a single command.
//Be careful if including this in a file, as you may get into a loop if it refers to its own file, etc.
File=defaults.txt
```

Appendix B – Custom Clock / different languages

There are several built-in versions of the "Language Clock" available, but if none of these match exactly the way that you wish to display the time, then the custom clock mode is for you.

A text file describes how to display the time, and what words to use for each minute and each hour. This allows you to tell the time in English but in a different way to the built-in methods (for example to round up to the nearest 5 minutes, or just say "5 past the hour" instead of "5 past 2"), to display the date and / or time numerically in a different format to the standard, or to create an entirely new language (a German example is included as the default, and is detailed below).

Languages other than English work in a similar way. CUSTOMCLOCK_<Language>.txt is created for each language. You can edit these files in the same way as the CUSTOMCLOCK.TXT file to make small tweaks/corrections to the way the time is displayed.

See comments within the example below for detail, but the overview is that you specify the verbose word for every minute in the [MINUTES] section, every hour in the [HOURS] section, then create masks that use variables to generate the full string displayed. The variables then get replaced with the verbose word defined.

For example:

```
%H:01-%H:04=%%H Uhr %%M
```

%H will be replaced with the numeric version of the current hour. So if you are in the 2pm hour, the above line would become:

```
14:01-14:04=%%H Uhr %%M
```

So if the time is currently between 14:01 and 14:04 (lets say 14:03) then this line is a match and the string displayed would be generated from the mask:

```
%%H Uhr %%M
```

%%H will be replaced by the verbose version of the current hour (%%I is the next hour, %%M is the current minute and %%N is the next minute). So the example becomes:

```
zwölf Uhr drei
```

The final setting, in the [OFFSET] section, is an offset value. This is the number of seconds past the minute that VClock treats it as the "next" minute.

So as an example, the default behaviour is with the offset set to 30. If the time is 15:35:28 then we treat the time as 15:35:00 and say the time is "Twenty Five to Four". When the time becomes 15:35:30, we treat it as 15:36:00 and say the time is "Twenty Nine minutes to Four". So in effect the clock rounds to the nearest minute with this setting. Setting the offset to 0 would say “One Minute to Twelve”, even when the time is 11:59:59.

The default CustomClock.txt file is below with comments:

```
//Format is %H:MM-%H:MM=String
// Where String can contain
//   %%H (hour, as specified in [HOURS] section),
//   %%I (following hour, as specified in [HOURS] section),
//   %%M (minute, as specified in [MINUTES] section),
//   %%N (minutes to next hour, as specified in [MINUTES] section),
//
//   %dow (day of week - in English)
//   %day, %month, %year, %hour, %min, %sec (numeric)
//   %ordinalday (day of month in ordinal format (1st, 2nd, etc) – only works in English.
//   %moy (month of the year in words) – will return the month in the OS language.
//
// First match is chosen. so put specific things such as 00:00-00:00="Midday" before the generic %H:00-%H:00 = %H O'Clock

//The below shows the date and time numerically when custom is selected.
//The rest of the file isn't used if this line exists as it matches all times.

%H:00-%H:59=%hour:%min:%sec - %dow %day/%month/%year

//Deleting the above line makes the rest of the file come into play... which will display the time in my best schoolboy German
//(apologies to all Germans!!).

00:00-00:00=Mitternacht
12:00-12:00=Mittag

%H:01-%H:04=%%H Uhr %%M
%H:05-%H:05=%%M nach %%H
%H:06-%H:09=%%H Uhr %%M
%H:10-%H:10=%%M nach %%H
%H:11-%H:14=%%H Uhr %%M
%H:15-%H:15=Viertel nach %%H

%H:16-%H:19=%%H Uhr %%M
%H:20-%H:20=%%M nach %%H
%H:21-%H:24=%%H Uhr %%M
%H:25-%H:25=%%M nach %%H
%H:26-%H:29=%%H Uhr %%M
%H:30-%H:30=halb %%I

%H:31-%H:34=%%H Uhr %%M
%H:35-%H:35=%%N vor %%I
%H:36-%H:39=%%H Uhr %%M
%H:40-%H:40=%%N vor %%I
%H:41-%H:44=%%H Uhr %%M
%H:45-%H:45=Viertel vor %%I

%H:46-%H:49=%%H Uhr %%M
%H:50-%H:50=%%N vor %%I
%H:51-%H:54=%%H Uhr %%M
%H:55-%H:55=%%N vor %%I
%H:56-%H:59=%%H Uhr %%M
%H:00-%H:00=%%H Uhr

[OFFSET]

//Number of seconds to add to time before working out what to display
// for example +30 would add 30 secs.  Then we would change the time on the bottom of the minute
// so for instance midnight could be displayed at 23:59:30 to 00:00:30, rather than 00:00:00 to 00:01:00

OFFSET=30

[HOURS]

//Format is number=verbose version of number

0=zwölf
1=ein
2=zwei
```

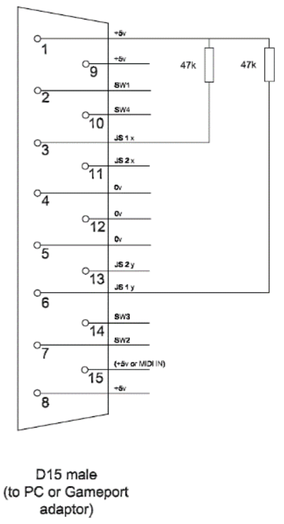
```
3=drei
4=vier
5=fünf
6=sechs
7=sieben
8=acht
9=neun
10=zehn
11=elf
12=zwölf
13=ein
14=zwie
15=drei
16=vier
17=fünf
18=sechs
19=sieben
20=acht
21=neun
22=zen
23=elf

[MINUTES]

//Format is number=english version of number

0=null
1=ein
2=zwei
3=drei
4=vier
5=fünf
6=sechs
7=sieben
8=acht
9=neun
10=zehn
11=elf
12=zwölf
13=dreizehn
14=vierzehn
15=fünfzehn
16=sechzehn
17=siebzehn
18=achtzehn
19=neunzehn
20=zwanzig
21=einundzwanzig
22=zweiundzwanzig
23=dreiundzwanzig
24=vierundzwanzig
25=fünfundzwanzig
26=sechsundzwanzig
27=siebenundzwanzig
28=achtundzwanzig
29=neunundzwanzig
30=dreißig
31=einunddreißig
32=zweiunddreißig
33=dreiunddreißig
34=vierunddreißig
35=fünfunddreißig
36=sechsunddreißig
37=siebenunddreißig
38=achtunddreißig
39=neununddreißig
40=vierzig
41=einundvierzig
42=zweiundvierzig
43=dreiundvierzig
44=vierundvierzig
45=fünfundvierzig
46=sechsundvierzig
47=siebenundvierzig
48=achtundvierzig
49=neunundvierzig
```

```
50=fünfzig
51=einundf�ünfzig
52=zweiundfünfzig
53=dreiundfünfzig
54=vierundfünfzig
55=fünfundfünfzig
56=sechsundfünfzig
57=siebenundfünfzig
58=achtundfünfzig
59=neunundfünfzig

[DAYS]
0=Sonntag
1=Montag
2=Dienstag
3=Mittwoch
4=Donnerstag
5=Freitag
6=Samstag
```

Appendix C – Games Port Wiring

A games port is intended to have a joystick or similar device connected, and we can pick up on the joystick's "buttons" to trigger VClock.

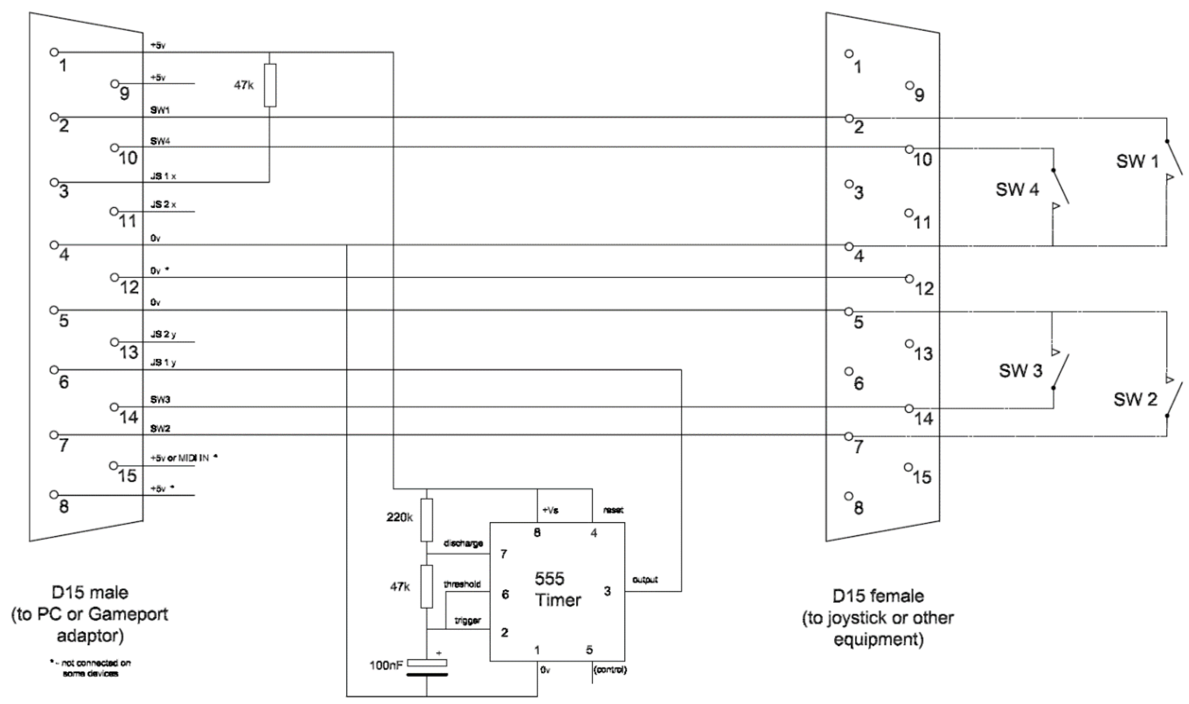
Some games ports allow you to simply short across pins to emulate a button press with no further wiring. Other models require some resistors on other pins to provide a small load, so that the port recognises that a joystick is attached, before it will detect the buttons being pressed.



NOTE: There is a limitation of game ports in all cases, whereby the state of the buttons only refresh when something changes. Therefore if VClock is started and a button is already pressed (for example a studio "on air" lamp), VClock will not light this lamp until another button (for example "mic live") changes state. This may be completely acceptable to you. But if not….

There is a hardware workaround for this – you can build a pass-through "dongle" to sit between your wiring and the games port, which incorporates a small 555 timer chip, which simulates moving one of the Axis of the joystick backwards and forwards. This is enough to trigger an update of the buttons to VClock, meaning that initial states of the buttons are read as soon as VClock starts. This dongle also incorporates resistors to emulate a joystick, allowing you to wire only the buttons on any model of games port.

If you want to make your own "dongle", then the circuit design follows:

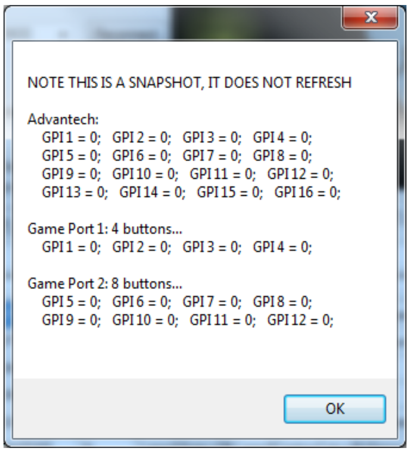


The detailed pin-outs of a games port are:

Pin	Description
1	+5v
2	Button 1 (connect to ground to trigger)
3	Joystick 1 – X axis position
4	Ground (use for Buttons)
5	Ground (use for Buttons)
6	Joystick 1 – Y axis position
7	Button 2 (connect to ground to trigger)
8	+5v on some games ports but not used on others. Best to ignore.
9	+5v
10	Button 4 (connect to ground to trigger)
11	Joystick 2 – X axis position
12	Ground on some games ports but not used on others. Best to ignore.
13	Joystick 2 – Y axis position
14	Button 3 (connect to ground to trigger)
15	+5v on some games ports but not used on others. Best to ignore.

VClock supports more than 1 games port being connected to the PC. It will append the 2nd set of buttons to the end of the first set, etc. For example:

If games port 1 is a 4-button port and games port 2 is an 8 button port, then you have a theoretical total of 12 buttons. The buttons of games port 2 will be treated as buttons 5-12:



```
NOTE THIS IS A SNAPSHOT, IT DOES NOT REFRESH

Advantech:
   GPI 1 = 0;  GPI 2 = 0;  GPI 3 = 0;  GPI 4 = 0;
   GPI 5 = 0;  GPI 6 = 0;  GPI 7 = 0;  GPI 8 = 0;
   GPI 9 = 0;  GPI 10 = 0;  GPI 11 = 0;  GPI 12 = 0;
   GPI 13 = 0;  GPI 14 = 0;  GPI 15 = 0;  GPI 16 = 0;

Game Port 1: 4 buttons...
   GPI 1 = 0;  GPI 2 = 0;  GPI 3 = 0;  GPI 4 = 0;

Game Port 2: 8 buttons...
   GPI 5 = 0;  GPI 6 = 0;  GPI 7 = 0;  GPI 8 = 0;
   GPI 9 = 0;  GPI 10 = 0;  GPI 11 = 0;  GPI 12 = 0;
```

Note that even though the second games port in the example above supports 8 buttons, only the first 4 are usable for VClock. It is important to realise though that if the devices were the other way around, you would be using GPI's 1, 2, 3, 4, 9, 10, 11 and 12 in VClock for the 8 buttons.

People have reported success with other, non-traditional USB-based Game Controller devices which can support many more than 4 x "buttons". For example the ION IED05 USB Drum Kit, which supports 10 x "drums" (which VClock sees as buttons). If Windows sees the devices as games controllers in Control Panel, and recognizes they have buttons, then VClock should see them too.

Appendix D – Interfacing with other equipment

Apart from using GPI contact closures, VClock accepts IP connections, makes IP connections to other equipment, and will accept RS232 Data. These are designed to be as flexible and generic as possible, matching simple strings sent to it on any port type against the salvo list in VClock and triggering when a match is found.

There is also special support for some equipment, to make it easier:

AEQ (Capitol and Forum Consoles)

VClocks IPClient can connect to the above consoles on port 2600, and receive status commands whenever a channel is turned on or off.

The commands are received in a bespoke format, for example:

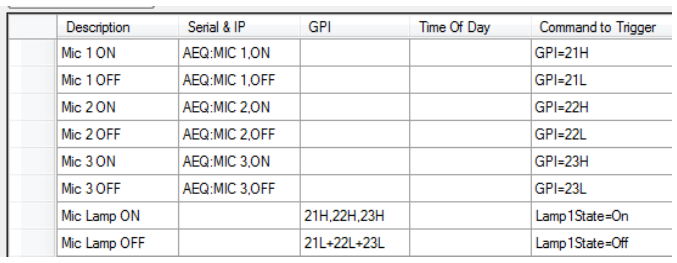
<0x53><0x01>MIC 1<0x00><0x01><0x4c>

…but VClock supports a simple way of referencing this in a Salvo. Simply use the format

AEQ:MIC 1,ON	AEQ~UniqueID:MIC 1,ON
AEQ:MIC 1,OFF	AEQ~UniqueID:MIC 1,OFF

(Where "MIC 1" is the name of the channel you are interested in). If no UniqueID is specified VClock will trigger on the command coming from ANY IPClient connection. The UniqueID will lock it to a single IPClient.

There is no overall "mic live" indication, so you will have to have a separate salvo for each mic channel, convert these to Virtual GPI numbers, then use GPI logic to create an overall Mic Live lamp. For example:



Description	Serial & IP	GPI	Time Of Day	Command to Trigger
Mic 1 ON	AEQ:MIC 1,ON			GPI=21H
Mic 1 OFF	AEQ:MIC 1,OFF			GPI=21L
Mic 2 ON	AEQ:MIC 2,ON			GPI=22H
Mic 2 OFF	AEQ:MIC 2,OFF			GPI=22L
Mic 3 ON	AEQ:MIC 3,ON			GPI=23H
Mic 3 OFF	AEQ:MIC 3,OFF			GPI=23L
Mic Lamp ON		21H,22H,23H		Lamp1State=On
Mic Lamp OFF		21L+22L+23L		Lamp1State=Off

Audionics eMM88 Audio Switcher

This is an 8x8 matrix mixing switcher, which can be controlled via a Web Interface. It is a powerful switcher, often used as a studio switcher to allow you to mix in split commercials, news services, etc. The one thing that it does not have is a GPI output, to display when a studio is on-air for example.

VClock can connect via the IP Client to the switcher, and poll it for the status of all of the matrix crosspoints. It can use these as triggers to light lamps on-screen, and pass the commands on to slave clocks too.

Configuration is simple.

- On the Settings Page, connect the IP Client to the EMM88, specifying the IP address and Port number to use (Port is usually 10303).
- Set the heartbeat to be EMM88:HEARTBEAT. This actually sends <0xff><0x01> once per second to request the status.

The switcher will then send the status of all crosspoints, once per second, to VClock. This looks like:

```
<0xfe><0x01><0x01><0x00><0x01><0x00><0x00><0x00><0x00><0x00><0x01><0x00>
<0x00><0x01><0x00>
<0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00
><0x00><0x00><0x00>
<0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00
><0x00><0x00><0x00>
<0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00
><0x00><0x00><0x00>
<0x00><0x00><0x00><0x00><0x00><0x00>
```

(<0xfe><0x01> is a header, followed by the 8 x 8 matrix position statuses (1 = on, 0 = off), starting with Output1/Input1, then Output1/Input2, etc across the grid).

To act on a trigger, set the Salvo to be EMM88:1,3,ON or EMM88:1,3,OFF (this example is Output 1/Input 3). If no UniqueID is specified VClock will trigger on the command coming from ANY IPClient connection. The UniqueID will lock it to a single IPClient. Use the format EMM88~UniqueID:1,3,ON or EMM88~UniqueID:1,3,OFF

A better way to interface to the EMM88 switcher is to use another of our utilities – AudionicsGateway. Go to www.voceware.co.uk for more info. Use of this to control VClock is free - it – makes a http post to VClock of any crosspoint changes, and allows the clock to interface to many emm88's. AudionicsGateway also provides an on-screen grid of all crosspoints and allows easy control of the emm88 via this grid or GPI closures.

Axia / Livewire – LWCP connection to an Axia Console

LWCP carries several commands that VClock can use and control. Connect to the console using an IPClient with the "LWCP" protocol (connects to the device on port 4010, and sets the IPClient Init String to "AXIA:INIT"). If the device has a password set, set the Init string to AXIA:INIT:password.

Element/Fusion Console Counters

The Element and Fusion consoles have a built in UpTimer and DownTimer. VClock can display these timers on any lamp or caption using the %UPCOUNTER, %UPCOUNTERMINIMUM, %DOWNCOUNTER and %DOWNCOUNTERMINIMUM VClock Variables. The UpCounter and DownCounter of VClock will mimic the built-in timers of the console – and starting/stopping the VClock counters will do the same to the Fusion ones. So Salvos can set pre-determined values for the Fusion DownTimer for example, as well as start/stop them.

Connect to the console with the "LWCP+TIMER" Protocol instead of "LWCP". Ensure only a single IPClient has this set.

Element/Fusion VMix inputs

Mix Engines contain typically 16 VMixes, each with 5 inputs. VClock has the ability to be triggered when each of these inputs is turned on/off, and VClock Plus can control them.

The Salvo format is:

LWCPVMIX:output,input,state or LWCPVMIX~UniqueID:output,input,state

Where output is 1-16, input is 1-5 and state is ON or OFF. If no UniqueID is specified VClock will trigger on the command coming from ANY IPClient connection. The UniqueID will lock it to a single IPClient.

Note you can also change the state of the inputs (with a "VClock Plus" license) with the command

LWCPVMIX=1,3,ON or LWCPVMIX~UniqueID=1,3,ON

(1 is the output, 3 is the input, ON is the state. Again the UniqueID will send it to a single IPClient otherwise it will be sent to ALL IPClients).

VMIX will work instead of LWCPVMIX, to maintain backwards compatibility.

Element/Fusion and IQ/IQx Buttons

VClock can trigger on button presses on panels attached to a Fusion/Element, and to button on expansion panels for the IQ/IQx. The Salvo format is:

LWCPBUTTON:<moduleNumber>,<buttonNumber>,<state>

(state = DOWN/ON/TRUE or UP/OFF/FALSE). LWCPBUTTON~UniqueID specifies which IPClient connection to use.

There are several commands that can be sent TO the buttons with the LWCPBUTTON= command, to change the indication of the button (on/off/flash, colour and text). See earlier section for details.

Element/Fusion Monitoring Panel Buttons

VClock can trigger on button presses on the 4 buttons on the monitoring panel of a Fusion. The Salvo format is:

LWCPMONBUTTON:<buttonNumber>,<state>

(state = DOWN/ON/TRUE or UP/OFF/FALSE). LWCPMONBUTTON~UniqueID specifies which IPClient connection to use.

There are several commands that can be sent TO the buttons with the LWCPMONBUTTON= command, to change the indication of the button (on/off/flash, colour and text). See earlier section for details.

Element/Fusion and IQ/IQx Profile Loading

VClock can trigger based on when a Profile is loaded on a console. The Salvo format is:

LWCPPROFILE:Profile Name (or LWCPPROFILE~UniqueID=ProfileName).

Element/Fusion and IQ/IQx "Mic Live"

VClock can trigger when either the Control Room or Studio Mic is live. The Salvo formats are:

LWCPCRMUTE:ON|MUTE|MUTED|TRUE or OFF|NORMAL|FALSE. LWCPCRMUTE~UniqueId: specifies which IPClient to use. Similarly LWCPSTMUTE: triggers for the Studio Mic Live (as opposed to the Control Room Mic Live).

Axia / Livewire – LWCPIQ connection to an Axia IQ/IQX/IQS Console

Axia IQ/IQX/IQS consoles, as well as having the standard LWCP port (4010), also have a special version on port 4040, aimed at console/surface control (selection of monitoring source, button presses, etc).

Connect to the console using an IPClient with the "LWCPIQ" protocol (connects to the device on port 4040, and sets the IPClient Init String to "AXIA:INIT"). If the device has a password set, set the Init string to AXIA:INIT:password.

VClock can then send/receive commands with the console.

To send, use the command

```
LWCPIQ=<object>,<property>,<command>
LWCPIQ~UNIQUEID=<object>,<property>,<command>
```

Any valid LWCP command can be sent using this format. Refer to Telos for a complete list. A couple of examples are:

```
LWCPIQ=crmo,ssel,1 (Control Room Monitoring, Source Selection, PGM1)
LWCPIQ=stmo,ssel,8 (Studio Monitoring, Source Selection, EXT2)
```

For Source Selection, 1-4 = PGM1=4, 7=EXT1 and 8=EXT2.

To trigger on a change from the console, use:

```
LWCPIQ:<object>,<property>,<command>
LWCPIQ~UNIQUEID:<object>,<property>,<command>
```

For example:

```
LWCP:crmo,ssel,1 (Control Room Monitoring, Source Selection, PGM1)
LWCP:stmo,ssel,!1 (Studio Monitoring, Source Selection, NOT PGM1)
```

(the ! symbol at the start of the value means NOT)

Axia / Livewire – LWRP Connection to an XNode or Console

Enable the IP Client, pointing at an Axia device such as a console or xNode (on Port 93).

Axia GPIO

You need to subscribe for GPO updates, so set the IP Client "Init String" to **AXIA:INIT**

Axia / Livewire GPI ports each have 5 separate GPIs. It isn't always the first one that you are interested in so a straight text match won't work. So VClock has special Salvo formats:

> LWO:<GPINumber>,<GPIMask> / LWO~UniqueID:<GPINumber>,<GPIMask>
> LWI:<GPINumber>,<GPIMask> / LWI~UniqueID:<GPINumber>,<GPIMask>

LWO is for GPOs (and is functionally the same as just "LW:"), LWI is for GPI. If no UniqueID is specified VClock will trigger on the command coming from ANY IPClient connection. The UniqueID will lock it to a single IPClient.

The GPINumber is in this case 8. The GPIMask is in the format xxxHx or xxxLx, where an x is ignored and a H or L has to match the same "bit" position of the incoming command.

So in the above example, LWO:8,Lxxxx and LWO:8,Hxxxx would be the triggers for the first 2, LWI:8,Lxxxx and LWI:8,Hxxxx for the second 2.

For info, to get "Mic Live" from an Axia IQ you should assign a logic port channel number in the show profile then assign this to one of the 4 (on a QoR16) or 8 (QoR32) GPIO ports. LWO:1,Lxxxx would be the trigger for this (and LWO:1,Hxxxx would be the "off" command) if you assigned it to GPIO Port 1.

Axia xNode Audio Matrix

Axia xNodes running firmware v2.0 or above can route audio in a matrix. VClock can monitor the status of the crosspoint and use these to trigger salvos.

Enable the IP Client, pointing at an Axia xNode, IP Port 93). You need to subscribe for Crosspoint updates, so set the IP Client "Init String" to **AXIA:INIT**. If the xNode has a password set, set the Init string to **AXIA:INIT:password**.

The matrix has 24 (mono) outputs (1-24) and 24 inputs (1-24). Each crosspoint can be on or off (in actual fact there can be a separate gain value for each crosspoint, but VClock just treats it as on or off). The Salvo format is:

LWMIX:output,input,state or LWMIX~UniqueID:output,input,state

(LWMIX: used to be XNODE:, which still works for backwards compatibility)

Where output is 1-24, input is 1-24 and state is ON or OFF. If no UniqueID is specified VClock will trigger on the command coming from ANY IPClient connection. The UniqueID will lock it to a single IPClient.

Note you can also change the crosspoints (with a "VClock Plus" license) with the command

LWMIX=1,3,ON or LWMIX~UniqueID=1,3,ON

(1 is the output, 3 is the input, ON is the state. Again the UniqueID will send it to a single IPClient otherwise it will be sent to ALL IPClients).

Valid states are ON, OFF or a db value in 1/10th of a db. Other special states are ONS/OFFS (stereo). These options automatically turn on/off the stereo pair of the crosspoint that you specify. So 1,3,ONS would turn on 1,3 AND 2,4. ONM/OFFM similarly turn on/off a mono source to a stereo output. So 1,3,ONM would turn on 1,3 and 2,3.

You can change multiple crosspoints at once by separating each crosspoint with a '+':

LWMIX=1,3,ON+2,4,ON+11,13,OFF+12,14,OFF

Changing Audio Routes

Connect to an XNode as usual (IPClient on port 93, AXIA:INIT as the Init string).

To change an audio destination (DST) you use the command

LWA=5,12345,Name

This sets the DST number 5 to Livewire channel 12345 and (if specified) changes the name to be "Name"). The last variable is optional. A negative LW channel is the ToSource, or a multicast address can be specified.

To trigger a salvo when this changes, simply use the trigger:

LWA:5,12345 (triggers when the destination is set to 12345)
LWA:5,239.192.48.57 (as above but using the multicast address)
LWA:5,!12345 (triggers when the destination is set to anything BUT 12345)
LWA:5,-12345 (triggers when the destination is set to the ToSource of 12345)

Axia / Livewire - Pathfinder memory slots

VClock can connect to Pathfinder using an IPClient connection, and set/get memory slots.

Enable the IP Client, pointing at the Pathfinder "SA Protocol" port (9500 by default). There needs to be a valid user/password set up in Pathfinder's File / User Database menu.

To set a memory slot (in this case setting "CHRIS" to "1"), send:

LOGIN USER PASSWORD<0x0d><0x0a>SMS CHRIS=1<0x0d><0x0a>

(<0x0d> is CR, <0x0a> is LF).

This can be particularly useful if you want to set a memory slot as an INIT command whenever the clock is restarted, to tell Pathfinder to refresh all of the GPIs using the GPIO method (see later).

To read (get) a memory slot (in this case "CHRIS"), send:

LOGIN USER PASSWORD<0x0d><0x0a>GMS CHRIS<0x0d><0x0a>

Pathfinder will then return a string including something like:

MemorySlot 1<0x09>CHRIS<0x09>1<0x0d><0x0a>

You could then set a salvo in VClock to use, say CHRIS<0x09>1

It is also possible to request multiple memory slot values, for example:

LOGIN USER PASSWORD<0x0d><0x0a>GMS CHRIS<0x0d><0x0a>GMS HEATHER<0x0d><0x0a>GMS THEO<0x0d><0x0a>

This could again be used as an INIT command, to set the initial state of all lamps when VClock starts (for example a "studio onair" lamp, which may already be active before VClock starts).

It could also be set as the "heartbeat" command for IPClient, which would send the command to Pathfinder once per second, and could be used to poll memory slots and act upon changes. But a better method is to use the GPIO Axia Node built into VClock (see "VClock becoming a Livewire GPIO Node" below).

Axia / Livewire - VClock advertising as a Livewire GPIO Node

One of the best ways to interface with Axia / Livewire (and also Broadcast Bionics' Phonebox (version 3 or above) and Caller One is to turn VClock into a Livewire GPIO Node.

This is achieved by enabling the IPServer, and setting the port to 93. Pathfinder, Phonebox, Caller One and other devices will then see the clock advertised on the network, and will be able to send it GPO closures. The salvo would then be set to LWO:<GPINumber>,<GPIMask> as above to trigger an event.

VClock will present 64 GPO's, each with 5 pins as above. So it can take in a total of 320 separate commands (more than I imagine will ever be required – but then it used to be 160 commands and someone managed to use them all up!).

Commands can be sent from VClock using LWO= and LWI=. If you want to send ONLY to the IPServer port, use the UniqueID IPSERVER (for example LWO~IPSERVER=1,Lxxxx)

Axia / Livewire - VClock receiving Livewire Multicast GPIO

Enabling the "Livewire Multicast GPIO" option on the settings page tells VClock to listen on the network for any GPIO multicasts. It will receive every GPIO to and from any Axia equipment on the network.

You tell VClock in Salvos which ones you are interested in, with a Salvo in the format

```
LWMC:<lwchannel>,<direction>,<pin>,<state>
```

For example

```
LWMC:12345,O,1,L
```

lwchannel is the multicast address of the GPIO. Direction is O (GPO) or I (GPI). Pin is the pin number (1-5) that you are interested in. Multicast GPIO only sends 1 pin at a time so there is no need for a mask in the format LxHxx above. State is the state of the pin (L – Low (on), H – High (off). At startup, VClock will query the multicast addresses referenced in Salvos to get their startup state.

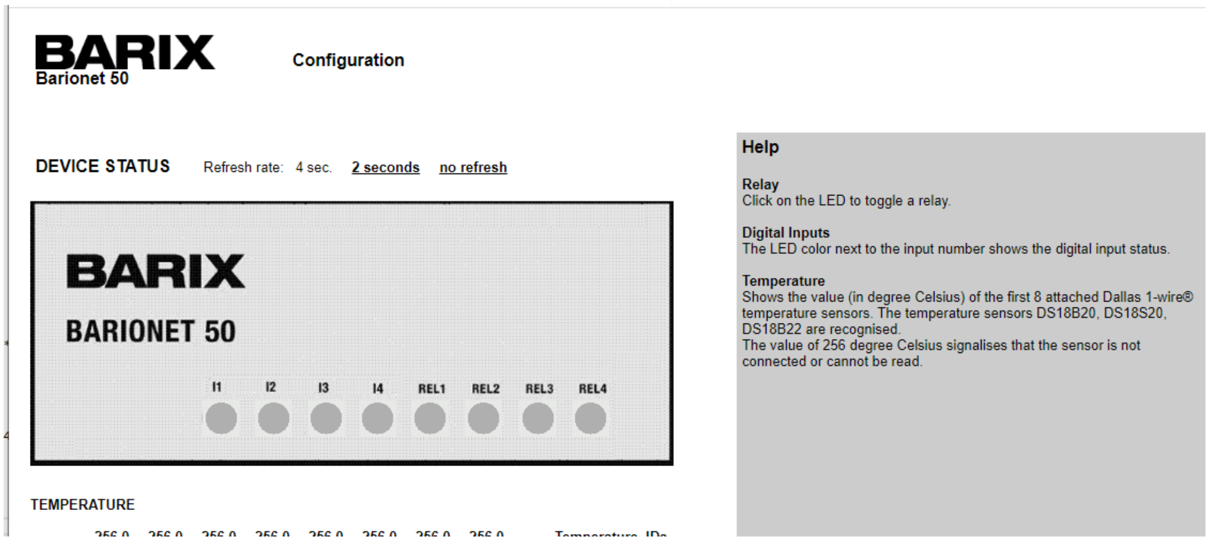
In versions of VClock previous to 4.7.7, the incoming Multicast GPIO was treated in the same way as other Axia GPIO (with LWI: or LWO:). This is still possible, and is enabled by default. It can be disabled in Settings. However, setting Salvos to look for this format will not query the GPIO states at startup. The option remains only for backward compatibility.

VClock can also send Multicast GPIO (with a VClock PLUS licence). See LWMC= above.

Barix Barionet 50 Network GPIO Device

VClock can receive GPI triggers from the Barionet 50, and (with a Plus licence) send GPO outputs as commands to it.

The Barionet obtains an IP address by DHCP by default. There is a Discovery Tool available on their website to find this address. You can then hit the unit with a Web Browser to configure it:



Help

Relay
Click on the LED to toggle a relay.

Digital Inputs
The LED color next to the input number shows the digital input status.

Temperature
Shows the value (in degree Celsius) of the first 8 attached Dallas 1-wire® temperature sensors. The temperature sensors DS18B20, DS18S20, DS18B22 are recognised.
The value of 256 degree Celsius signalises that the sensor is not connected or cannot be read.

You can see the status of inputs and outputs here.

Click on Configuration – There you can set a fixed IP address.

Under the Control Tab, set a TCP Command Port (9001 for example) and leave the Timeout at 0. Ensure "TCP Initial I/O state subscriptions" is set to "Local I/O", then press OK and reboot the unit.

In VClock, set up an IPClient to point at the Barionet's IP address and port. Select BARIX as the protocol, which will set "\r" as the separator.

Triggers for incoming commands are in the format

BARIX:<pin>,<state>

For example BARIX:3,1 (or BARIX:3,H or BARIX:3,ON) will trigger when GPI 3 is active.

With a PLUS licence, you can also control the GPO outputs. The command would be

BARIX=<pin>,<state>

For example BARIX=3,1 (or BARIX=3,H or BARIX=3,ON) to turn on output 3.

Blackmagic Smart Videohub series

VClock can interface with a Blackmagic Smart Videohub video switcher both to switch it, and to receive tallys from it.

Set an IPClient to connect to the router's IP address on port 9990. When it connects, the switcher will send the current state of all outputs to VClock. So there is no need for an INIT command.

Set up Salvos in the format

BMROUTE:10,15 (or BMROUTE~UNIQUEID:10,15)

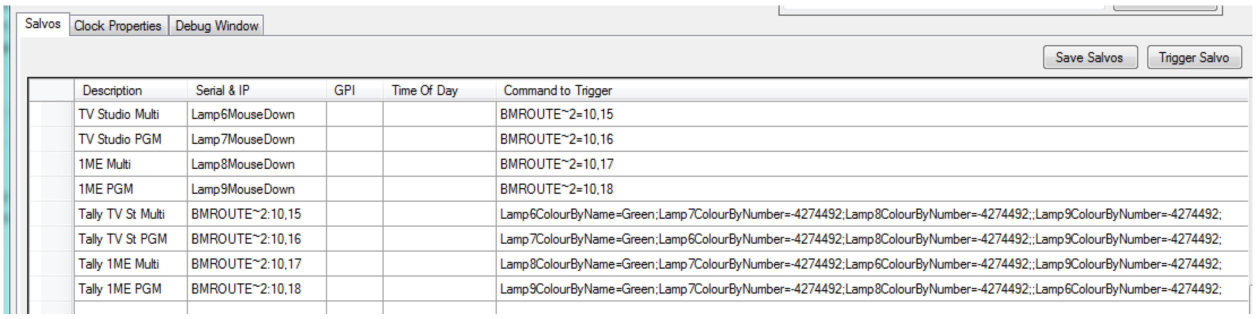
This will trigger when Output 10 selects Input 15. These numbers are based on starting at 1 (as opposed to the API which is zero based).

To switch the output to a different input, simply set up a command of

BMROUTE=10,20 (or BMROUTE~UNIQUEID=10,20)

This will tell the switcher to switch output 10 to input 20 immediately.

Below are some example Salvos, which allow you to click lamps (buttons) to switch the video source, then the tally back lights the relevant button light up as a full loop reconciliation of the action:



Description	Serial & IP	GPI	Time Of Day	Command to Trigger
TV Studio Multi	Lamp6MouseDown			BMROUTE~2=10,15
TV Studio PGM	Lamp7MouseDown			BMROUTE~2=10,16
1ME Multi	Lamp8MouseDown			BMROUTE~2=10,17
1ME PGM	Lamp9MouseDown			BMROUTE~2=10,18
Tally TV St Multi	BMROUTE~2:10,15			Lamp6ColourByName=Green;Lamp7ColourByNumber=-4274492;Lamp8ColourByNumber=-4274492;;Lamp9ColourByNumber=-4274492;
Tally TV St PGM	BMROUTE~2:10,16			Lamp7ColourByName=Green;Lamp6ColourByNumber=-4274492;Lamp8ColourByNumber=-4274492;;Lamp9ColourByNumber=-4274492;
Tally 1ME Multi	BMROUTE~2:10,17			Lamp8ColourByName=Green;Lamp7ColourByNumber=-4274492;Lamp6ColourByNumber=-4274492;;Lamp9ColourByNumber=-4274492;
Tally 1ME PGM	BMROUTE~2:10,18			Lamp9ColourByName=Green;Lamp7ColourByNumber=-4274492;Lamp8ColourByNumber=-4274492;;Lamp6ColourByNumber=-4274492;

When the config is read in from the switcher (and when values are updated), Memory slots are set/updated for the names of each source, destination, and the currently selected source for each destination. These can be accessed with %GETMEM(INPUTxNAME), %GETMEM(OUTPUTxNAME) and %GETMEM(OUTPUTxSOURCENAME). As well as the above BMROUTE: triggers, you can also trigger on SETMEM:OUTPUTxSOURCENAME and so on.

Broadcast Bionics - Caller One

Caller One can talk to an Axia GPIO Node, therefore it can also talk to VClock!

To configure it I would contact Broadcast Bionics directly, but the headlines are:

- In configuration settings, under Global / Advanced, check "Enable a GPO ring lamp", set the type to Axia, and set the IP Address of the VClock PC
- In each service, under Advanced, set the port and pin you want to use. Port 1 Pin 3 for example would be referenced in VClock as a Salvo of "LWO:1,xxLxx".
- Multiple services can use the same pin, or you can assign a different pin for each.

Broadcast Bionics - PhoneBOX (v3 or higher)

Phonebox can talk to an Axia GPIO Node, therefore it can talk to VClock!

To configure it I would contact Broadcast Bionics directly, but the headlines are:

- Hit your PhoneBOX Server with a webbrowser on port 3000, and log in.
- Add VClock to PhoneBOX as an External Interface (set Flash Interval to 500mS and the parameters are the IP address of VClock followed by a single comma)
- Add an output (as an example, output 3 is the 3rd pin of the first port that VClock advertises, so it would map to a Salvo of "LWO:1,xxLxx". Output 7 is the 2nd pin of port 2, so "LWO@2,xLxxx".)
- In System Settings / Services, edit the particular service you want the lamp to flash for. Under Service Outputs, add a new one – select the output you created above and set the type to "Ringing" (so it flashes when the phone rings!)

Broadcast Bionics – Phonebox SOLO

Solo is an older product and does not support Axia GPIO. But there is a way to get a ring detect out of it. You need to be on a 2019 build (dated 2019-06-21). Contact Broadcast Bionics if you are not on this version.

- In Tools / Admin / General Settings / GPIO, set the "Axia Element Console" to the IP address of your VClock PC
- In VClock, set the IP Server port to be port 4010
- There is a limitation whereby Solo will not attempt to reconnect once started, so VClock must be running first, and if VClock is restarted, Solo will also need to be restarted.

The trigger will then be in the format

PROVIDER#0 RINGING=ON
PROVIDER#0 RINGING=OFF

(each SIP account will increment the number, starting at #0 for the first account).

Broadcast Tools SRC-8 / SRC-16 / SRC-32 / ACS 8.2C Serial Devices

VClock can receive GPI triggers and send GPO triggers (VClock Plus required) from/to the Broadcast Tools GPIO devices. It can also change audio crosspoints on the ACS 8.2C.

The units are serial devices. They can be connected to VClock's Serial Port, or an IP to Serial device can be used and VClock can use an IP Client connection to the units.

The switcher needs to be set in Ascii mode. For example on the device (SRC-32), DIP switch 5 needs to be OFF to use Ascii Mode. DIP switch 1/2 set the Unit ID (all off is 0), 3/4 set the baud rate (all off = 9600 all on = 38400). JP09 reverses the RS232 send/receive pins. Set to NN to use a null modem cable to a PC.

VClock needs BT:INIT to be set as the Initialisation string for the connection. This will request the initial state of the GPIs, and on some devices set the length of a GPO pulse to be 300mS.

Match incoming GPI with BTI:[<deviceid>,]<pin>,<state>

BTI:0,32,H
BTI:32,L
BTI~UniqueID:0,32,H
BTI~UniqueID:32,H

UniqueID = "SERIAL" will send on serial port. Omitting a Unique ID will send to IP Clients and Serial ports with a BT device selected as the protocol.

DeviceID is inferred as 0 (the "main" device) unless specified.

State for GPO can be H (high/on), L (low/off).

Send GPO (VClock Plus licence required) using BTO=[<deviceid>,]<pin>,<state>

```
BTO=0,32,H
BTO=32,L
BTO=32,P
BTO~UniqueID=0,32,H
BTO~UniqueID=32,H
```

UniqueID = "SERIAL" will send on serial port. Omitting a Unique ID will send to IP Clients and Serial ports with a BT device selected as the protocol.

DeviceID is inferred as 0 (the "main" device) unless specified.

State for GPO can be H (high/on), L (low/off) or P (pulse, 300mS).

Make an audio crosspoint (VClock Plus Licence required) with BTA=[<devideid>,]<output>,<input>,<state>

```
BTA=0,1,5,ON
BTA=1,5,OFF
BTA~UniqueID=0,1,5,ON
BTA~UniqueID=1,5,ON
```

UniqueID = "SERIAL" will send on serial port. Omitting a Unique ID will send to IP Clients and Serial ports with a BT device selected as the protocol.

DeviceID is inferred as 0 (the "main" device) unless specified.

Output is 1 or 2

Input is 1 – 8

State can be ON or OFF

This will switch on/off individual inputs to each output, leaving the others in their original state (so mixing).

You can also use BTAS= (same parameters as BTA=) to do a SWITCH rather than a mix. This will force all other inputs OFF on the specified output.

Only the ACS 8.2C supports audio.

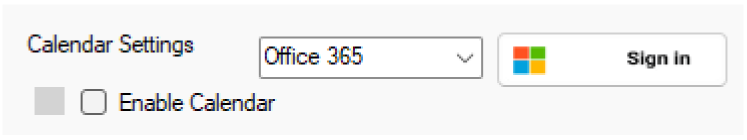
Calendars (Office 365 / Google / VCalendar)

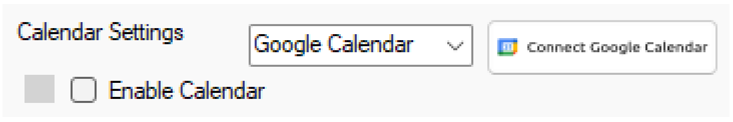
(Requires a VClock "Plus" license)

VClock can interface with an Office 365, Google or VCalendar, and poll every minute for the current entries in one or more calendars.

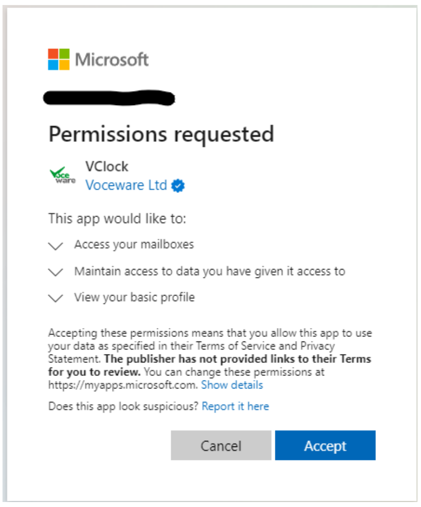
The use case is to display room or Studio bookings on a screen.

The user will first have to define which calendars to check, and **authenticate** to Microsoft or Google. In Tools/Settings, under the "Email and Calendar Settings" tab, select Office 365 or Google Calendar on the dropdown, then press the Sign In button. Alternatively just select VCalendar – no authentication is supported.





This will launch your browser and Microsoft/Google will walk you through the permissions requested, and ask you to give VClock permission to connect to the calendars. This would ideally be a user created with rights to see the relevant shared calendars, specifically for VClock, but can be any user with those rights – just be aware that you are providing VClock permission to ready your own calendar as well, if you use your own account.



Microsoft

[redacted]

Permissions requested

VClock
Voceware Ltd

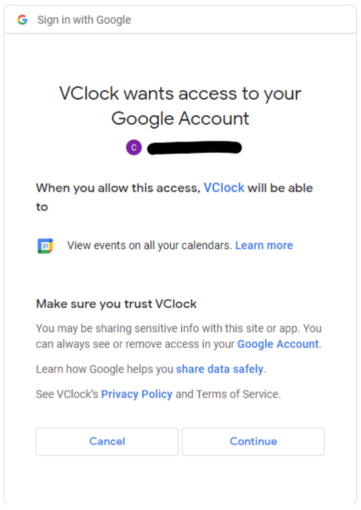
This app would like to:

- Access your mailboxes
- Maintain access to data you have given it access to
- View your basic profile

Accepting these permissions means that you allow this app to use your data as specified in their Terms of Service and Privacy Statement. **The publisher has not provided links to their Terms for you to review.** You can change these permissions at https://myapps.microsoft.com. Show details

Does this app look suspicious? Report it here

Cancel | Accept



Sign in with Google

VClock wants access to your Google Account

[redacted]

When you allow this access, VClock will be able to

View events on all your calendars. Learn more

Make sure you trust VClock

You may be sharing sensitive info with this site or app. You can always see or remove access in your Google Account.

Learn how Google helps you share data safely.

See VClock's Privacy Policy and Terms of Service.

Cancel | Continue

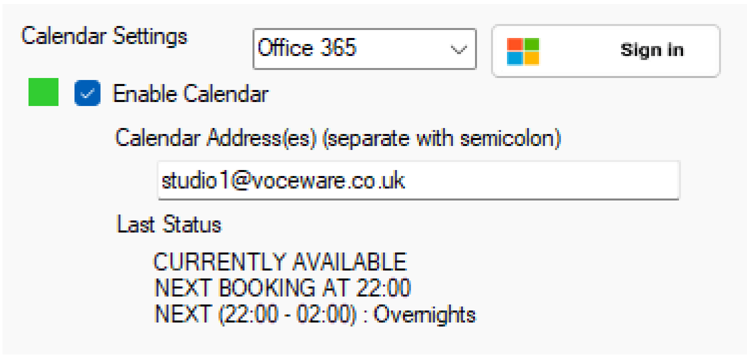
IMPORTANT – Do not close the browser page when authenticating until prompted to – if you close it prematurely, VClock will never recover and you will have to End-Task the VClock process.

Once you approve access, the web page will tell you it can be closed, then if you go back to VClock, the "latest update" on the settings page should be "Authorised Successfully".

VClock now stores a "refresh token", which is valid for 14 days. So long as VClock isn't left closed for more than 14 days, this token will be refreshed and re-stored, and VClock will maintain access to the calendars indefinitely.

Next enter a semicolon separated list of the address of each calendar you want to monitor (1 is fine too!). Then check "Enable Calendar".

The lamp next to the enable checkbox (and on the main page and top of the settings pages) will briefly go red, then should turn green once it connects and reads a calendar. It will flash black briefly at the top of every minute thereafter, as it re-checks the calendar for updates. The "Last Status" window will tell you the last information it retrieved (if you have more than 1 calendar, it will be for the last one in the list)


Calendar Settings: Office 365

Sign in

🟩 ✓ Enable Calendar

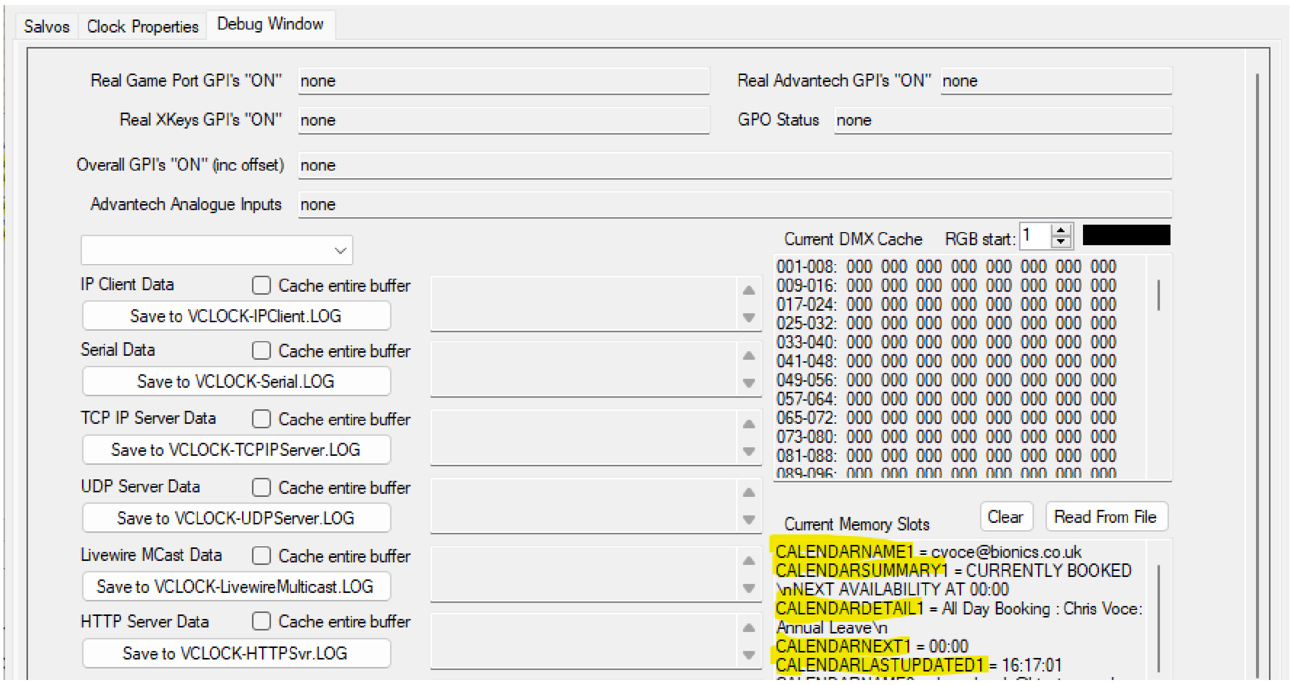
Calendar Address(es) (separate with semicolon)

studio1@voceware.co.uk

Last Status

CURRENTLY AVAILABLE
NEXT BOOKING AT 22:00
NEXT (22:00 - 02:00) : Overnights

VClock sets memory values for key information from the current and next booking. Back on the main configuration pages, you can see these in the "Debug Window" tab:



Real Game Port GPI's "ON": none

Real Advantech GPI's "ON": none

Real XKeys GPI's "ON": none

GPO Status: none

Overall GPI's "ON" (inc offset): none

Advantech Analogue Inputs: none

IP Client Data ☐ Cache entire buffer — Save to VCLOCK-IPClient.LOG

Serial Data ☐ Cache entire buffer — Save to VCLOCK-Serial.LOG

TCP IP Server Data ☐ Cache entire buffer — Save to VCLOCK-TCPIPServer.LOG

UDP Server Data ☐ Cache entire buffer — Save to VCLOCK-UDPServer.LOG

Livewire MCast Data ☐ Cache entire buffer — Save to VCLOCK-LivewireMulticast.LOG

HTTP Server Data ☐ Cache entire buffer — Save to VCLOCK-HTTPSvr.LOG

Current DMX Cache   RGB start: 1

```
001-008:  000  000  000  000  000  000  000  000
009-016:  000  000  000  000  000  000  000  000
017-024:  000  000  000  000  000  000  000  000
025-032:  000  000  000  000  000  000  000  000
033-040:  000  000  000  000  000  000  000  000
041-048:  000  000  000  000  000  000  000  000
049-056:  000  000  000  000  000  000  000  000
057-064:  000  000  000  000  000  000  000  000
065-072:  000  000  000  000  000  000  000  000
073-080:  000  000  000  000  000  000  000  000
081-088:  000  000  000  000  000  000  000  000
089-096:  000  000  000  000  000  000  000  000
```

Current Memory Slots   Clear   Read From File

```
CALENDARNAME1 = cvoce@bionics.co.uk
CALENDARSUMMARY1 = CURRENTLY BOOKED
\nNEXT AVAILABILITY AT 00:00
CALENDARDETAIL1 = All Day Booking : Chris Voce:
Annual Leave\n
CALENDARNEXT1 = 00:00
CALENDARLASTUPDATED1 = 16:17:01
```

These can be used in LampCaptions and so on.

> **CALENDARNAMEx** is the account name (x is the index of the account if you specified more than one in the semicolon-separated list)

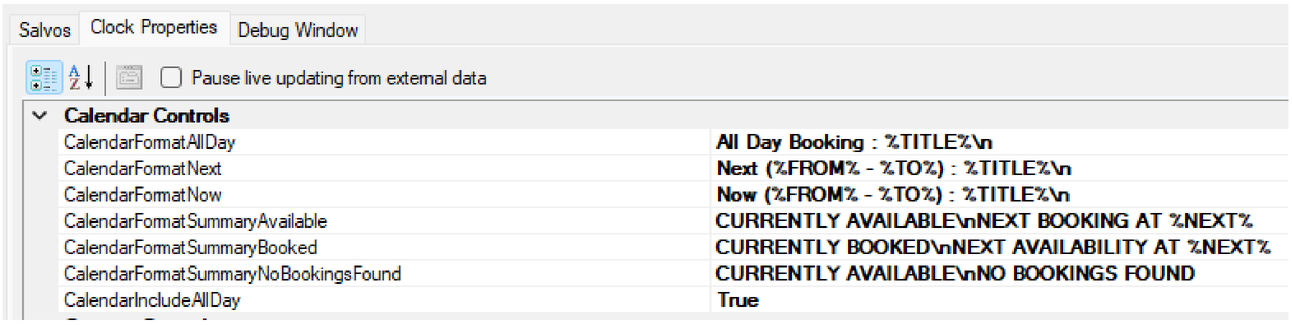
CALENDARLASTUPDATEDx is the time the calendar last received valid data

CALENDARNEXTx is the time the booking status next changes

CALENDARSUMMARYx is a summary of the current state (available/unavailable and when that next changes)

CALENDARDETAILx is the detail of the now/next booking (and any all-day bookings)

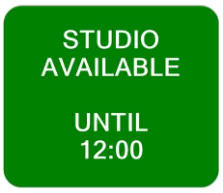
These messages are built from some templates, set up in the Clock Properties under the Calendar Settings section:



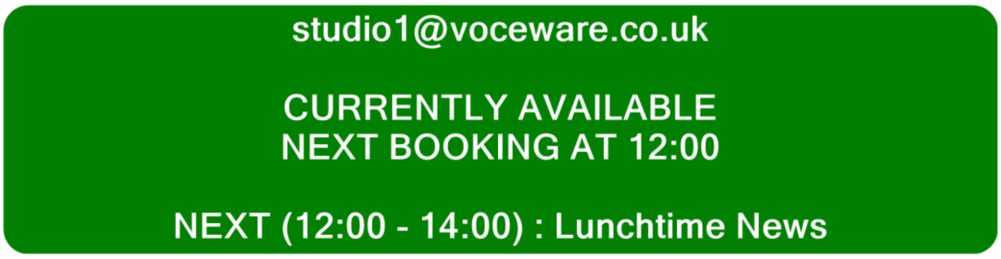
Calendar Controls	
CalendarFormatAllDay	All Day Booking : %TITLE%\n
CalendarFormatNext	Next (%FROM% - %TO%) : %TITLE%\n
CalendarFormatNow	Now (%FROM% - %TO%) : %TITLE%\n
CalendarFormatSummaryAvailable	CURRENTLY AVAILABLE\nNEXT BOOKING AT %NEXT%
CalendarFormatSummaryBooked	CURRENTLY BOOKED\nNEXT AVAILABILITY AT %NEXT%
CalendarFormatSummaryNoBookingsFound	CURRENTLY AVAILABLE\nNO BOOKINGS FOUND
CalendarIncludeAllDay	True

So for example, a LampCaption could be set to

STUDIO\nAVAILABLE\n\nUNTIL\n%GETMEM(CALENDARNEXT1)



%GETMEM(CALENDARNAME1)\n\n%GETMEM(CALENDARSUMMARY1)\n\n%GETMEM(CALENDARDETAIL1)



There are also some triggers that get fired on status changes:

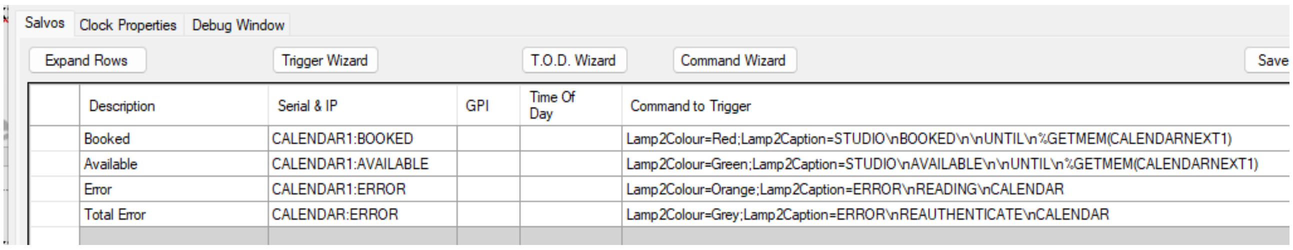
CALENDARx:BOOKED (triggered when the calendar currently has a booking)

CALENDARx:AVAILABLE (triggered when there is no booking)

CALENDARx:ERROR (triggered when there is a problem retrieving data for one of the calendars)

CALENDAR:ERROR (note no X – will trigger if there is an authentication error – could be used to clear the lamp caption and tell the user to re-authenticate)

These can be used to set the colour and text of the Lamp for example.

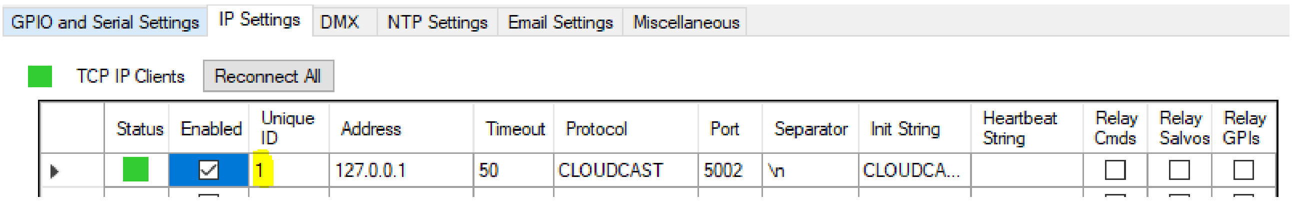


Description	Serial & IP	GPI	Time Of Day	Command to Trigger
Booked	CALENDAR1:BOOKED			Lamp2Colour=Red;Lamp2Caption=STUDIO\nBOOKED\n\nUNTIL\n%GETMEM(CALENDARNEXT1)
Available	CALENDAR1:AVAILABLE			Lamp2Colour=Green;Lamp2Caption=STUDIO\nAVAILABLE\n\nUNTIL\n%GETMEM(CALENDARNEXT1)
Error	CALENDAR1:ERROR			Lamp2Colour=Orange;Lamp2Caption=ERROR\nREADING\nCALENDAR
Total Error	CALENDAR:ERROR			Lamp2Colour=Grey;Lamp2Caption=ERROR\nREAUTHENTICATE\nCALENDAR

Cloudcast BDS (Requires a VClock "Plus" license)

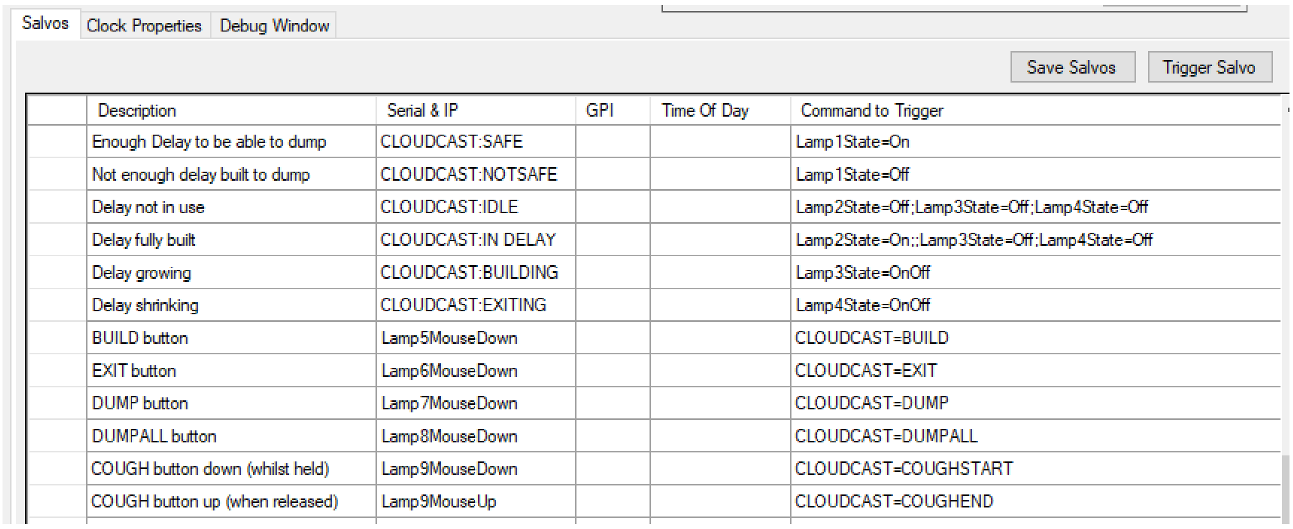
VClock can interface with a Cloudcast Broadcast Delay Service (BDS) and give similar functionality to a Telos PDM (see above).

The connection to the BDS to get the current delay is via an IP Client connection to port 5002 by default (the BDS Control Protocol - though this port number can be changed on the BDS webpage). An INIT string of "CLOUDCAST:INIT" will subscribe to updates for status changes from the BDS, The "Unique ID" of the IPClient connection must match the Delay's ID.



	Status	Enabled	Unique ID	Address	Timeout	Protocol	Port	Separator	Init String	Heartbeat String	Relay Cmds	Relay Salvos	Relay GPIs
▶	🟩	☑	1	127.0.0.1	50	CLOUDCAST	5002	\n	CLOUDCA...		☐	☐	☐

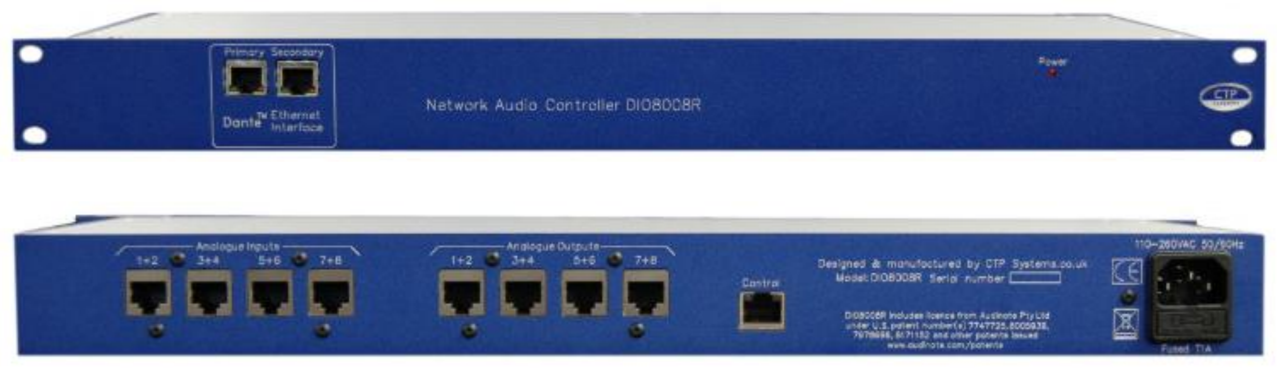
It's then down to a few simple Salvos to enable the control/tally lights:



Description	Serial & IP	GPI	Time Of Day	Command to Trigger
Enough Delay to be able to dump	CLOUDCAST:SAFE			Lamp1State=On
Not enough delay built to dump	CLOUDCAST:NOTSAFE			Lamp1State=Off
Delay not in use	CLOUDCAST:IDLE			Lamp2State=Off;Lamp3State=Off;Lamp4State=Off
Delay fully built	CLOUDCAST:IN DELAY			Lamp2State=On;;Lamp3State=Off;Lamp4State=Off
Delay growing	CLOUDCAST:BUILDING			Lamp3State=OnOff
Delay shrinking	CLOUDCAST:EXITING			Lamp4State=OnOff
BUILD button	Lamp5MouseDown			CLOUDCAST=BUILD
EXIT button	Lamp6MouseDown			CLOUDCAST=EXIT
DUMP button	Lamp7MouseDown			CLOUDCAST=DUMP
DUMPALL button	Lamp8MouseDown			CLOUDCAST=DUMPALL
COUGH button down (whilst held)	Lamp9MouseDown			CLOUDCAST=COUGHSTART
COUGH button up (when released)	Lamp9MouseUp			CLOUDCAST=COUGHEND

Under clock settings, ClockRTCTraceMaxSeconds should be set higher than your maximum delay to display the trace and ClockRTCTraceBrightness controls how transparent the trace is. %PDMDELAYn can be used in a caption to show the current delay (n = 0 – 3 and denotes the decimal places to round to, eg %PDMDELAY1).

CTP Systems DIO8008R Dante Audio Switcher


Primary
Secondary
Dante™ Ethernet Interface
Network Audio Controller DIO8008R
Power
CTP
Analogue Inputs
1+2
3+4
5+6
7+8
Analogue Outputs
1+2
3+4
5+6
7+8
Control
Designed & manufactured by CTP Systems.co.uk
Model: DIO8008R Serial number
110-260VAC 50/60Hz
Fused T1A

VClock can interface with a CTP Systems Dante Switcher and control audio crosspoints.

The unit has 8 x Analogue inputs/outputs and 16 x Dante in/outs. 1 or several inputs can be routed to each output.

The Switcher is controlled with the command CTP= and Salvos can be triggered with CTP:

(see above for details on CTP= and CTP: command formats).

Note – the control protocol isn't great for these devices. Only a single device can connect to them, and it doesn't handle multiple commands in quick succession well.

For example, a salvo of

```
CTP=A1,D1,ONS;CTP=A1,D3,OFFS; CTP=A1,D5,OFFS; CTP=A1,D7,OFFS
```

… is likely to either miss switching some crosspoints, or miss reporting the changes back to us.

Using the + option to send these as a single command, seems to be reliable:

```
CTP=A1,D1,ONS+A1,D3,OFFS+ A1,D5,OFFS+ A1,D7,OFFS
```

This is a limitation of the switcher, not VClock (repeated using Telnet).

DHD Consoles

VClock can connect to DHD Consoles using their "DHD External Control Protocol", to receive Logic State Commands.

Logic States can be set up with in the DHD system for things such as a channel being on, to complex logic using AND and OR commands to generate a Logic State for a specific set of circumstances.

To connect to the DHD Console simply set VClock's IPClient to the IP address of the Console on port 2008. The actual command received by VClock is a complex 16 byte command, for example:

```
<0x03><0x00><0x11><0x0e><0x00><0x00><0x01><0xaf><0x01><0x00><0x00><0x00
><0x00><0x02><0xdc><0x18>
```

…but to simplify triggering on this, VClock takes a special Salvo, in the format:

DHD:416,ON   or DHD~UniqueID:416,ON

416 is the specific Logic ID (in decimal) that you are interested in. ON (or 1) will trigger when the state is on, OFF (or 0) will trigger when the state is off.

If no UniqueID is specified VClock will trigger on the command coming from ANY IPClient connection. The UniqueID will lock it to a single IPClient.

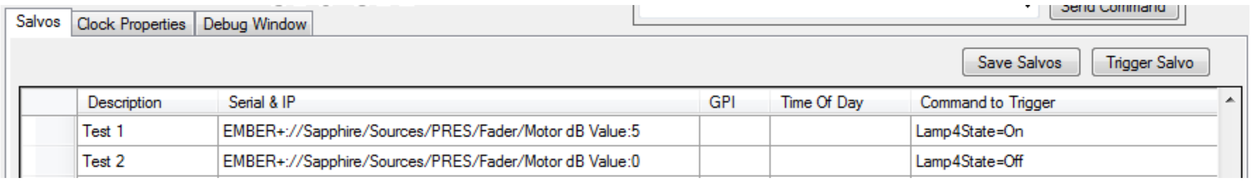
Ember Plus

Ember Plus is a protocol originally designed by Lawo, but is open source and available for anyone to use. It's a bit like XML, in that it is a wrapper which can be used to contain pretty much any data that the provider would like to share. It can be used to read and to write date from/to the Provider.

VClock can connect to an Ember+ Provider, on a specific IP address and Port via the settings on the "IP Settings" Tab in Tools / Settings.

Ember+ is a tree of data. To trigger a Salvo for a specific parameter, you need to know the path to the parameter, the parameter name and the value that you want to check for. You would enter this in the following format:

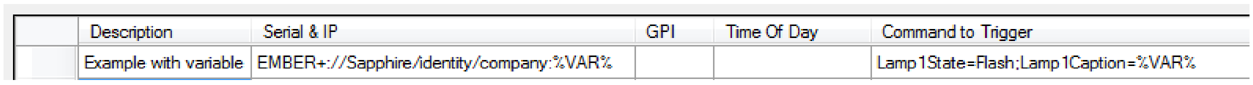
EMBER+://full/path/to/variable/name:<value> (EMBERPLUS: also works).



Description	Serial & IP	GPI	Time Of Day	Command to Trigger
Test 1	EMBER+://Sapphire/Sources/PRES/Fader/Motor dB Value:5			Lamp4State=On
Test 2	EMBER+://Sapphire/Sources/PRES/Fader/Motor dB Value:0			Lamp4State=Off

VClock will add a watch for any variables in the Salvo list when Salvos are saved or when the Ember Plus connection is made, and will then receive info whenver a value changes. If the change matches what VClock is looking for, it will trigger the Salvo.

Some Ember+ fields are text strings which may be useful to display in VClock. To do this, specify %VAR% as the variable, and use %VAR% in the command. The %VAR% in the command will be replaced with the current value of the variable:



Description	Serial & IP	GPI	Time Of Day	Command to Trigger
Example with variable	EMBER+://Sapphire/identity/company:%VAR%			Lamp1State=Flash;Lamp1Caption=%VAR%

Lawo Consoles

Enable the IP Client, pointing at a Lawo "DMS" Interface IP Port. This works best from the frame associated with a studio. The DMS Interface is on port 18510.

VisTool (the Lawo screen software) has on-screen lamps, which are configurable within Lawo. When the state of these lamps change, the DMS Interface broadcasts details. In fact it is the frame that tells VisTool to display the lamp, via the same port that we are connecting to. The actual protocol is quite complex and is not a string of ascii characters, but VClock can still trigger on these unique commands.

An example stream coming from VisTools (using Telnet) is:

This is just a string of ascii characters though, and using the **Debug Window** tab on the config screen, you can easily see the commands coming in.

The actual command you need for "mic live", etc will vary depending on the "slot" and "lamp" number in Lawo. To make it easier, you can set the salvo name to the following:

    LAWO:1,18,RED   or   LAWO~UniqueID:1,18,RED

If no UniqueID is specified VClock will trigger on the command coming from ANY IPClient connection. The UniqueID will lock it to a single IPClient.

(this would trigger when lamp 18 on slot 1 goes red (other states are OFF, GREEN, YELLOW, WHITE). This actually translates to:

    $<0x01><0x00><0x00>y<0xdS><0xLL><0x00><0xCC>

(where S is slot number, LL is lamp number, CC is a code for colour (0=Off, 1=Red, 2=Green, 3=Yellow, 4=White).

Lawo also requires a heartbeat to be sent (a *, once per second. You can tell VClock to send a "heartbeat string" on the settings screen. Lawo also sends one to VClock once per second.

Finally, Lawo can send you the current status if you send it a special command. You can tell VClock to send this command whenever a connection is initiated to Lawo, on the settings screen. Set the Init command to be:

    LAWO:INIT

This actually translates to:

    $<0x00><0x00><0x00>p<0x13><0x00><0x00>

MIDI

VClock can send and receive MIDI commands.

On the Settings page, under the "GPIO and Serial Settings" tab, you need to select the relevant device, and enable MIDI.

MIDI commands are then received and can be used to trigger salvos using the following format:

MIDI:MessageType,Channel,Controller|Note,Value

For **ControlChange**, the Value is the "Value (0-127). OFF equates to 0, ON equates to "not" 0 (!0).

The ! can be used to precede the Value, to say "not"… so will match any value other than the stated one. Useful for example for a fader, so OFF (0) is when the fader is off, and any other value means the fader is "ON".

For **NoteOn** / **NoteOff**, the Note ranges from C0 to G10, where C0 is the numeric value 0 through to G10 being the numeric value 127. VClock can match either the note, or the numeric value.

The notes are always expressed as sharps not flats in Midi, but VClock can use either in Salvos. So

"C", "C#", "D", "D#"/"Eb", "E", "F", "F#", "G", "G#"/"Ab", "A", "A#"/"Bb", "B"

Value is then the Velocity of the Note (0-127).

For example

MIDI:ControlChange,1,BankSelect,1 (button bank 1 selected on a RØDECaster)

MIDI:ControlChange,1,15,OFF (fader 1 off on a RØDECaster)

MIDI:NoteOn,1,A#9,127
MIDI:NoteOn,1,Bb9,127 (Bb is the same as A#)
MIDI:NoteOn,1,A#9,ON (ON is the same as !0)
MIDI:NoteOn,1,118,127 (with the note expressed numerically)

Incoming Midi commands can be seen on the "Debug Window" tab in VClock.

There are simpler commands to use for a RØDECaster - see the RØDECaster section below. The MIDI salvos give raw access to the MIDI commands being received from that and any other compatible device.

MIDI:INIT will be triggered whenever the Midi interface connects / reconnects.

To send Midi, use the command format

```
MIDI=ControlChange,Channel,Controller,Value
```

or

```
MIDI=NoteOn|NoteOff,Channel,Note,Velocity
```

For example, to switch a Rodecaster SMART Pad to bank 1 (zero based):

```
MIDI=ControlChange,1,BankSelect,0
```

Controller may be a known enum such as BankSelect, or can be the numeric value (0-127).

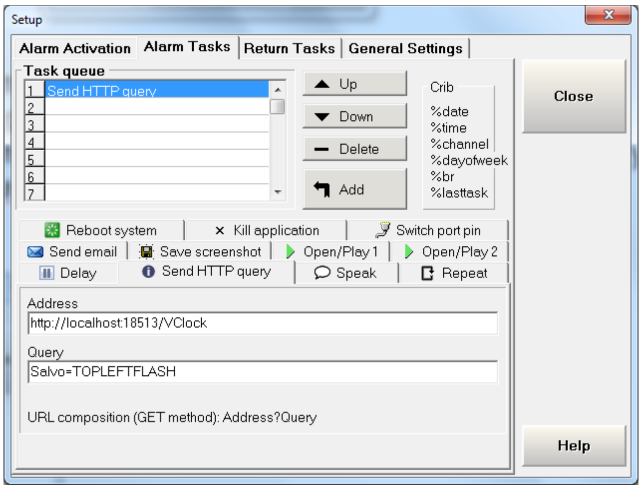
Pira CZ Silence Detector

(tested with v1.3)

http://pira.cz/eng/silence.htm

This is a software based silence detector that can do various actions (such as send an email) when silence is detected. It can also do a http get command, which can be pointed to VClock's HTTP Server interface.

Configuration is simple. Under the "Alarm Tasks" (triggered when silence is detected), choose the "Send HTTP Query" tab and point it to VClock. The address should be the clock's HTTP Server interface (In the format http://localhost:18513/VClock) and the Query should be in the format Salvo=<salvoyouwanttosend>. Remember to press the ADD button to add it to the list:



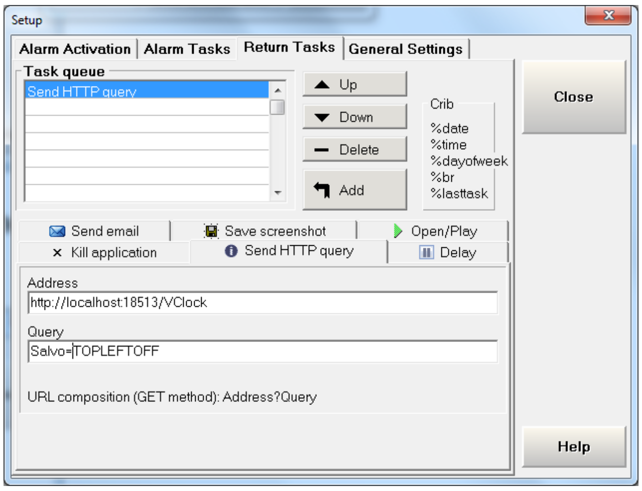
Task queue: 1 Send HTTP query

Address: http://localhost:18513/VClock

Query: Salvo=TOPLEFTFLASH

URL composition (GET method): Address?Query

Then under "Return Tasks" (triggered when audio is restored), do the same but with a different salvo:



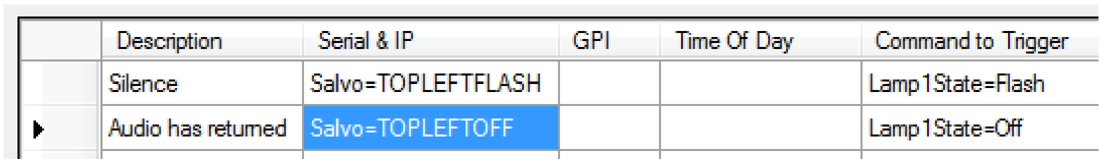
Task queue: Send HTTP query

Address: http://localhost:18513/VClock

Query: Salvo=TOPLEFTOFF

URL composition (GET method): Address?Query

Then in VClock's Salvo list, simply trigger on "Salvo=TOPLEFTOFF":



	Description	Serial & IP	GPI	Time Of Day	Command to Trigger
	Silence	Salvo=TOPLEFTFLASH			Lamp1State=Flash
▶	Audio has returned	Salvo=TOPLEFTOFF			Lamp1State=Off

ProntoNet Codecs

These can send SNMP traps to the clock. Examples of how these are received are:

```
//No Output Level Failure
** SNMPv1 TRAP from 192.168.1.1:1048
*** community public generic id: 6 specific id: 1
*** PDU count: 3
**** Vb oid: 1.3.6.1.4.1.2007.5.666.2.1.1.1.0 type: Integer32 value: 36
**** Vb oid: 1.3.6.1.4.1.2007.5.666.2.1.1.4.0 type: OctetString value: No output level detected in 0 seconds
**** Vb oid: 1.3.6.1.4.1.2007.5.666.2.1.1.2.0 type: OctetString value: Wed Nov 28 14:01:27 2012
\\SNMPv1\192.168.1.1\public\6\1\3.
\\SNMPv1\192.168.1.1\public\6\1\3\1.3.6.1.4.1.2007.5.666.2.1.1.1.0\36.
\\SNMPv1\192.168.1.1\public\6\1\3\1.3.6.1.4.1.2007.5.666.2.1.1.4.0\No output level detected in 0 seconds.
\\SNMPv1\192.168.1.1\public\6\1\3\1.3.6.1.4.1.2007.5.666.2.1.1.2.0\Wed Nov 28 14:01:27 2012.
** End of SNMPv1 TRAP

//No Output Level OK
** SNMPv1 TRAP from 192.168.1.1:1048
*** community public generic id: 6 specific id: 2
*** PDU count: 4
**** Vb oid: 1.3.6.1.4.1.2007.5.666.2.1.1.1.0 type: Integer32 value: 36
**** Vb oid: 1.3.6.1.4.1.2007.5.666.2.1.1.4.0 type: OctetString value: No output level detected in 0 seconds
**** Vb oid: 1.3.6.1.4.1.2007.5.666.2.1.1.2.0 type: OctetString value: Wed Nov 28 14:01:27 2012
**** Vb oid: 1.3.6.1.4.1.2007.5.666.2.1.1.3.0 type: OctetString value: Wed Nov 28 14:01:29 2012
\\SNMPv1\192.168.1.1\public\6\2\4.
\\SNMPv1\192.168.1.1\public\6\2\4\1.3.6.1.4.1.2007.5.666.2.1.1.1.0\36.
\\SNMPv1\192.168.1.1\public\6\2\4\1.3.6.1.4.1.2007.5.666.2.1.1.4.0\No output level detected in 0 seconds.
\\SNMPv1\192.168.1.1\public\6\2\4\1.3.6.1.4.1.2007.5.666.2.1.1.2.0\Wed Nov 28 14:01:27 2012.
\\SNMPv1\192.168.1.1\public\6\2\4\1.3.6.1.4.1.2007.5.666.2.1.1.3.0\Wed Nov 28 14:01:29 2012.
** End of SNMPv1 TRAP
```

So the minimum trigger in VClock, to identify the specific codec, state and which fault it is reporting would be:

No OUTPUT level detected, fail: \\SNMPv1\192.168.1.1\\\1\\\36.
No OUTPUT level detected, ok: \\SNMPv1\192.168.1.1\\\2\\\36.

Similarly…

No INPUT level detected, fail: \\SNMPv1\192.168.1.1\\\1\\\9.
No INPUT level detected, ok: \\SNMPv1\192.168.1.1\\\1\\\39.

ISDN not present, fail: \\SNMPv1\192.168.1.1\\\1\\\15.
ISDN not present, ok: \\SNMPv1\192.168.1.1\\\1\\\15.

PSquared Myriad

Myriad can send “now playing” information to VClock, to be displayed across the top or bottom of the screen. It does this via a “http get” command, which can pass commands to VClock via its HTTP Server.

This uses OCP (v3.6 or above). Steps to set up OCP are:

- In OCP, add a new output
- Select HTTP server
- Paste the line of code you send to force a top caption entry, (http://localhost:18513/VClock?Command=TopCaption=Artist by title)
- Edit this to choose your own VClock HTTP Server IP address and Port, and what you want to display using the OCP editor.
- From the type of message to send - choose GET

RCS Master Control / Zetta

RCS playout systems can send "now playing" information to VClock, to be displayed across the top or bottom of the screen. It does this via a "http get" command, which can pass commands to VClock.

This uses Glue to receive RCS Billboard XML and convert it to a http post/get.

The configuration of Glue is as follows:

```
[Main]
TCPBillboard=yes

[TCPBillboard1]
Port=9001
PlayoutSystem=Zetta
HttpGetNotPost=true
HttpPostLocation=http://localhost:4001/VClock
HttpPostFormat=Command=TopCaption=%SNGART% - %SNGTIT%
```

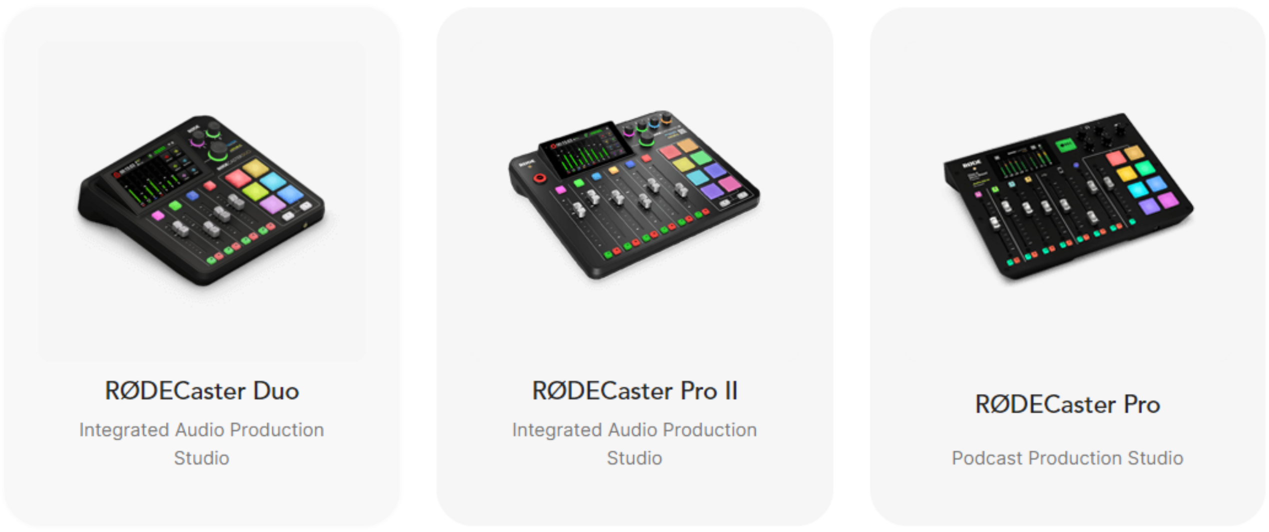
Most of this is probably already set up, and you should contact RCS for further details. The key lines are **HttpPostLocation** and **HttpPostFormat**.

The **HttpPostLocation** is the URL to VClock's HTTP Server. The **HttpPostFormat** is the data that gets sent (in the case above updating TopCaption with the now-playing Artist and Title).

If there already destinations configured, you can add new ones to the end of the line, separating them with the "|" symbol. Be sure to add the same number of destinations to each of **HttpPostFormat** and **HttpPostLocation**.

HttpGetNotPost can also be set in this way if there are multiple destinations (eg false|false|true). But VClock can accept a post or a get on its HTTP Server port.

RØDECaster Duo / RØDECaster Pro / RØDECaster Pro II


RØDECaster Duo
Integrated Audio Production Studio
RØDECaster Pro II
Integrated Audio Production Studio
RØDECaster Pro
Podcast Production Studio

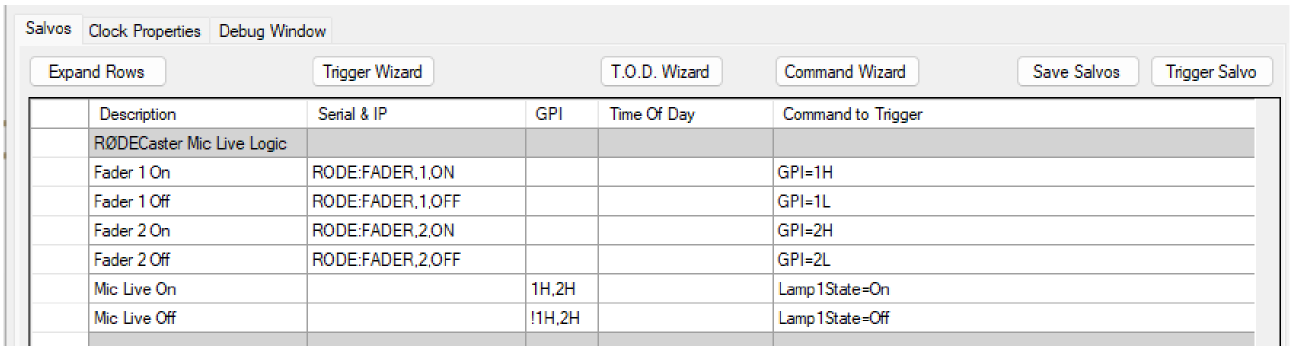
VClock can receive MIDI commands. RØDECaster sends MIDI commands for some of its functions.

VClock has a simplified list of the common MIDI commands required from a RØDECaster:

RODE:FADER,1,ON (triggers when fader 1 is not on its backstop)

RODE:FADER,1,OFF (triggers when fader 1 is on its backstop)

These can be used to set a GPI state, and GPI logic can then be used to build a “Mic Live” signal which could drive a Lamp:



Description	Serial & IP	GPI	Time Of Day	Command to Trigger
RØDECaster Mic Live Logic				
Fader 1 On	RODE:FADER,1,ON			GPI=1H
Fader 1 Off	RODE:FADER,1,OFF			GPI=1L
Fader 2 On	RODE:FADER,2,ON			GPI=2H
Fader 2 Off	RODE:FADER,2,OFF			GPI=2L
Mic Live On		1H,2H		Lamp1State=On
Mic Live Off		!1H,2H		Lamp1State=Off

Buttons on the SmartPad of a RØDECaster Pro II can be configured to send MIDI commands. Assuming you use “Control” and Channel 1, you can use the shortcuts:

RODE:BUTTON,17,ON (triggers when a button assigned ch17 is pressed)

RODE:BUTTON,17,OFF (triggers when a button assigned ch 17 is released)

The older RØDECaster console couldn’t assign specific Midi commands to buttons, but you could load ½ a second of silence onto the buttons and it sends different

Midi commands when they are pressed (the same command for button 1 regardless of which “bank” is loaded though. So the use case is a bit limited, but they can be used in Salvos using the format:

RODE:BUTTON,FX1,ON (triggers when FX button 1 is pressed)

RODE:BUTTON,FX1,OFF (triggers when FX button 1 is released)

The RØDECaster Pro II also sends a Midi command when a bank is loaded:

RODE:BANKSELECT,3 (triggers when the specified bank of buttons is selected)

RØDECaster can also receive MIDI commands to control some of its functions. This is a list of functions that we have found, there doesn’t appear to be a documented list available from RØDE at this time.

RODE=BANKSELECT,1 (select SMART Pads bank X)

The following need a button down and button up (or ON and OFF) event - you are pressing and releasing a button (with LampMouseDown/LampMouseUp for example)

```
RODE=RECORD,ON|OFF|DOWN|UP
RODE=HOTKEY,1,ON|OFF|DOWN|UP
RODE=CHANNELSOLO,1,ON|OFF|DOWN|UP
RODE=CHANNELMUTE,1,ON|OFF|DOWN|UP
RODE=CHANNELSELECT,ON|OFF|DOWN|UP
```

Sadly is isn’t possible currently to get the status of these buttons (ie VClock can’t tell if they are currently on or off, we can only toggle them). So the commands are of limited use.

SAS Systems

VClock can exchange commands with an SAS System.

Simply connect the IP Client to the IP address of the System, the default Port is 1350. Set the "Init String" to be **SAS:INIT.** The Initialization command will tell VClock to check all of the SAS Relay / Opto / XPoint references in the Salvos table whenever the connection is established / re-established, to ensure that VClock knows the initial state.

The commands can be passed through as-is (with SAS=), or there are some special commands for the most used functions / triggers.

See the "Other Commands" section for details on sending commands:

```
SAS=<command>
SASTAKE=<dest>,<source>
SASSALVO=<number>
SASRELAY=<number>,<state>
SASOPTO=<number>,<state>
SASMODULE=<consoleID>,<source>,<state>
SASXPOINT=<destination>
```

It is also possible to trigger on incoming commands:

Exact match

```
SAS:M00050907
```

Modules

SASMODULE:<ConsoleID>,<Source>,<State> (State can be "ON | OFF | CUEON | CUEOFF)

This will be matched when the specified source on a module is set to the specified state.

Optos

SASOPTO:<OptoNumber>,<State> (State can be "ON | OFF")

This will be matched when an SASOPTO "QUERY" is sent as a command, or when the Opto changes.

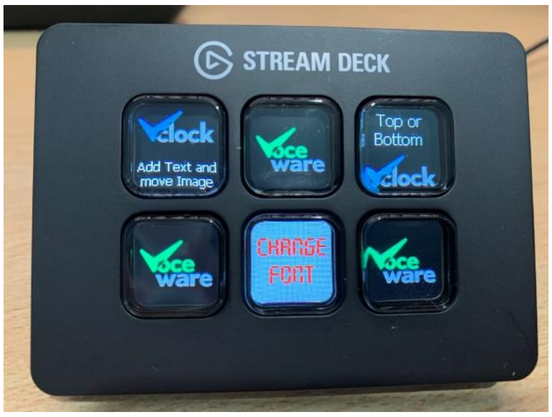
Crosspoints

```
SASXPOINT:<Dest>,<Source>
```

This will be matched when the specified destination changes to the specified source or when queried with the SASXPOINT= command. It is also possible to match when it is NOT set to the specified source using ! preceding the source number. For example:

```
SASXPOINT:1234,5678  (matches when source is 5678
SASXPOINT:1234,!5678 (matches when source is anything but 5678)
```

StreamDeck (by Elgato)



VClock can control StreamDecks in a couple of ways.

Command STREAMDECK= can be used to send a background colour, an image and text to a StreamDeck key, along with some positioning information, font/style/colour etc.

Alternatively (and more powerfully), a StreamDeck key can be linked to a standard VClock Lamp.

Each Lamp has a LampXStreamDeckKey= setting. Set it to the StreamDeck key you want to use (numbered from 1 top-left, then across each row) and the key will display the same image as the on-screen lamp.

You can then disable the on-screen lamp, with LampXStreamDeckMode=StreamDeckOnly.

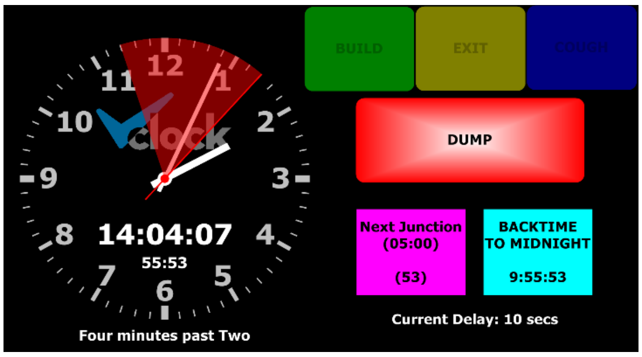
StreamDeck key presses are handled with the Salvo trigger STREAMDECK:

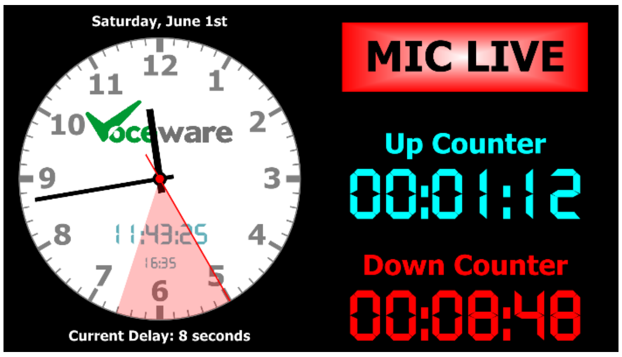
(see above sections for more details on STREAMDECK= and STREAMDECK:)

Telos 25-Seven PDM (Requires a VClock "Plus" license)

VClock can interface with a Telos Programme Delay Manager, and show the time offset by the current delay being added by the PDM. This means the presenter can do accurate time checks and backtiming to hit junctions for third parties receiving the programme post-delay.

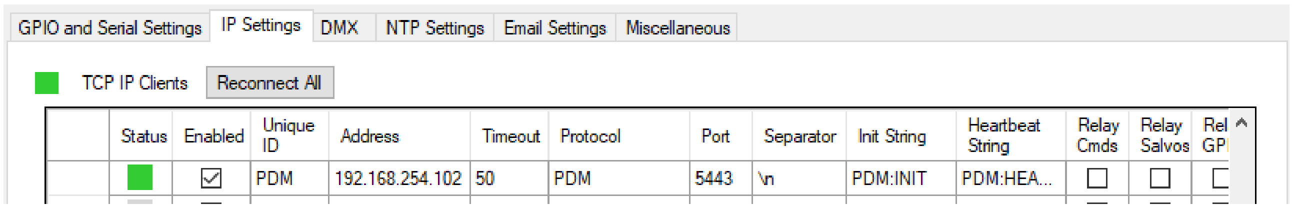
Example interfaces may look like this:





The buttons on the left example can be programmed to control the PDM, and tally its state. The second hand can show the pre or post-delay programme time whilst the trace shows the current delay size. Time can be adjusted to take into account the delay, then the backtimers all take into account that delay.

The connection to the PDM is via an IP Client connection to port 5443. An Init string of "PDM:INIT" will tell it to send us all lamp statuses each time we connect. The heartbeat string of "PDM:HEARTBEAT" will request the current delay once per second:



GPIO and Serial Settings | IP Settings | DMX | NTP Settings | Email Settings | Miscellaneous

🟩 TCP IP Clients   Reconnect All

	Status	Enabled	Unique ID	Address	Timeout	Protocol	Port	Separator	Init String	Heartbeat String	Relay Cmds	Relay Salvos	Rel GP
	🟩	☑	PDM	192.168.254.102	50	PDM	5443	\n	PDM:INIT	PDM:HEA...	☐	☐	☐

It's then down to a few simple Salvos to enable the control/tally lights.

For control, commands can be sent as follows:

PDM=BUILD | EXIT | COUGH | DUMP | BYPASS,DOWN | UP | TRIGGER

Build/Exit/Cough/Dump/Bypass are the front panel buttons. TRIGGER will momentarily press the button, as you would in most situations. Down will hold the button down and Up will release it – useful if you want to have a button to hold down whilst you cough for example. This would generally used with the LampxMouseDown event triggering PDM=COUGH,DOWN and LampxMouseUp event triggering PDM=COUGH,UP.

For tally, there are 2 options. You can emulate the front panel lamps:

PDMLAMP:BUILD|EXIT|COUGH|DUMP|BYPASS,ON|OFF

So PDMLAMP:BUILD,ON would be assigned to LampxState=On and PDMLAMP:BUILD:OFF would be assigned to LampxState=Off

… or you can get status updates when certain triggers are received from the PDM:

PDMSTATUS:BYPASS|BUILDING|DELAYSAFE|DELAYFULL|MUTED|EXITING|
DELAYEMPTY,TRUE|FALSE

The PDM passes through these states as a delay builds, past the 4 second mark to be "delaysafe", until it reaches the max value ("delayfull"). As you come out of delay it is "exiting", then "delaysafe" will be set to false as it falls below the 4 second mark, until finally "delayempty". "muted" is triggered if there is no delay but the dump or cough buttons are held in… it can't dump any more content so it broadcasts silence instead.

Under clock settings, ClockRTCTraceMaxSeconds should be set higher than your maximum delay to display the trace and ClockRTCTraceBrightness controls how transparent the trace is. %PDMDELAYn can be used in a caption to show the current delay (n = 0 – 3 and denotes the decimal places to round to, eg %PDMDELAY1 would count in 0.1 0.2 0.3 seconds).

Using a Visual Basic Script

Create a file called post.vbs and paste the following into it:

```
Set ArgObj=Wscript.Arguments
HTTPPost ArgObj.Item(0), ArgObj.Item(1)

Function HTTPPost(sUrl, sRequest)
  set oHTTP = CreateObject("Microsoft.XMLHTTP")
  oHTTP.open "POST", sUrl,false
  oHTTP.setRequestHeader "Content-Type", "application/x-www-form-
urlencoded"
  oHTTP.setRequestHeader "Content-Length", Len(sRequest)
  oHTTP.send sRequest
  HTTPPost = oHTTP.responseText
End Function
```

To send a command to VClock, type the following command at a command prompt (or put it into a batch file):

```
Cscript.exe post.vbs "http://localhost:18513/vclock" "Command=GPI=1H"
```

VMix (Video Mixing Software)

https://www.vmix.com/

(Not to be confused with VMixes in an Axia Console – see LWCPVMIX for these!)

VClock can switch different inputs to the main programme output of VMix and to the Preview window. It can also turn on/off the 4 x overlay layers, selecting which input to overlay with.

VMIX=<inputNumber>,LIVE|PREVIEW|OVERLAY1ON|OVERLAY1OFF|OVERLAY2ON|OVERLAY2OFF|OVERLAY3ON|OVERLAY3OFF|OVERLAY4ON|OVERLAY4OFF|STARTCRIPT|STOPSCRIPT

For example VMIX=1,LIVE will put Input 1 Live. VMIX=7,OVERLAY1ON will select Input 7 for Overlay, and turn it on.

The Input Number is irrelevant for turning an Overlay off, but one still needs to be supplied. 0 or x is perfectly valid for this.

For starting/stopping Scripts, the InputNumber is the Script NAME. So for example VMIX=Segue,STARTSCRIPT will run the script called "Segue" (case sensitive).

Similarly, VClock can receive triggers when an input or overlay is turned on/off

VMIX:<inputNumber>,ON|OFF|PREVIEWON|PREVIEWOFF|OVERLAY1ON|OVERLAY1OFF|OVERLAY2ON|OVERLAY2OFF|OVERLAY3ON|OVERLAY3OFF|OVERLAY4ON|OVERLAY4OFF

For example VMIX:1,ON will trigger whenever Input 1 is sent to the main programme output. VMIX:3,OVERLAY1ON will trigger whenever Input 3 is set to be used for Overlay 1, and it is enabled.

Wget

You can download a utility called WGet from

http://sourceforge.net/projects/gnuwin32/files/wget/1.11.4-1/wget-1.11.4-1-setup.exe/download?use_mirror=switch

(It is installed in C:\Program Files\GnuWin32\bin… you may want to copy wget.exe from there to the folder you are running your batch file from).

Then simply run

Wget.exe http://localhost:18513/VClock?Command=GPI=1H

Wheatstone

VClock can connect to a Wheatstone Blade and read SLIO output values, to be used as a trigger in Salvos.

Simply connect the IP Client to the IP address of the Blade, Port 55776, set the "Init String" to be **WHEAT:INIT** and the Heartbeat String to be **<>**. The Initialization command will tell VClock to subscribe to all of the SLIOs that are in a Salvo, and the Heartbeat will be sent to keep the connection to Wheatstone alive.

Within Wheatstone's configs you can map any action from any blade to an SLIO output. So use the power of Wheatstone's configs to get all of the SLIOs that you are interested to come out of a single blade, which VClock can connect to.

The format of the Salvo is WHEAT:1,H (SLIO 1 High) and WHEAT:1,L (SLIO 1 Low).

If no UniqueID is specified VClock will trigger on the command coming from ANY IPClient connection. The UniqueID will lock it to a single IPClient. Use the format WHEAT~UniqueID:1,H

Triggering commands from a command prompt / batch file

The easiest way to send commands to the VClock from a batch file is to enable the http Server. There are then 2 easy ways to send it commands:

XKeys

VClock has long since supported the XKeys XK-12 USB GPI device as a cost effective and reliable way of getting contact closures into VClock. There is also a small USB dongle supporting 3 inputs (the XK-3) and a range of simple buttons and foot switches designed to be connected to either device (or you can wire your own of course).

If you want more flexibility / more permanent wiring, the XKeys USB HD15 Wire interface offers 10 GPIs and 2 GPOs on a HD15 connector, alongside an extra 4 GPIs on the standard 3.5mm stereo jacks.

For ultimate flexibility, the USB GPIO interface is similar to the USB HD15 Wire interface – but the 10 inputs on the HD15 are configurable to each be a standard (acfive Low) input, an active High input, or an Output. So this device can offer up to 14 x GPI and 2 x GPO, or 4 x GPI and 12 x GPO.

XKeys also make a variety of button panels. The most interesting of which (I think) are the XK-24 and the rack-mount XKE-40:

Each button is seen as a GPI in the same way as the XK-12. Each button also has a backlight which VClock can control. In fact most units have 2 x backlight colours ("bank 1" - blue and "bank 2" - red). VClock can turn on/off 1 or both colours independently for each lamp. They can also flash.

The rackmount panel in particular would make an ideal audio selector in a Racks Room controlling an XNode output, or a transmission router / OS router / talkback panel in a studio.

There are many other panels available, take a look at https://www.x-keys-uk.com/ or https://xkeys.com/.

Check out the Staff Mug Shots on the About Us page of the American site too - https://xkeys.com/about/mugshots.html. There really are some pretty bad ones on there 😉

To connect an XKeys device to VClock, simply plug it in to a USB port and go to Tools/Settings and "Enable XKeys". You can set an Offset for the incoming GPI (useful if you have several devices so they can trigger separate commands), and can select a different XKeys device if the machine has more than one (a 2nd instance of VClock running on the same machine can either share the same XKeys device, or connect to a separate one).

For the USB GPIO device, the selection screen also allows you to set, for each I/O pin, whether it is an active low input, active high input or an output.

Button presses are seen as GPIs into VClock and Salvos can be set up using GPI logic etc, like any other physical or Virtual GPI.

GPO Outputs (on supported devices) are controlled with the GPO command. Lamps on a button panel can be controlled with the commands below. A Plus licence is required for this (output) functionality.

A salvo can be set up using XKEYS:INIT to trigger whenever the device is reconnected, or at VClock startup. It can be used to set the default lamps, lamp intensity and lamp flash rate.

Valid commands are:

XKEYSLAMPFLASHRATE=<rate>

- rate = 1 (fast) to 255 (approx. every 4 seconds)

XKEYSLAMPINTENSITY=<bank1Intensity>[,<bank2Intensity>]

- Intensity = 0 (off) to 255 (brightest). If only 1 intensity is set, it is applied to both banks

XKEYSLAMPSTATE=<lampNo>,<state>

- Lamps are in blocks of 8 and sometimes lamps are missing. For example the XK-24 uses 1-6 for the first column, 9-14 for the 2nd, etc.
- For single colour devices, valid states are ON, OFF, FLASH.
- For 2 colour devices (blue and red), valid states are:
  BLUE (turns on blue / off red)
  RED (turns on red / off blue)
  BLUEFLASH (flashes blue / off red)
  REDFLASH (flashes red / off blue)
  BLUEFLASHREDON (flashes blue, with red solid on)
  REDFLASHBLUEON (flashed red, with blue solid on)
  BOTH (turns blue and red on)
  BOTHFLASH (flashes blue and red)
  OFF (blue and red both off)
  (ON and FLASH will control the BLUE lamp on devices with 2 colours.)

XKEYLAMPSTATES=[range],<state>

- A short form way of triggering multiple lamps. Range needs to be in [ ]
- Range can be with a dash (1-3) or a list separated with commas (9,11,13). Several ranges can be used in 1 command.
- State as above can be ON, OFF or FLASH
- So for example XKEYLAMPSTATES=[1-3,5-6,9,11,13-14]=BLUE