VISUAL PRODUCTIONS

QUADCORE MANUAL

 WWW.VISUALPRODUCTIONS.NL

Revision History

Revision	Date	Author(s)	Description
8	09.08.2023	JL	Updated MTC informations.
7	05.07.2023	ME	FCC Declaration.
6	12.02.2021	ME	Improved images.
5	04.02.2021	ME	Added examples for Variables, Timers and User Lists. Expanded the topic of Block-features and Time Server.
4	07.06.2018	ME	Added: Rackmount accessory & password protection feature. Updated vManager chapter to reflect app-store distribution. Moved majority of Kiosc information to a dedicated Kiosc manual. Added timespan settings and API feedback.
1	15.09.2017	ME	Initial version.

Contents

VISUAL PRODUCTIONS

Declaration of Conformity

CE

We, manufacturer Visual Productions BV, herby declare under sole responsibility, that the following device:

QuadCore

Conforms to the following EC Directives, including all amendments:
EMC Directive 2014/30/EU

And the following harmonized standards have been applied:
NEN-EN-IEC 61000-6-1:2019

The object of the declaration is in conformity with the relevant Union harmonisation Legislation.

Full name and identification of the person responsible for product quality and accordance with standards on behalf of the manufacturer

Date:
November 24th, 2022

Place:
Haarlem, The Netherlands

ing. Maarten Engels
Managing Director
Visual Productions BV

VISUAL PRODUCTIONS BV
IZAAK ENSCHEDEWEG 38A
NL-2031CR HAARLEM
THE NETHERLANDS
TEL +31 (0)23 551 20 30
WWW.VISUALPRODUCTIONS.NL
INFO@VISUALPRODUCTIONS.NL
ABN-AMRO BANK 53.22.22.261
BIC ABNANL2A
IBAN NL18ABNA0532222261
VAT NL851328477B01
COC 54497795

SUPPLIER'S DECLARATION OF CONFORMITY (No. 2023/202-02)

This Declaration of Conformity is issued under the sole responsibility of the manufacturer

MANUFACTURER

Company name	Visual Productions BV
Full address	Izaak Enschedeweg 38A 2031 CR Haarlem
Country	The Netherlands

RESPONSIBLE PARTY – U.S. CONTACT INFORMATION

Company name	ACT Entertainment
Full address	3581 Larch Lane Jackson, MO 63755
Country	United States of America
Contact details (Phone)	+ 1 800 255 9822

DESCRIPTION AND IDENTIFICATION OF THE EQUIPMENT

Generic denomination	Lighting control system
Function/intended use	Architectural lighting controller
Models	CueCore2, QuadCore, IoCore2, TimeCore, LPU-1, LPU-2

The object of the Declaration described above is in conformity with all relevant provisions of:

FCC (47 CFR Part 15B)

Supplementary information

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation.

Signed for and on behalf of:

Place of issue: Haarlem, The Netherlands
Identity: Maarten Engels
Function: Managing Director

Date of issue: 13-06-2023
Signature: .....

ЕВРАЗИЙСКИЙ ЭКОНОМИЧЕСКИЙ СОЮЗ

ДЕКЛАРАЦИЯ О СООТВЕТСТВИИ

Заявитель Общество с дополнительной ответственностью "Арт Рамос Студио",
зарегистрировано в Едином государственном регистре юридических лиц и индивидуальных предпринимателей за № 101125337,
место нахождения (адрес юридического лица):
Республика Беларусь, Минская обл., 220116, г. Минск, пр-т газеты "Правда", д. 29, пом. 8,
номер телефона: +375 17 2442299, адрес электронной почты: sales@ars-by.com
в лице заместителя директора Стрельцова Антона Валерьевича (доверенность № 1 от 03.01.2017)

заявляет, что световые контроллеры «Visual Productions» модели CueCore2, QuadCore, IOCore2, TimeCore, LPU-1, LPU-2, B-Station2
изготовитель: Visual Productions BV (Королевство Нидерландов),
место нахождения (адрес юридического лица): Izaak Enschedeweg 38A NL-2031 CR Haarlem (Королевство Нидерландов)
изготавливается по технической документации изготовителя
код ТН ВЭД ЕАЭС: 8537 10 910 9
серийный выпуск
соответствует требованиям технического регламента Таможенного союза ТР ТС 020/2011 «Электромагнитная совместимость технических средств».
Декларация о соответствии принята на основании протокола испытаний № 102-19-1927 от 30.09.2019, выданного испытательной лабораторией открытого акционерного общества "Испытания и сертификация бытовой и промышленной продукции "БЕЛЛИС", аттестат аккредитации №BY/112 1.001.;
Схема декларирования соответствия – 1д.
Дополнительная информация: Условия хранения: температура хранения от + 5°С до + 40°С; влажность 60 % при + 20 °С при отсутствии в воздухе кислотных, щелочных и иных агрессивных примесей с соблюдением правил пожарной безопасности. Срок службы 5 лет. Только для профессионального использования.
Примененные стандарты:
ГОСТ EN 55103-1-2013 Совместимость технических средств электромагнитная. Электромагнитные помехи от профессиональной аудио-, видео-, аудиовизуальной аппаратуры и аппаратуры управления световыми приборами для зрелищных мероприятий. Нормы и методы измерений
ГОСТ 30805.22-2013 (CISPR 22:2006) Совместимость технических средств электромагнитная. Оборудование информационных технологий. Радиопомехи индустриальные. Нормы и методы измерений
ГОСТ 30805.14.1-2013 (CISPR 14-1:2005) Совместимость технических средств электромагнитная. Бытовые приборы, электрические инструменты и аналогичные устройства. Радиопомехи индустриальные. Нормы и методы измерений
ГОСТ 30804.3.2-2013 (IEC 61000-3-2:2009) Совместимость технических средств электромагнитная. Эмиссия гармонических составляющих тока техническими средствами с потребляемым током не более 16 А (в одной фазе). Нормы и методы испытаний
ГОСТ 30804.3.3-2013 (IEC 61000-3-3:2008) Совместимость технических средств электромагнитная. Ограничение изменений напряжения, колебаний напряжения и фликера в низковольтных системах электроснабжения общего назначения. Технические средства с потребляемым током не более 16 А (в одной фазе), подключаемые к электрической сети при несоблюдении определенных условий подключения. Нормы и методы испытаний

Общество с дополнительной ответственностью "Арт Рамос Студио" является уполномоченным лицом Visual Productions BV (Королевство Нидерландов) на основании договора от 01.10.2019 № б/н.

Декларация о соответствии действительна с даты регистрации по 29.10.2024 включительно.

Стрельцов Антон Валерьевич

(подпись) М.П. (Ф. И. О. заявителя)

Регистрационный номер декларации о соответствии:
ЕАЭС № BY/112 11.01. ТР020 000 07206
Дата регистрации декларации о соответствии: 31.10.2019

Chapter 1

Introduction

Thank you for choosing the QuadCore; a lighting controller designed for (semi-)permanent installations. The engineering team at Visual Productions is proud to present to you one of the latest development in solid-state lighting control.

Figure 1.1: QuadCore

At the time of writing this manual the QuadCore's firmware was at version 1.38.

1.1 Design Goals

After successfully developing and marketing the original CueCore1 , the team at Visual Productions set out to design a new generation stand-alone lighting controller. We took our experience in developing solid-state controllers and combined it with the feedback received from CueCore1 users all over the world. We added our lessons learnt from supporting year's worth of projects and installs, all in order to design the best lighting controller for (semi-)permanent installations. During the design process we focused on a set of priorities that we valued the most:

1.1.1 Solid-State

The solid-state aspect was perhaps the most important trait that made the CueCore1 a default choice for many system designers. The QuadCore continues this design without any moving part, without forced cooling, and its data safely stored in flash memory. The resulted reliability outperforms any PC based lighting system.

1.1.2 Multi-zone Replay Unit

One of the principle functions of the QuadCore is to playback DMX shows. We took a lot of consideration into making this feature as powerful as possible with the given hardware platform. The playback mechanism we developed can control six different zones. Each zone will be controlled by a completely independent playback. This playback features many options that provides the freedom to the users to employ very smart programming. These options include intensity, rate, precedence, release-time, repeating, fading and inter-cue conditions.

1.1.3 Networking

Our world is one big connected network and the QuadCore will blend in nicely. This Ethernet-based device is setup via DHCP or static address, hosts a modern web-interface for programming and is of course also powered by PoE.

1.1.4 Protocol Conversion

One of the strongest Unique Selling Points of the products designed at Visual Productions is the number of communication protocols supported. The QuadCore further raises this bar. It contains protocols familiar to the CueCore1 (DMX, Art-Net, UDP, OSC) and introduces a fresh set of new protocols: sACN, KiNet, TCP and NTP. This vast collection of protocols can be used for recording, sending, triggering and converting.

1.1.5 Scalability

One QuadCore can do a lot, multiple QuadCore units can do even more. Through using CueluxPro it is possible to control up to 32 universes by connecting multiple QuadCore units. For stand-alone scenarios we even developed a brand new Master/Slave protocol that, once set up with just a few mouse clicks, allows 25 QuadCore to work together and keep 100 universes synchronised at 40 frames per second.

We hope that you enjoy integrating the QuadCore into your lighting designs.

The QuadCore engineering team.

1.2 Compliance

This device is in compliance with CE, UKCA, EAC and FCC regulations.

1.3 Features

The feature set of the QuadCore includes:

- 4 x DMX-512 optically isolated port (bi-directional)
- Art-Net timecode
- Art-Net, sACN & KiNet
- TCP, UDP & OSC
- Master-slave protocol for synchronising multiple QuadCore units
- Scheduling with Real-Time clock[1], weekdays and sunrise/sunset
- NTP time synchronisation
- Desktop or DIN Rail mounted
- Kensington lock
- Locked power cable protection
- PoE (Power Over Ethernet) Class I
- Bundled with CueluxPro, vManager and Kiosc software

1.4 Comparison

The following table visualises the difference between the CueCore3, CueCore2 , QuadCore and CueCore1. This overview might prove to be helpful to CueCore users considering choosing the model for their new designs.

[1]Please note that there is no battery fitted inside the QuadCore. The Real-Time clock has a backup charge through means of a super-capacitor.

	CueCore3	QuadCore	CueCore2	CueCore1
CPU	4x1.2GHz	180MHz	180MHz	120MHz
Memory	8GB	32MB	32MB	8MB
DMX	4 in/out	4 in/out	2 in/out	2 out + 1 in
RDM	yes	-	-	-
MIDI	in+out	-	in+out	in+thru+out
GPI	4x digital/analog	-	4x digital/analog	4x digital
SMPTE	in	-	in	in
MTC	in	-	in	in
Art-Net	in+out	in+out	in+out	in+out
sACN	in+out	in+out	in+out	-
KiNet	out	out	out	-
TCP	in+out	in	in	-
UDP	in+out	in+out	in+out	in+out
OSC	in+out	in+out	in+out	in+out
POE	class III	class I	class I	class I
DHCP	yes	yes	yes	-
NTP	yes	yes	yes	-
Real-time Clock	yes	yes	yes	yes
CueluxPro	4 universes	4 universes	2 universes	2 universes

1.5 Limitations

The QuadCore is a powerful device with many possibilities, there are however some limitations, as shown in the following table.

Playbacks	6
Cues per Playback	32
Tracks	128
Action lists	8
Actions per list	48
Actions system-wide	64
Tasks per Action	8
Tasks system-wide	128
Variables	10
Timers	4

1.6 What's in the box?

The QuadCore packaging contains the following items (see figure 1.2):

- QuadCore
- Ethernet cable
- Power supply
- 4x international plug
- Info card

Figure 1.2: QuadCore box contents

1.7 Saving data to memory

This manual will describe how to configure the QuadCore and program tracks, playbacks, action, etc. The unit's web-interface is used for editing these kinds of elements. When changes are made, these changes are directly stored in the RAM memory of the QuadCore and the programming will directly influence the behaviour of the unit. RAM memory is, however, volatile and its content will be lost through a power cycle. For this reason the QuadCore will copy any changes in the RAM memory to its onboard flash memory. Flash memory retains its data even when not powered. The QuadCore will load all its data back from the flash memory upon startup.

This memory copy process is conducted automatically by the QuadCore and should not be of any concern of the user. One point of consideration is, however, that after making a change the unit should be given time to perform the copy to flash. As a rule of thumb, do not disconnect the power from the device within 30 seconds from making a programming change.

1.8 Document organisation

This manual discusses setting up and programming the unit. Chapter 2 provides background information on the communication protocols used the QuadCore. Chapters **??** and 5 cover how to set up the unit and configure the network connection.

Chapter 7 and 8 cover recording, storing and playback of lighting content.

Programming the automation, triggering and converting functionality is done in the 9 chapter.

When in a hurry, you could skip all chapters and directly follow the quickstart tutorials in chapter 3.

1.9 Further Help

If, after reading this manual, you have further questions then please consult the online forum at `http://forum.visualproductions.nl` for more technical support.

Chapter 2

Protocols

The QuadCore is fitted with several communication ports and supports various protocols. This chapter describes these protocols and to which extent they are implemented in the QuadCore

2.1 DMX-512

DMX-512 is the standard communication protocol for stage lighting. Its official name is E1.11-2008 USITT DMX512-A. Nowadays the reach of the DMX protocol has extended beyond entertainment lighting and is also used for architectural lighting.

Originally one DMX network contained 512 channels which is called a 'universe'. With the growing size and complexity of lighting systems it is now very common for a system to compose of multiple universes, each conveying 512 channels.

It is advised to use a shielded twisted pair cable for DMX cabling. The cable should be terminated with an 120 Ohm resistor.

DMX-512 is a very successful protocol with, however, a few limitations. The maximum number of attached devices is limited to 32 and they all have to be connected in bus-topology having one cable running via each device. Furthermore, a DMX-512 cable should not be longer than 300 meters.

Figure 2.1: Visual Productions' RdmSplitter

The DIN Rail RdmSplitter from Visual Productions (See figure 2.1) helps tackle those inconvenient limitations. The Splitter takes a DMX signal and sends it out again on its 6 DMX output ports for scaling group topology. Each output port is capable of driving 32 more devices. The Splitter can also function as a signal booster as each port supports another 300 meter long connection.

The QuadCore has four DMX ports and is therefor able control 2,048 channels. Each port can also be configured to become a DMX input allowing external DMX data to be recorded or to use an external DMX source to trigger events within the QuadCore.

2.2 Art-Net

The Art-Net protocol primarily transfers DMX-512 data over Ethernet. The high bandwidth of an Ethernet connection allows Art-Net to transfer up to 256 universes.

The data sent out for Art-Net does put a certain load on the network, therefore it is recommended to disable Art-Net when not in use.

Additional to transmitting DMX-512 data, Art-Net can also be used for transferring timecode information for equipment synchronisation.

Each QuadCore supports sending and receiving of 4 Art-Net universes as well as receiving Art-Net timecode.

2.3 sACN

The streaming Architecture of Control Networks (sACN) protocol uses a method of transporting DMX-512 information over TCP/IP networks. The protocol is specified in the ANSI E1.31-2009 standard.

The sACN protocol supports multi-cast in order to take efficient use of the network's bandwidth.

The QuadCore supports sending and receiving of four sACN universes.

2.4 KiNet

KiNet is a proprietary protocol of Philips Color Kinetics to control their LED fixtures and power supplies. It is a lightweight Ethernet-based protocol that carries DMX-style data. Within the QuadCore it can only be used to output data.

2.5 TCP

The Transmission Control Protocol (TCP) is a core protocol of the Internet Protocol Suite. It is used for its reliable, ordered and error checked delivery of a stream of bytes between applications and hosts over IP networks. It is considered 'reliable' because the protocol itself checks to see if everything that was transmitted was delivered at the receiving end. TCP allows for the re-transmission of lost packets, thereby making sure that all data transmitted is received.

The QuadCore supports receiving TCP messages.

2.6 UDP

User Datagram Protocol (UDP) is a simple protocol for sending messages across the network. It is supported by various media devices like video projectors and Show Controllers. It does not incorporate error checking, therefor it is faster than TCP but less reliable.

There are two ways how to have the QuadCore respond to incoming UDP messages. The API (see page 111) makes typical QuadCore functions available through UDP. Furthermore, custom messages can be programmed in the Show Control page (see page 55). This is also the place where to program outgoing UDP messages.

2.7 OSC

Open Sound Control (OSC) is a protocol for communicating between software and various multi-media type devices. OSC uses the network to send and receive messages, it can contain various information.

There are apps available for creating custom-made user interfaces on iOS (iPod, iPhone, iPad) and Android. These tools allow to program fool-proof user-interfaces for controlling the device. E.g. Kiosc from Visual Productions.

There are two ways how to have the QuadCore respond to incoming OSC messages. Firstly, the API (see page 108) makes typical QuadCore functions available through OSC. Secondly, custom messages can be programmed in the Show Control page (see page 55).

2.8 NTP

Network Time Protocol (NTP) is a networking protocol for clock synchronisation between computer systems over networks.

The real-time clock (RTC) in the QuadCore can have a small deviation and drift over time. By occasionally synchronising (e.g. once per day) to an external time server - using the NTP protocol - the RTC stays accurate.

2.9 DHCP

The Dynamic Host Configuration Protocol (DHCP) is a standardised network protocol used on Internet Protocol (IP) networks for dynamically distributing network configuration parameters, such as IP addresses.

The QuadCore is a DHCP client.

Chapter 3

Quickstart

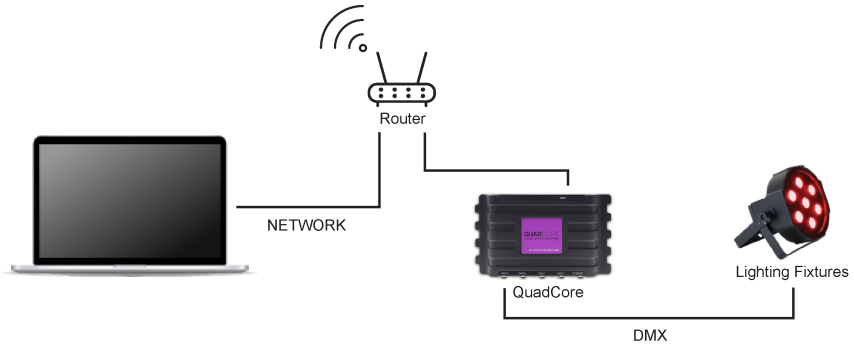
This chapter provides step by step tutorials on how to program your QuadCore for some typical tasks:

- Playback lighting scenes based on the scheduler
- Choose between different lighting scenes via incoming UDP messages
- Record a show from an external DMX console
- Configure as Art-Net Node

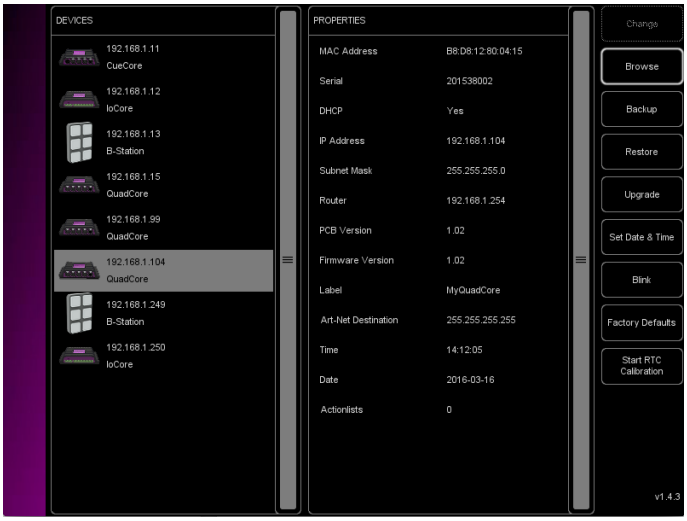
3.1 Playback based on scheduler

This tutorial shows how to create a lighting scene and have it activated at a certain time of the day. The scene will be de-activated at another time. Follow the steps below:

1. Connect to the network
   Connect the QuadCore with an Ethernet cable to the router. It is required that the network is managed by a router that features a DHCP server. If the network router is not DHCP capable then read the network chapter on page 33 for alternative setups.

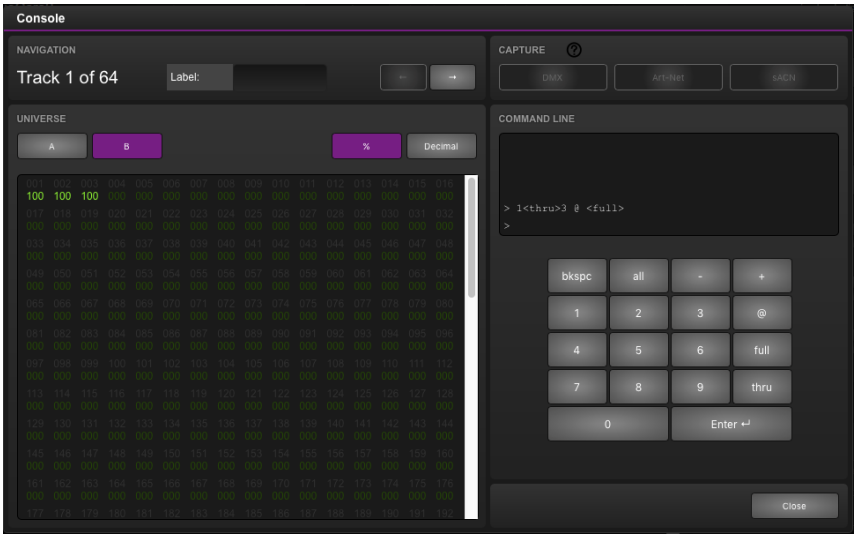


2. Install the vManager
   To access the web-interface of the QuadCore, the vManager tool is required. This tool can be downloaded from the Visual Productions website. Once the installation is complete, run the vManager to discover the IP address of the QuadCore.

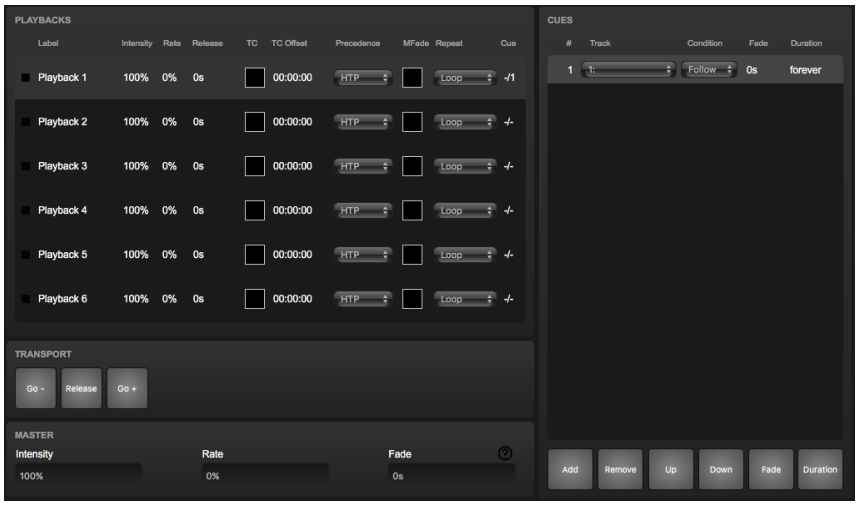


3. Open the web-interface
   Choose the QuadCore from the device list and click on the *Browse* button to open the web-interface.

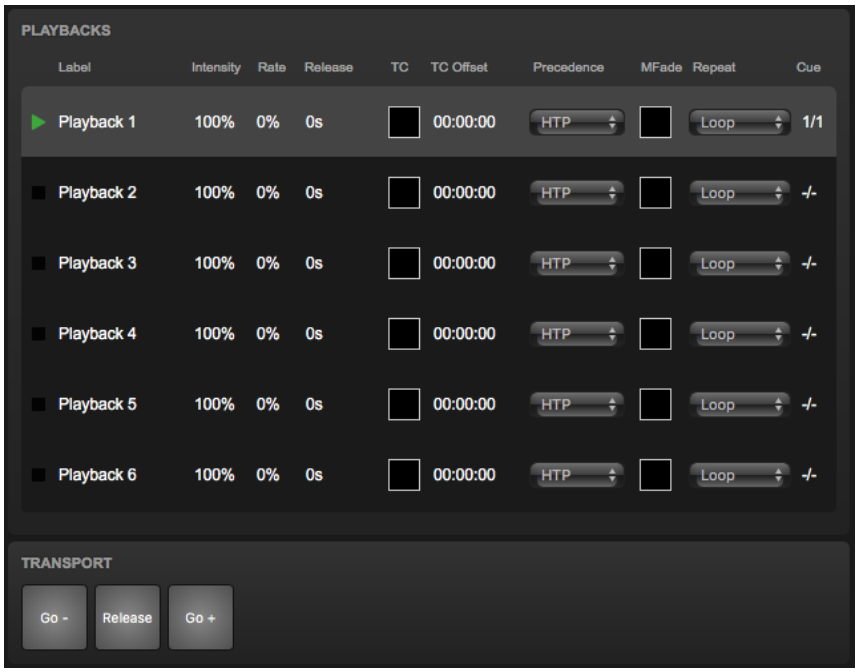
4. Create the scene
   Use the browser to go to the QuadCore's 'Track' page. Select a track from the table and press the 'Open Console' button. Create a scene by using the command-line syntax. E.g. `1<thru>3 @ <full>`



5. Create a cue
   Go to the Playback page and select Playback 1. Press the Add button to create a new cue. Once the Cue is added it will automatically refer to Track 1.



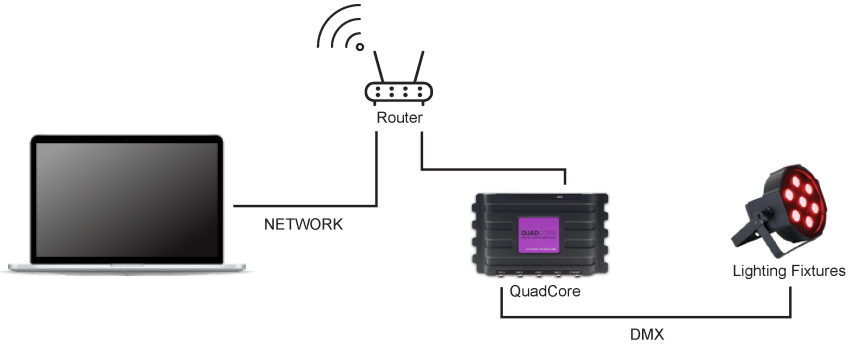
6. Start playback
   Press Go+ on the transport area to start the Playback. The playback now indicated the green 'play' icon.



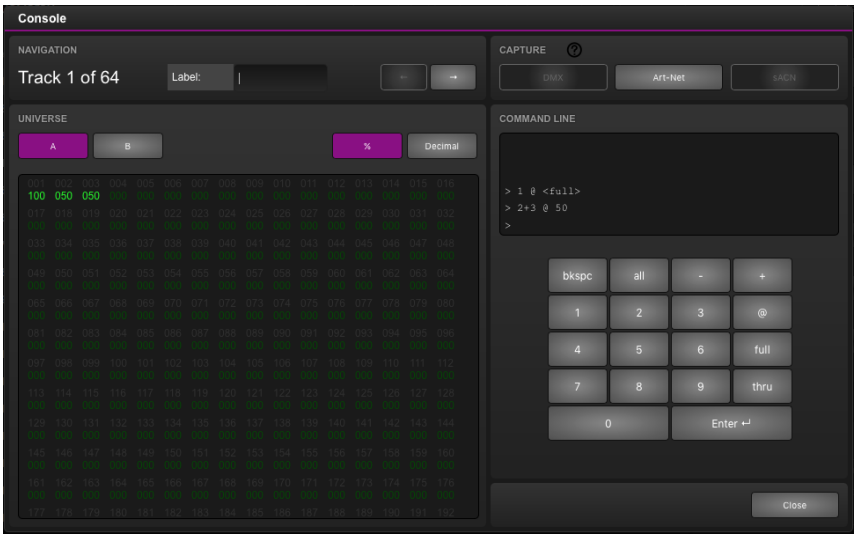
3.2 Choose scenes via UDP

This example will create two lighting scenes. They will be put into a single playback. This means only one scene will be active at a time. Furthermore, a cross-fade will be defined between the scenes and the scenes will be triggered by receiving simple UDP network messages. Please take the following steps:

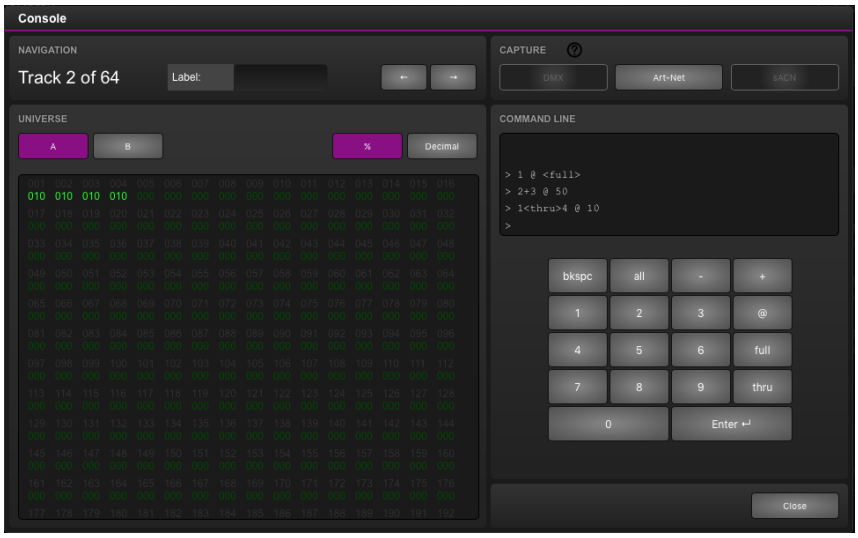
1. Connect to the network
   Connect the QuadCore with an Ethernet cable to the router. It is required that the network is managed by a router that features a DHCP server. If the network router is not DHCP capable then read the network chapter on page 33 for alternative setups.



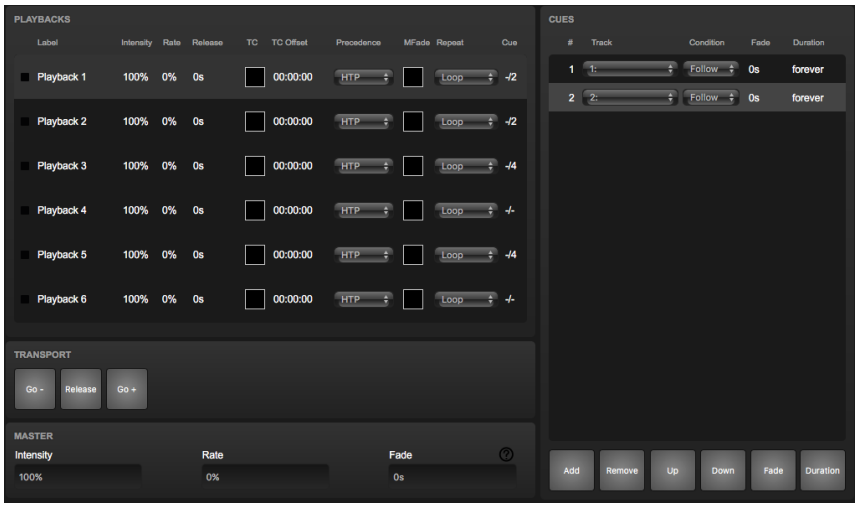
2. Create the first scene
   Use the browser to go to the QuadCore's 'Track' page. Select a track from the table and press the 'Open Console' button. Create a scene by using the command-line syntax. E.g. `1 @ <full>` or `2+3 @ 50 <enter>`



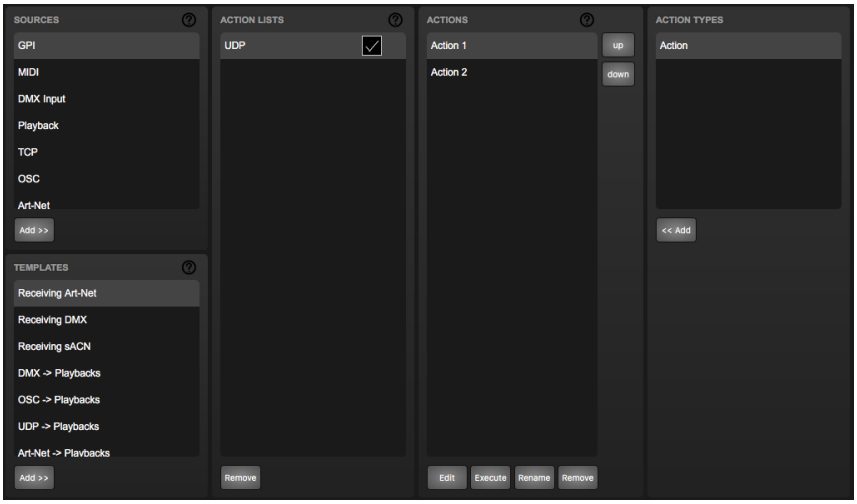
3. Create the second scene
   Press the 'right arrow' button to switch to the next track. Again make a scene by using some command-line syntax; e.g. `1 THRU 4 @ 10 ENTER`



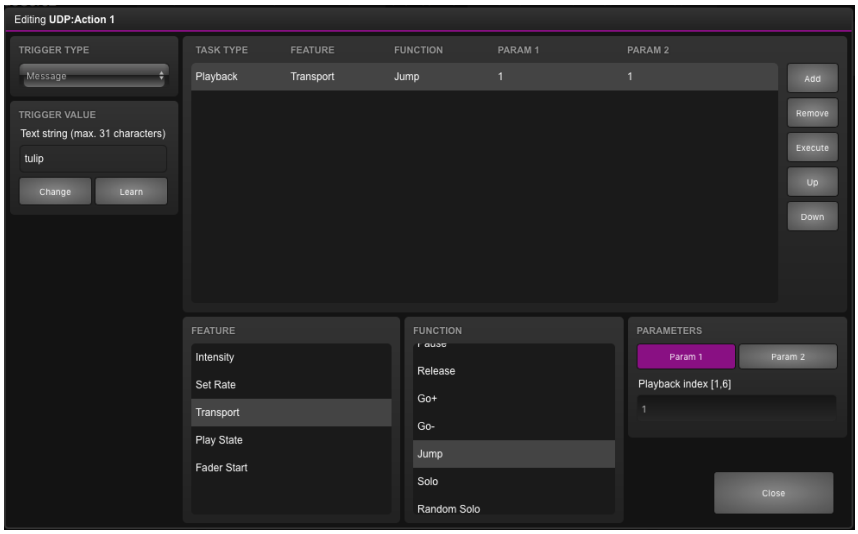
4. Program the playback
   Go to the 'Playback' page, select the first of the six playback and insert two cues by pressing the 'add' button. Set cue #1 to refer to your first track and cue #2 to refer to your second track.



5. Create an action list
   Go to the 'Show control' page. Select 'UDP' from the 'Sources' table. Copy UDP to the 'Action list' table by using the 'Add >>' button. Select the new UDP action list and insert two actions by pressing the '<< Add' button twice.

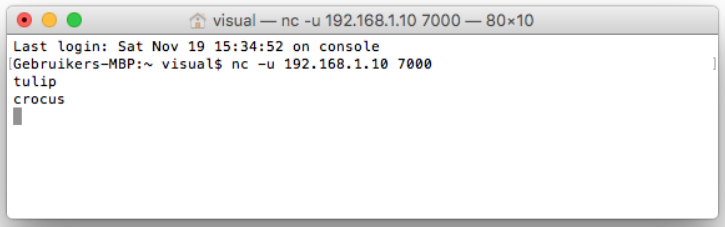


6. Create actions
   Select the first action and press 'Edit' to open the dialog. Change the trigger value to "tulip". Add one task by using the 'Add' button. Choose 'Playback' from the list of task types. Select the newly added task and set the 'feature' to 'Transport' and set the 'function' to 'Jump. Parameter 1 should be set to '1' (addressing the first playback) and parameter 2 should be set to '1' (jump to the first cue).

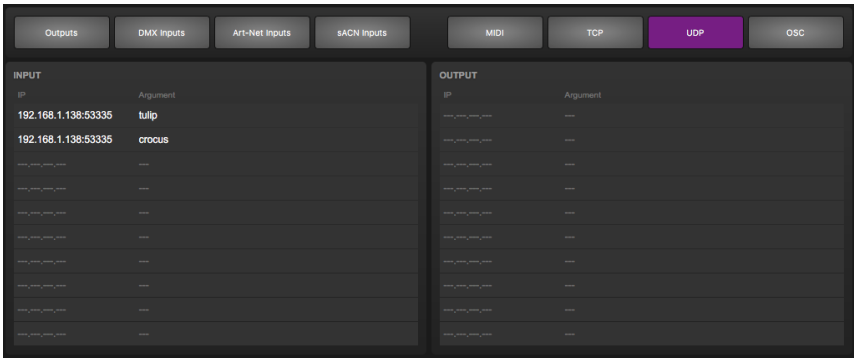


   Press the 'Close' button, select the second action and press 'Edit' again. Change this trigger value to "crocus". Add a task by pressing 'Add' and choose the 'Playback' task-type. Select the newly added task and set the 'feature' to 'Transport' and set the 'function' to 'Jump. Parameter 1 should be set to '1' (addressing the first playback) and parameter 2 should be set to '2' (jump to the second cue).

7. Test with netcat and monitor
   On your computer, use a simple command-line tool like netcat to send a UDP string to the QuadCore. On Mac OSX netcat is started with the command `nc -u 192.168.1.10 7000` (replace 192.168.1.10 with IP address of your QuadCore). From now on you can type `tulip` <enter> or `crocus` <enter> to send this messages to the QuadCore.



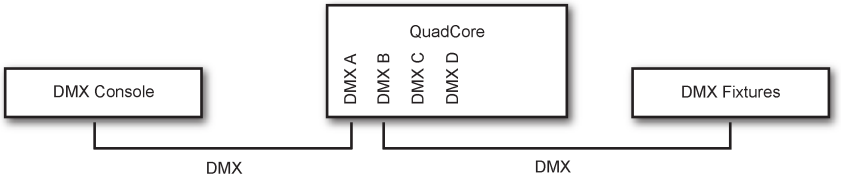
   Go to the 'Monitor' page in your browser and select 'UDP In' to verify your device is receiving the UDP messages correctly. On the 'Playback' page you should see playback #1 respond to the incoming UDP commands by activating either cue #1 or cue #2.



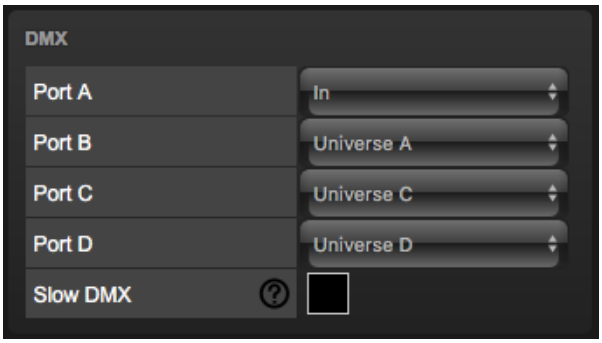
3.3 Record a show from an external DMX

The QuadCore is capable of recording DMX data. This tutorial explains the required procedure.

1. Connect the external console
   Connect the DMX output of the DMX console to *Port A* of the QuadCore. Connect the fixtures to *Port B*.

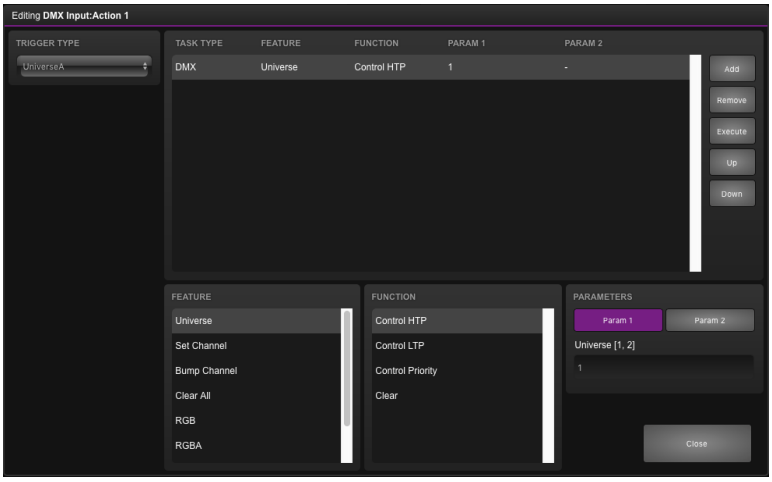


2. Configure port settings
   Go to the *Settings* page and set DMX *Port A* to *In*. Set *Port B* to *Universe A*, it will now transmit DMX channels 1-512. *Port C* and *Port D* are not used in this example.

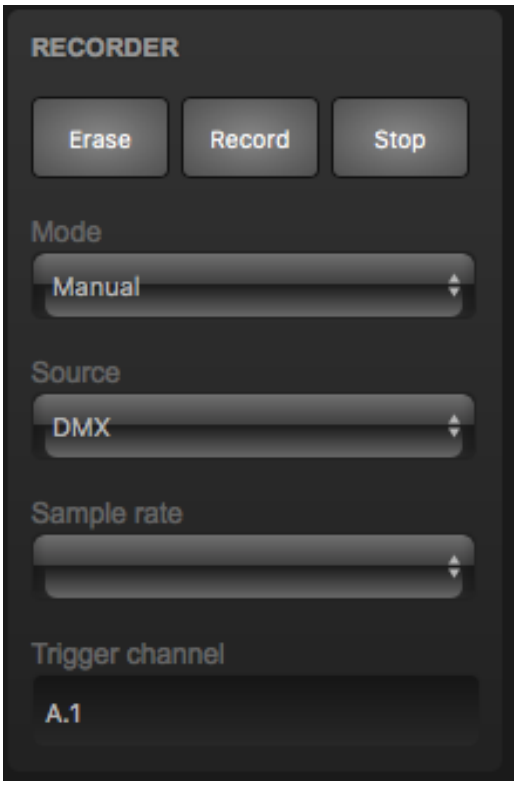


3. Throughput the DMX
   The DMX received by the QuadCore will not automatically be output to the fixtures, however, it is desirable to see the console's output on the actual fixtures. To achieve throughput of the DMX, go to the *Show Control* page. Create a *DMX Input* action list and insert one action.

Edit the action. Set the *Trigger Type* to *UniverseA*. Add a *DMX* task and set its feature to *Universe* and its function to *Control HTP*, the first parameter should be set to 1.



4. Configure the recording
   Go to the *Track* page. Select the first track and press the *Erase* button. Wait until the erase process is completed. Set *Mode* to *Manual*. Set *Source* to *DMX* and set *Sample rate* to 40 FPS.



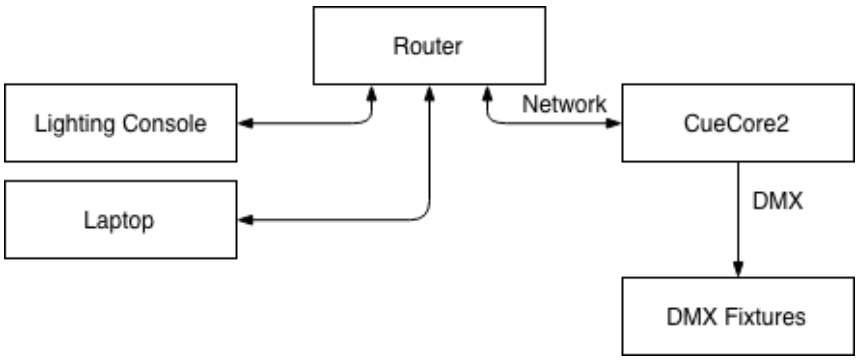
5. Record
   Press the *Record* button at the begin of the console's show. Press the *Stop* button when the show is finished.

6. Test the result
   Make sure the console outputs only zero values. Then playback the track's content by enabling the *Track Preview* checkbox.

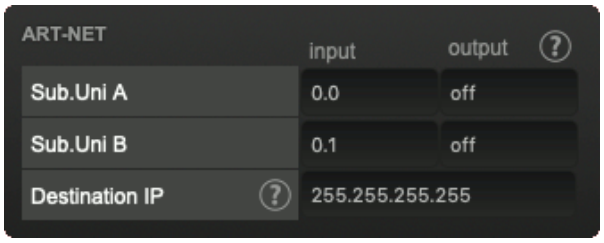
3.4 Configure as Art-Net Node

The QuadCore is capable of sending and receiving various data protocols. This tutorial shows you how to receive Art-Net, and transmit the data through the DMX ports on the QuadCore.

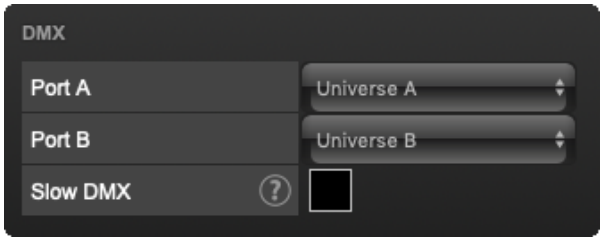
1. Connect to the network
   Connect the QuadCore with an Ethernet cable to the router. It is required that the network is managed by a router that features a DHCP server. If the network router is not DHCP capable then read the network chapter on page 33 for alternative setups.



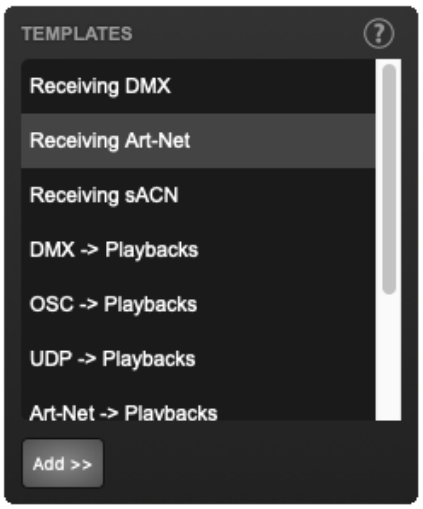
2. Configuring the Art-Net input
   Go to the *Settings* page and set the desired Art-Net universes for port A and B at *Sub.Uni A* and *Sub.Uni B* The QuadCore counts the Art-Net universes from 0.0. For example, Art-Net universe 1 is at 0.0, Art-Net universe 16 is at 0.15 and Art-Net universe 17 is at 1.0. It is also possible to enter the Art-Net universe number (0, 16 or 17 for example) and the QuadCore will automatically convert the value to a valid format.



3. Configure port settings
   Go to the *Settings* page and set DMX *Port A* to *Universe A* and *Port B* to *Universe B*.



4. Forwarding Art-Net to the DMX outputs
   To forward the Art-Net values to the DMX ports, the *Receiving Art-Net* template can be used. Select it, and add it by clicking the *Add >>* button. The Art-Net input is now forwarded to the DMX output.



Chapter 4

Installation

This chapter discusses how to set up the QuadCore.

4.1 DIN Rail Mounting

The device can be DIN Rail mounted. The device is prepared for DIN Rail mounting by using the 'DIN rail holder TSH 35' from Bopla (Product no. 22035000).

Figure 4.1: Bopla DIN rail adapter

This adapter is - amongst others - available from:

- Farnell / Newark (order code 4189991)
- Conrad (order code 539775 - 89)
- Distrelec (order code 300060)

4.2 Rackmount

There is an adapter available for mounting the QuadCore into a 19" rack . The rackmount adapter is 1U and is sold separately. It fits two units, however, it is supplied with one position closed by a blind panel, see figure 4.2.

Figure 4.2: Rackmount adapter

4.3 Kensington Lock

The device can be secured by using a Kensington style laptop lock.

Figure 4.3: Kensington lock

4.4 Power

The QuadCore requires a DC power supply between 9 and 12 Volt with a minimum of 500mA. The 2,1 mm DC connector is center-positive. The QuadCore is also Power-over-Ethernet (PoE) enabled. It requires PoE Class I.

Figure 4.4: DC polarity

Chapter 5

Network

The QuadCore is a network capable device. A network connection between a computer and the unit is required to configure and program the QuadCore, however, once the device is programmed then it is not necessary anymore for the QuadCore to be connected to an Ethernet network.

There are multiple arrangements possible for connecting the computer and the QuadCore. They can be connected peer-to-peer, via a network switch or via Wi-Fi. Figure 5.1 illustrates these different arrangements.

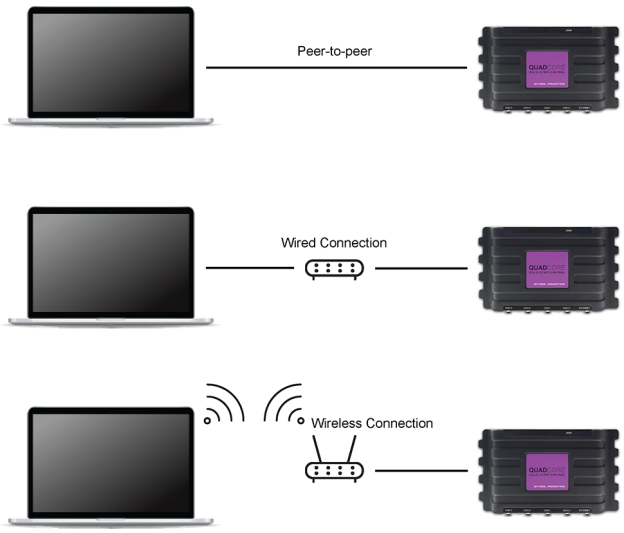


Figure 5.1: Network arrangements

The Ethernet port on the QuadCore is auto-sensing; it does not matter whether a cross or straight network-cable is being used. Although the Ethernet port is classified as 100 Mbps, buffer limits may apply for specific tasks as API messages.

5.1 IP Address

The QuadCore supports both static IP addresses and automatic IP addresses. By default, the QuadCore is set DHCP in which it will be automatically assigned an IP address by the DHCP server in the network. The 'DHCP server' is typically part of the router's functionality.

Static IP addresses are useful when there is no DHCP server in the network, for instance when there is a direct peer-to-peer connection between a QuadCore and a computer. It is also useful in permanent installations where the IP address of the QuadCore is known by other equipment and therefor should not change. When using DHCP there is always the risk of automatically being given a new IP address in the event that the DHCP server is replaced. When using static IP addresses make sure that all equipment on the network have unique IP addresses.

The QuadCore's LED helps to determine which kind of IP address is set. The LED will indicate red when using DHCP and it will indicate white in the case of a static IP address.

There are three ways to change the IP address setting of the QuadCore.

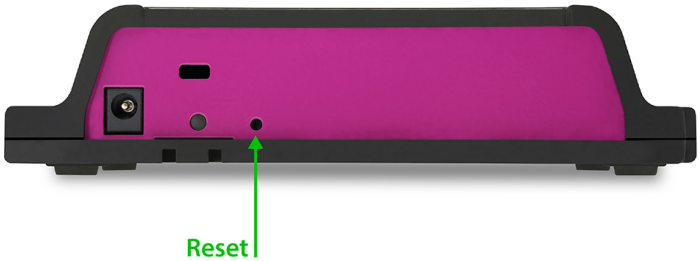


Figure 5.2: Reset button

- **vManager** can be used to detect a QuadCore on the network. Once found, the vManager software (figure chapter 13) allows for changing the IP address, subnet mask and DHCP settings.
- If the IP address is already known then browsing to this address using the computer's browser will show the QuadCore's **web-interface**. The Settings page on this web-interface enables changing the IP address, subnet mask and DHCP settings.
- By briefly pressing the **reset button** on the device it toggles between static and automatic IP addresses. By pressing and holding the reset button (see figure 5.2) on the device for 3 seconds, it will reconfigure the unit to the factory default IP address and subnet mask. No other settings will be changed. The default IP address is 192.168.1.10 with the subnet mask set to 255.255.255.0.

5.2 Web-interface

The QuadCore features an inbuilt web-server. This web-interface can be accessed via a standard browser. It is recommended to use any of the following browsers:

- Microsoft Edge
- Google Chrome (v102 or higher)
- Apple Safari (v15 or higher)
- Mozilla Firefox (v54 or higher)

The web-interface enables you to configure and program the QuadCore. When browsing to the unit the home page (figure 5.3) will appear first. The home page is read-only; it provides information but does not allow for changing any setting. The other pages present many settings that can be edited. These pages will be discussed in the subsequent chapters.

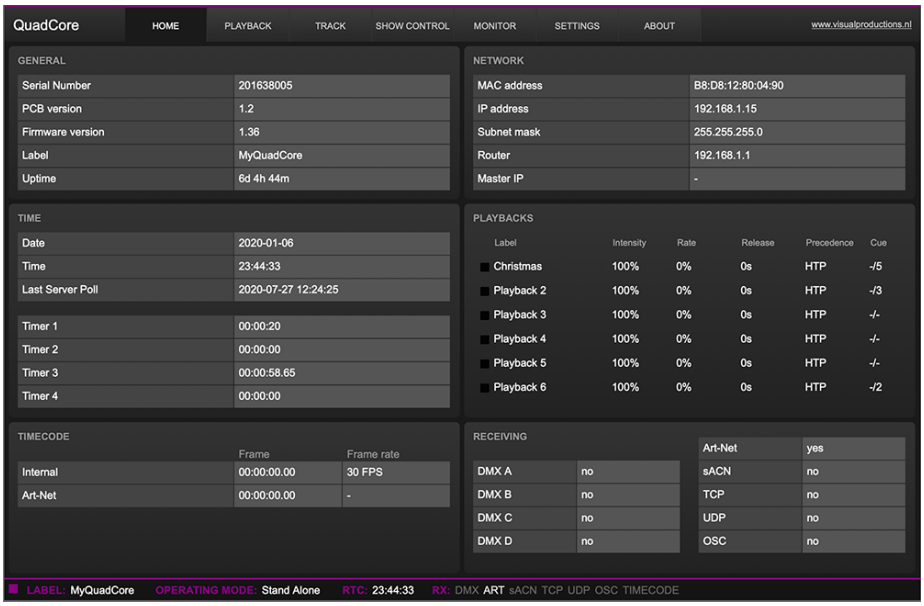


Figure 5.3: Home page

5.2.1 Uptime

This field indicates how long the unit has been alive since its last reboot.

5.2.2 Last Server Poll

Indicates the last time the time & date was fetched from a NTP time server.

5.2.3 Master IP

When the unit is not in Stand Alone mode, then this field displays the IP number of system that is mastering the QuadCore. Refer to chapter 6 for more information on operating modes.

5.3 Access via Internet

The QuadCore can be accessed through the Internet. There are two ways to achieve this: Port Forwarding and VPN.

- **Port Forwarding** Is relatively easy to setup in the router. Each router is different so it is advised to consult the router's documentation (sometimes it is referred to as NAT or Port-Redirecting). Please note that port forwarding is not secure, since anybody could access the QuadCore this way.
- Accessing via a **Virtual Private Network** (VPN) tunnel requires more setup efforts, also the router needs to support the VPN feature. Once set up, this is a very secure way to communicate with the QuadCore. A VPN is a network technology that creates a secure network connection over a public network such as the Internet or a private network owned by a service provider. Large corporations, educational institutions, and government agencies use VPN technology to enable remote users to securely connect to a private network. For further information about VPN please refer to `http://whatismyipaddress.com/vpn`.

Chapter 6

Operating Modes

A QuadCore can operate in three modes, each mode resulting in a different behaviour of the device.

- Stand-alone
- Slave
- CueluxPro

By default the QuadCore operates in the Stand-alone mode.

Figure 6.1: Status bar

The status bar at the bottom of the web-interface (figure 6.1) indicates the current operating mode. When mastered by CueluxPro the home page of the web-interface show the IP address of the CueluxPro system (figure 6.2).

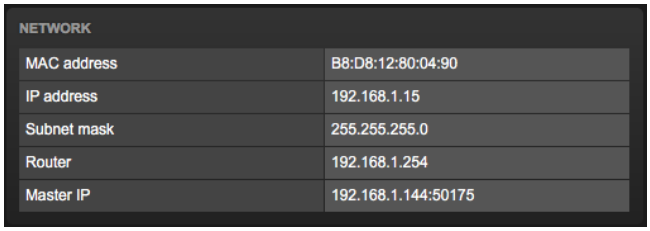


Figure 6.2: Master IP

6.1 Stand-alone mode

In this mode the QuadCore is an autonomous device for controlling lighting. Typically it is loaded with lighting content and programmed to respond to external triggers and/or internal scheduling. This is the default behaviour of a

QuadCore; the stand-alone mode is active whenever the QuadCore is not in the slave or CueluxPro mode.

6.2 Slave Mode

Some demanding lighting designs can require more than four universes of DMX. When multiple QuadCore units are combined to create a large multi-universe system there is the need for synchronisation of those QuadCore devices. The Slave mode facilitates this. See figure 6.3.

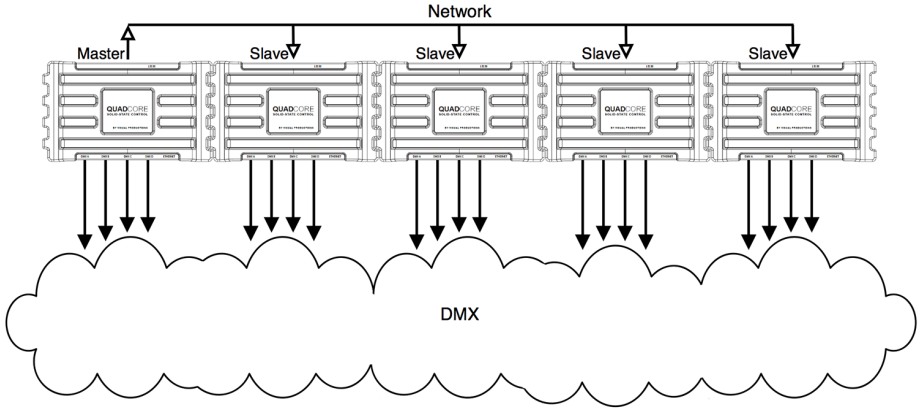


Figure 6.3: Master/Slave setup

When in Slave mode the QuadCore is taken over by a master-QuadCore and is no longer responsible for its playbacks and scheduling; the master takes care of this. All the slave requires is to contain the lighting content in its tracks. The master-QuadCore will control all its slaves to activate the same tracks and keep the playback of those tracks synchronised.

It is necessary to put all action-programming in the master-QuadCore. In fact, the playback information inside the slaves will be overwritten by the master. The master does this because it stores a copy of its playback-data in each slave to enable the slave to continue autonomously in case the communication between master and slave is interrupted.

The logical place for the action lists and action for a master/slave system is also inside the master, however, it is allowed to place actions in a slave and they will get executed.

The Slave mode is enabled in the Settings page (See chapter 12, page 74). Once enabled, the Slave mode is entered as soon as the master connects to the slave. The Slave mode reverts back to the Stand-alone mode when the master disconnects or when the slave disables Master/Slave in the Settings page.

6.3 CueluxPro Mode

CueluxPro (see figure 6.4) is a software-based lighting console that is bundled with the QuadCore. The purpose of the QuadCore in this mode is to be an interface between CueluxPro and the DMX lighting fixtures. Therefore the QuadCore will forward the data received from the CueluxPro software to its DMX outlets. During this mode all internal playback and scheduling within the QuadCore is suspended. Figure 6.5 illustrates a typical CueluxPro/QuadCore system.

Figure 6.4: CueluxPro

The QuadCore enters the CueluxPro mode as soon as it is patched to one or more universes within the CueluxPro software. This mode is exited by unpatching the QuadCore or closing down the CueluxPro software.

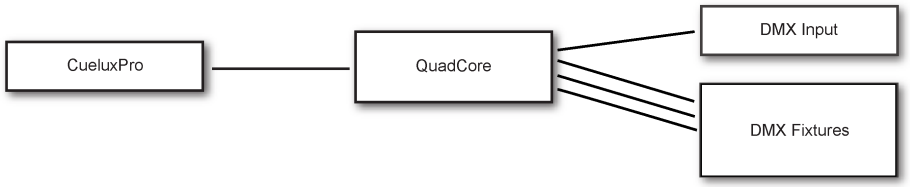


Figure 6.5: A typical CueluxPro system

Using the CueluxPro software in combination with the QuadCore results in a lighting control system with a larger feature set than using the QuadCore on its own in the stand-alone mode. CueluxPro features:

- Personality library with 3000+ fixtures
- FX Generator
- Matrix Pixel-mapping

- Groups
- Palettes
- Timeline editor

CueluxPro can also be used for generating the lighting content that can be uploaded to the QuadCore. After uploading, the QuadCore can continue to be used stand-alone. For information on how to use CueluxPro please refer to the CueluxPro manual on the Visual Productions website. This manual provides instructions for connecting to CueluxPro and uploading content to the QuadCore.

Chapter 7

Tracks

A Track is a piece of lighting content that can be activated by a playbacks. Tracks can contain dynamic lighting effects; each track can be a 'DMX recording' with a certain duration. Of course a static scene can also be stored in a track.

There are three different ways to put the content inside the track. The 'Console' page allows the user to create and edit a static scene directly via the web-interface. This page also is capable of recording a static scene from an external DMX, Art-Net or sACN source. The Console page is discussed in detail on page 42.

The second way for storing content into the tracks is done via the 'Recorder' section; this section of the Tracks page contains control for recording dynamic DMX content from external DMX, Art-Net and sACN sources.

Furthermore, it is also possible to create the lighting content using the CueluxPro software and upload it to the QuadCore. This can be dynamic as well as static content. For more information on CueluxPro see chapter 6, page 37.

7.1 Number of Tracks

The QuadCore has a fixed memory chip onboard. This memory chip is divided into a number of equally sized slots called 'Tracks'. Go to the Settings page to choose the amount of slots the memory chip is divided into. The QuadCore offers a choice of 1, 2, 4, 8, 16, 32, 64 or 128 tracks. More tracks will result in a smaller memory size per track.

Once the number of tracks has been set, the content of the tracks must be erased. It is recommended to choose how many tracks will be used before filling them with content.

Warning: Changing the number of tracks will result in loosing the current content of the tracks.

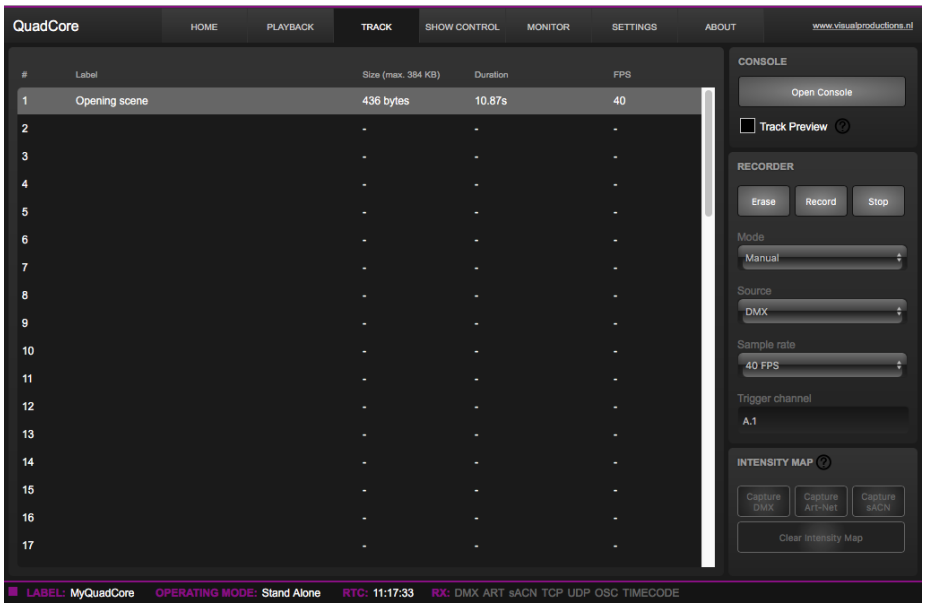


Figure 7.1: Tracks

7.2 Track Properties

The Track listing (See figure 7.1) displays the following track properties:

Label	The name of the track; this field can be changed by double-clicking.
Size	The number of bytes used by the data inside the track. The maximum size is indicated at the top of this column. This maximum depends on the 'number of tracks' selected in the Settings page.
Duration	The length of the track displayed in hours:minutes:seconds.milliseconds.
FPS	The sample rate of the track displayed in Frames Per Second (FPS). The sample rate has been chosen during the recording process and cannot be altered afterwards.

7.3 Console

The Console page (see figure 7.2) allows to edit a track directly through the web-interface, however, a track does need to be a static scene; it should only contain a single DMX frame. If the track already contains more than one DMX frames and thus it is a dynamic track, then it can be made static by erasing it. The track can be edited by selecting the track in the table and then pressing the 'Open Console' button. This will automatically enable the 'Track Preview'

checkbox so the content that is being edited in the Console page is also outputted live.

The 'Track Preview' is a useful option to briefly test the content stored in a track without having to configure a playback for it. Please note that any active playback will be released when the Track Preview is enabled.

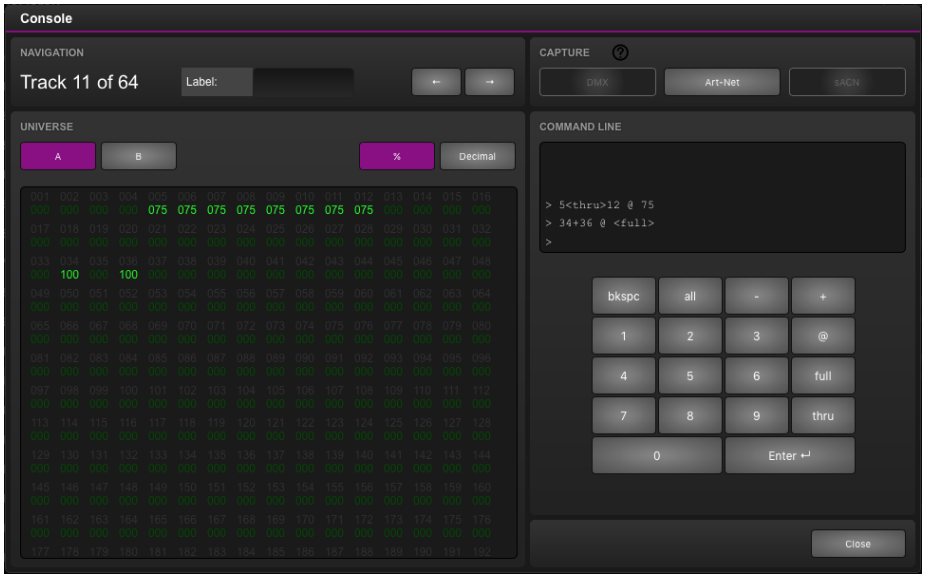


Figure 7.2: Console page

Inside the Console page the DMX values of the track can be changed by using the Command-line interface. The following table offers examples of the supported commands.

Command	Function
`1 @ 50 ENTER`	Sets channel 1 at 50%
`1 + 2 @ FULL`	Sets channel 1 and 2 at 100%
`1 THRU 3 @ 0`	Sets channels 1 through 3 at 0%
`1 THRU 3 + 5 @ 0 ENTER`	Sets channels 1, 2, 3, and 5 at 0%
`ALL @ 100 ENTER`	Sets all channels in the selected universe at 100%
`1 @ + 10 ENTER`	Increases channel 1 value with 10%
`ALL @ - 20 ENTER`	Decreases all channels in the selected universe by 20%

By default the Console page presents the DMX values in percentage (%). When

the representation is switched to decimal (by using the 'Decimal' button) then the values in the table above would be interpreted as decimal values as well. E.g. `1 @ 50 ENTER` would set the channel at decimal value 50 which relates to 20%.

Instead of setting the values manually, the Console page also offers to make a snapshot - record the entire scene - from an external DMX, Art-Net or sACN source. The buttons in the Capture section become available when the QuadCore is receiving the signal of the corresponding protocol. I.e. that the 'DMX' button is disabled unless the unit is receiving actual DMX. Please be aware that - once enabled - pressing one of the capture buttons will overwrite the current channel levels in all universes.

7.4 Recorder

The Recorder section is used to capture dynamic content from an external source and store it inside a track. In order to be stored in flash memory, a track requires to be erased first. It is advised to manually erase the track before starting a record. This is done by selecting it in the table and then pressing the 'Erase' button. In case a non-erased track will be directly recorded then the QuadCore will automatically first erase the track, however, this gives less control over the timing of the start of the recording, especially in the Manual mode.

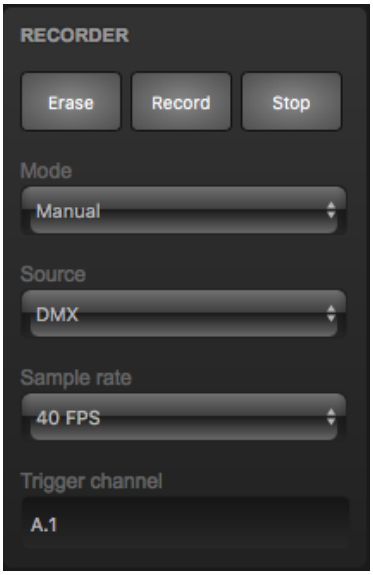


Figure 7.3: Recorder section

The icons in the track table visualise the different states of the recorder. The 'trash icon' indicates a track is being erased. The 'orange dot' signifies a track being ready to start recording, this corresponds to the Triggered or Timecode mode. A 'red dot' indicates a recording in progress.

7.4.1 Mode

The triggering modes define how the recorder is initiated. There are three different modes.

- The most simple mode is **Manual**. In this mode the user has to manually press the 'Record' button to start and press the 'Stop' button to end the recording. Although simple to operate, this mode does not give accurate control over the timing of the begin and end of the recording. Both human interaction and operation through a web-based user-interface will introduce some degree of lag.
- An automated way of starting and stopping the recording process is done in the **Triggered** mode. One of the data channels is allocated to control the start/stop. The channel address is denoted by the 'Trigger Channel' field. It is advised to include this channel in the show programming done on the external source; a typical lighting console allows accurate timing of DMX channels which gives fine control over when the recording starts and ends in relationship to the show content. When using the Triggered mode pressing the 'Record' button will prepare the track for recording; it will be erased when necessary and then stay idle in anticipation of the trigger channel going over 50% to indicate 'start'. The recording is ended by setting the trigger channel back under 50%.
- The **Timecode** mode allows for the recording process to be synchronised by incoming timecode. Pressing the 'Record' button will prepare the track for recording; it will be erased when necessary and then stay idle in anticipation of the timecode to start running, it stops when the timecode stops running. Always record from frame 00:00:00.00. If the content is supposed to run at a different frame then use the playback's 'TC Offset' property to achieve that.

A typical challenge with recording dynamic DMX data is to create a seamless loop. Often the manual mode will most likely not be sufficiently accurate to achieve a seamless loop. The triggered mode offers a way to remote control and make the recording seamless. Alternatively, the lighting content can be designed in CueluxPro instead of recording from an external source, as CueluxPro automatically takes care of making its content seamless.

7.4.2 Sources

The QuadCore is capable of recording DMX data from an external source by using three different protocols:

- DMX
- Art-Net
- sACN

Please consider that the operation of these protocols depend on their properties the Settings page.

7.4.3 Sample Rate

The Sample Rate setting will determine how many samples of the data are taken per second and stored in memory. This setting variants are 5, 10, 30 and 40 Frames Per Second (FPS). 40 FPS gives maximum quality in terms of smooth dimming curves. 5 FPS is a low value but useful for slow DMX changes and consumes much less memory. The 40 FPS setting is recommended unless there is a reason to reduce the sample rate.

7.4.4 XLR Adapter

The DMX ports on the QuadCore are mainly used for outputting DMX and therefor are fitted with female XLR connectors. When using the ports as an input it is likely that the XLR needs to 'gender change' into a male connector. Neutrik offers the NA5MM (figure 7.4), a 5-pin XLR male-to-male adapter for this purpose.

Figure 7.4: Neutrik NA5MM

7.5 Track Capacity

The QuadCore has 32MB memory, of which 24MB is reserved for the tracks. The device uses a compression algorithm to store the data and optimise the storage for best use. The duration of the recording that the track can hold depends on several parameters: number of tracks, dynamic lighting content and the number of DMX channels used. Therefor the maximum duration is hard to specify, however, some guidance can be provided:

In a typical scenario where 32 moving heads - together consuming 512 channels - are constantly changing their primary attributes (position, shutter, colour & gobo) then the memory will hold approx. 16 minutes per track in a 8-track setup. In a 32-track it will hold 3 minutes per track. Both examples are recording at 40 FPS.

In a worst-case scenario with 2,048 channels actively changing to random values (pixel-mapping content) then a 1-track setup will hold approx. 6m32s, a 16-track setup will hold 24s per track. Both examples are recording at 40 FPS.

If the limits of the capacity are reached then there are three different ways to help overcome this.

- Reduce the 'number of tracks' in the settings page. Note that the current track content is lost when changing this setting.
- Reduce the sample rate.
- Spread the content over multiple tracks. They can be linked together later on the Playback page (For more information go to chapter Playbacks, page 20). This way cross-fades can be generated by the QuadCore instead of being recorded.

7.6 Intensity map

Typically, a DMX recorder stores the values of the channels without knowing its functions. When reducing the output level at the Playback all channels are reduced, also the ones that contain information other than intensity/dimmer levels. This has the undesired effect that RGB or Pan Tilt channels are also affected, whereas ideally only the intensity levels should be lowered. This is a challenge all DMX recorders have. The intensity map (figure 7.5) overcomes this issue by specifying to the QuadCore which channels control intensity.

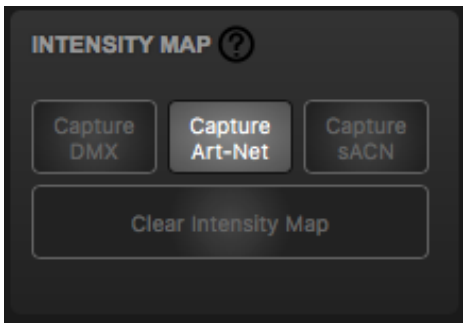


Figure 7.5: Intensity Map section

To set up the Intensity Map follow steps below:

1. Connect an external lighting console via DMX, Art-Net or sACN.
2. Create a scene in which intensity channels are set to 100%. In case of 16-bit dimming, the coarse (or MSB) channels are set to 100% and the Fine (or LSB) channels are set to 50%. All other channels go to 0%.
3. Press the 'Capture' button.

The recording of this lighting scene is now saved in the Intensity Map.

The capture buttons remain disabled while the QuadCore is not receiving the actual signal from the corresponding protocol. The 'Clear Intensity Map' button is only enabled when there is an intensity map present; a disabled 'Clear

Intensity Map' button is an indication that there is no map stored in the memory.

Chapter 8

Playbacks

A playback is capable of activating the lighting content stored in the tracks. Tracks are merely storage for lighting scenes and effects; the playbacks actually plays them. The playbacks are located in the Playback page in the web-interface, see figure 8.1.

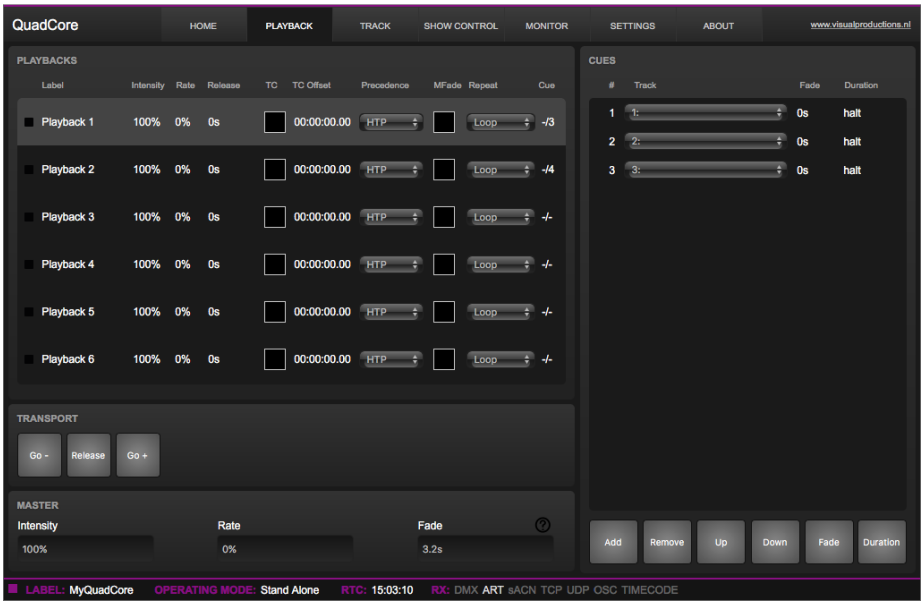


Figure 8.1: Playback page

There are 6 playbacks available. Each can contain up to 32 steps, called 'cues'. A cue will contain a reference to a track plus additional information such as fade-time and duration. Figure 8.2 illustrates the structure of a playback.

Playbacks can be run independently of each other; they can all start or stop at different times. It is possible to control the same DMX channels from multiple

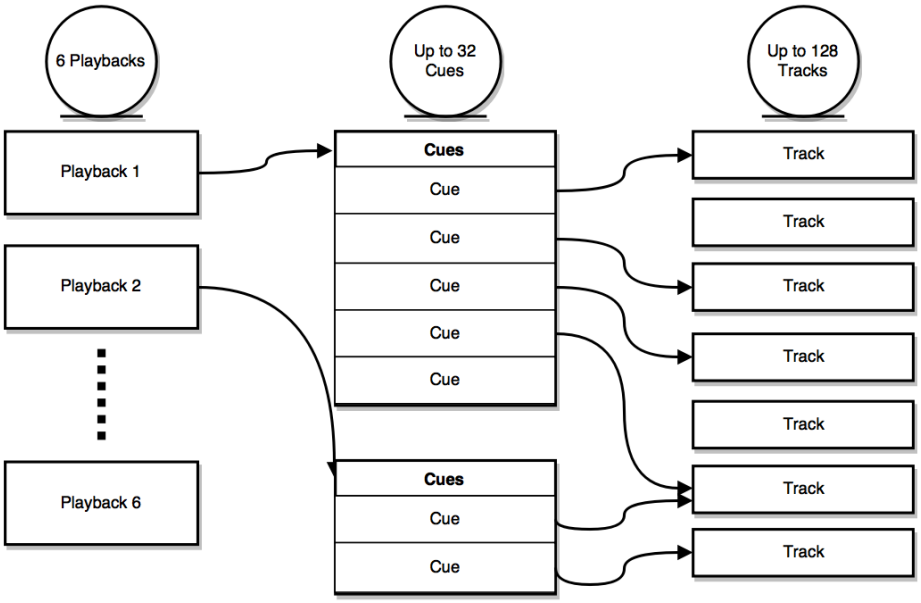


Figure 8.2: Playback structure

playbacks and have them merged together. Also, it is possible to have each playback control a different set of DMX channels; making each playback responsible for a different zone. Figure 8.3 shows an example of controlling multiple zones in a hypothetical restaurant.

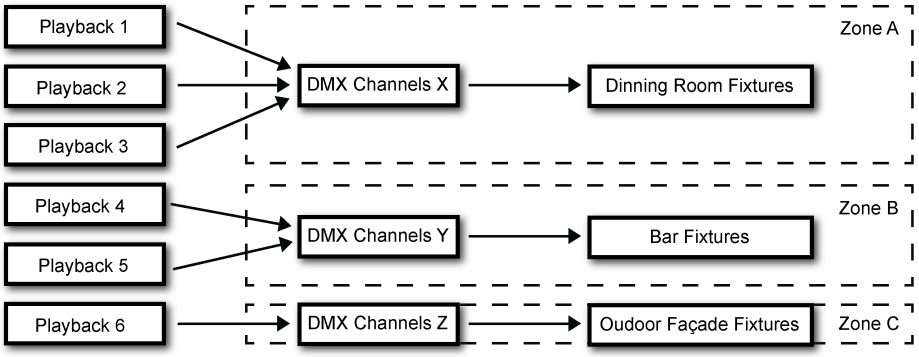


Figure 8.3: Playbacks controlling zones in a restaurant

8.1 Precedence

All active Playbacks produce DMX values. These values will be merged together and sent to the DMX output. The precedence setting determines how this merging is done. Each playback can be set to either HTP (Highest Takes Precedence), LTP (Latest Takes Precedence) or Priority.

HTP is the most common choice in precedence. With HTP the output of all playbacks is compared to each other; for each DMX channel the level is set to the highest value found in that particular channel amongst all playbacks. The table below shows an example of HTP merging.

	Playback 1	Playback 2	Playback 3	Merged Output
Channel 1	0%	0%	25%	25%
Channel 2	100%	0%	25%	100%
Channel 3	0%	0%	0%	0%
Channel 4	0%	100%	25%	100%

In the LTP approach only one playback is active amongst all LTP playbacks. The output of that active playback is included in the merge with all HTP playbacks. All other LTP playbacks are ignored. Which LTP playback is active is determined by which playback is started latest, or which received a Go+ command latest. Please consider figure 8.4.

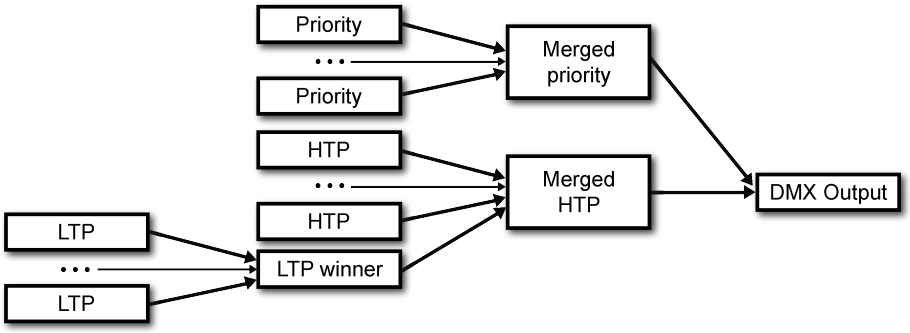


Figure 8.4: Playback precedence

If there is a playback active with its precedence set to Priority then all other playbacks are ignored. When there are multiple Priority playback then those will be merged together according to the HTP principle.

8.2 Playback Properties

Each playback provides a set of properties that can be used to customise the playback's behaviour. Some properties are changed by double-click.

Label	The name of the playback.
Intensity	The output level of the playback.
Rate	The speed of the playback. By default, it is set to 0%. It can go up to 100% (faster) and down to -100% (slower).
Release	When released the playback can fade out to zero. This release time defines how long this fade out will take. Setting it to 0s will result in an instant release.
TC	When enabled, the playback is synchronised to the current timecode (TC). By default, TC is disabled. Note that the Settings page provides a field for selecting the timecode protocol, e.g. 'internal' or 'Art-Net'.
TC Offset	Specifies at which timecode frame the playback starts.
Precedence	Determines how the output of the playbacks is merged together, as explained on page 50.
MFade	Normally the fade time between cues is determined by the 'fade' field in the cue properties. When Mfade is enabled then the playback will ignore the cue's fade times and use the master fade time for all its cues.
Repeat	This property determines what the playback does when it finishes the last cue. Loop: Will start over from the beginning. Bounce: Will make it traverse back to the beginning, and it will keep going back and forth. Random: The order of the cues will be random. Off: The Playback will automatically release when reaching the end of the cues.
Cue	Current/Total of Cues. Indicates which cue is currently active and indicates the total number of cues in the Playback.

The intensity and rate properties are not stored in the QuadCore's internal flash memory. It is expected that these properties can change often during the operation of the QuadCore and could consequently wear out the flash memory.

A consequence of not storing these properties is that after a power cycle their levels will be reverted to the default values. If the intensity or rate requires to be permanently set to a value other than the default value then it is recommended to use the Show Control page and create an action in the 'System' action list. This action can have its trigger set to 'Startup' and contain tasks to set the playback's intensity and rate to the desired values.

8.3 Cue

A cue is a step inside a playback. A playback can contain up to 32 cues. A cue does not contain a lighting scene, rather, it refers to a track which does

contain the lighting scenes. It is possible for multiple cues to refer to the same track. The cue does contain information on how long the lighting scene should be played and if it should be cross-faded from the previous cue.

Figure 8.5: Cues

Each cue provides the following properties:

Track	A reference to the track that will be played in this step.
Fade	The cue will fade from the current levels to its programmed levels. The time it takes to cross-fade is specified by 'Fade'. When the fade is set to 0 then there will be no cross-fade; the values will change instantly.
Duration	Determines how long the cue will be active before traversing to the next cue. This is the time between the completion of the cross-fade into this cue and and the start of the cross-fade to the next cue. The duration field accept not only 'time' input such as ".5" "30s" or "1m15", it also accepts 'number of cycles'; the playback can run the cue "1x" or "10x". This is particularly useful when the track referred to by the cue contains a (seamless-)looped effect. Please note that if the track contains a static scene; i.e. the track only holds a single DMX frame, then running it for a number of cycles will create a very short cue as a single DMX frame only takes 25ms. The third option for the duration field is to input "halt". In this case the cue will continue to run indefinitely; it requires a Go+, Go-, Jump or Release command to traverse to the next cue.

The Playback page provides the following buttons to edit the cues:

- Add: Will add a new empty cue.
- Remove: Will remove the selected cue
- Up: Will move the selected cue up a position.
- Down: Will move a selected cue down a position.
- Fade: Will open a pop-up window where you can set the fade time.
- Duration: Will open a pop-up window where you can set the duration.

8.4 Transport

The transport section offers buttons to control the playbacks.

Go+	Jump to the next cue.
Go-	Jump to the previous cue.
Release	Deactivates the selected playback. Press and hold to release all playbacks.

8.5 Master

The master section provides features that are applied to all playbacks.

Intensity	The master intensity works like a theatrical 'grand master'; it dims the output of all playbacks taking their individual intensity setting into account.
Rate	The master rate will control the play speed of all playbacks; with taking their individual rate settings into account.
Fade	The master fade time overrides the fade time of all cues. This only applies to playbacks that have 'MFade' enabled.

Similar to some of the playback properties, the master properties are not stored in the internal flash memory. Please refer to the discussion on page 52.

Chapter 9

Show Control

The QuadCore can interact with the outside world; it can receive messages and values through various protocols and it can send out many protocols. It is possible to automate the QuadCore by having it respond automatically to incoming signals. An example of this would be to start a playback upon receiving a specific UDP network message.

Another option is integrating the QuadCore with other systems, by using the various protocols it can receive to trigger its functionality. The *Show Control* page (See figure 9.1) enables this kind of programming to be made.

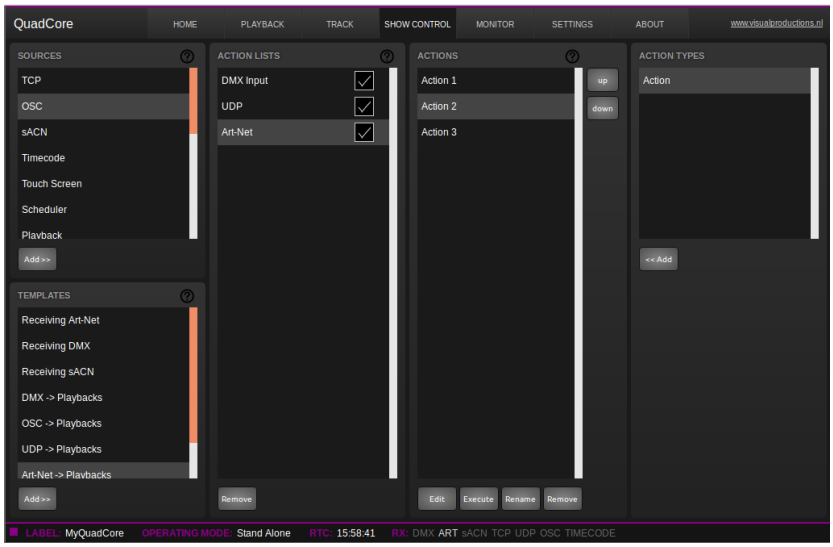


Figure 9.1: Show Control page

The *Show Control* page presents a system of *actions*. A signal that the QuadCore should to respond to or perhaps convert into some other signal, needs to be expressed in an action. Before programming actions please consider the show control structure in figure 9.2.

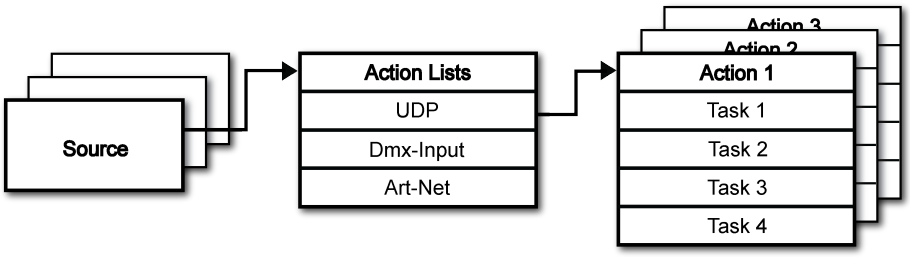


Figure 9.2: Show Control structure

The QuadCore is capable of listening to various protocols. These available protocols are listed in *Sources*, however, the QuadCore can only actively listen to 8 protocols at once. The active protocols are listed in *Actionlists*. Each actionlist can contain actions. Within a protocol/source each individual signal requires its own action. For example, when listening to channel 1 and 2 on the incoming DMX, the DMX actionlist needs two actions; one for each channel.

Inside the action we define the trigger and tasks. The trigger specifies for which signal to filter. In the above DMX example the trigger would be set to 'channel 1' and 'channel 2' respectively. The tasks determine what the QuadCore will do when this action is triggered. Several tasks can be placed in the action. There are tasks available for a wide range of QuadCore features and external protocols. Task types are detailed in Appendix B on page 98.

Please consult the API appendix on page 108 before implementing incoming OSC or UDP messages; the API already exposes typical functionality through OSC and UDP and therefor it might not be necessary to implement custom messages.

9.1 Sources and Actionlists

The Sources listing presents all protocols that the QuadCore is capable of receiving. It also includes internal features that can create events that can be used for triggering actions, such as the calendar-scheduler. These sources are available, however, they will only be actively listened to once moved to the action-list table.

UDP	Incoming UDP network messages
TCP	Incoming TCP network messages
OSC	Incoming OSC network message
DMX Input	DMX received on one or more of the DMX ports (switch port to input in the settings page)
Art-Net	Incoming Art-Net DMX data
sACN	Incoming sACN DMX data
Timecode	Timecode signal, specify the incoming timecode protocol on the Settings page.
Kiosc	Triggers from Kiosc. For each Action various controls can be chosen such as buttons and sliders, colour picker etc. The order of the actions will control the arrangement in Kiosc.
Scheduler	Triggers based on time, date, weekdays, sunrise, sunset & timespan.
Playback	Events generated by the playbacks
Randomiser	The randomiser can generate a random number
System	Events such as 'Start up'
Variable	The Variable source works in combination with the variable task (For more information about the Variable task please refer to Task Types). The Variable task will set a value of which an enabled actionlist type with Variable as Source will use as a trigger. The QuadCore will keep the values of the 10 variables even after shut down so long as the RTC battery is not empty.
Timer	There are 4 internal timers in the QuadCore. An event will be raised when a timer expires. Timers are set and activated by the Timer tasks.
Actionlist	Gives an event when an actionlist is enabled or disabled.
User List 1-4	These actionlists will never trigger an event, however, they are useful for advanced programming.

Actionlists can be temporarily suspended by disabling their checkbox in the Show Control page. There is also a task available to automate changing the state of this checkbox.

9.2 Actions

Actions are executed when a certain signal is received. This signal is defined by the trigger. A trigger is always relative to the actionlist the action belongs to. For example, when the trigger-type is set to 'Channel' then it refers to a single DMX channel if the action is placed inside a 'DMX Input' list and it means a single Art-Net channel if the action resides in an Art-Net actionlist.

Figure 9.3 shows the screen when editing an action.

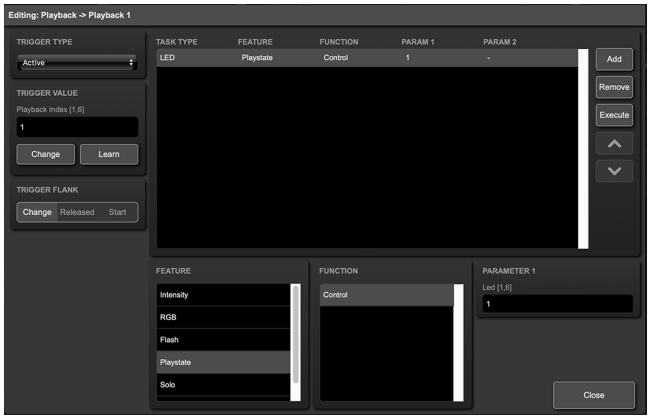


Figure 9.3: Edit Action dialog

A trigger is determined by the trigger-type, trigger-value and trigger-flank fields. Although these fields are not applicable for all actionlists and are therefor sometimes omitted in the web GUI. The trigger-type field specifies what kind of signal the action will be triggered by.

For example, when making an action in the Scheduler list there is the choice between 'DateAndTime' and 'WeekdayAndTime' trigger-types. The trigger-value specifies the actual signal value. In the schedular example the trigger-value could be set to "2016-03-24 11:00" or "Weekend 10:00" respectively.

In some actionlists actions do also need to specify the trigger-flank. The flank further specifies the value that the signal should have before triggering the action. For example, when an action is triggered from the Kiosc list, the flank will determine whether to trigger only when the button goes down or only when it goes up. Appendix A provides an overview of the available trigger-types.

An actionlist can have up to 48 actions, system-wide there is a maximum of 64 actions.

9.3 Tasks

Tasks are added to an action in order to specify what to do when it gets executed. Up to 8 tasks can be included in an action, systemwide there is a maximum of 128 tasks. The tasks are executed in the order of the list. There is a wide selection of tasks available to choose from, they include altering any of the internal software features like playbacks and recorder but also sending out messages through any of the supported protocols. The tasks are organised in categories. Once a task is chosen from these categories each task allows for further choice between several *Features* and *Functions*. Tasks contain up to two parameters that might be required for its execution.

If the event that triggers the action passes a parameter along then this parameter can be used in a task. The *Set* function makes a task use a fixed value, however,

when using the *Control* function then the trigger's parameter is used. This is very useful for conversions between protocols.

For example when converting OSC to DMX the OSC action specifies the URI (e.g. /dmx100) and flank (e.g. OnChange) on which it will trigger. The actual OSC (float) data received in the message will be passed along and fed into the action. Then when a task (e.g. DMX) uses the function *Control* this OSC float-level will be used for setting the DMX value.

A task can be tested by selecting it and pressing the 'execute' button in the *Edit Action* dialog. The complete action can also be tested; go to the *Show Control* page, select the action and press the *Execute* button.When either of these *Execute* buttons are used, the source of the *Control* value for tasks will be the *Execute* button. The result will depend on the chosen task and feature, but will most likely be 100%, 1.0, or 255 when pressed and 0%, 0.0, or 0 when depressed.

Appendix B provides a detailed overview of the available tasks, features, functions and parameters.

9.4 Templates

The *Show Control* page presents a list of templates. A template is a set of actionlists, actions and task. These templates configure the QuadCore to perform typical functions; for example convert Art-Net to DMX or control the 6 playbacks through OSC. The templates thus save time; otherwise actions should have been set up manually. They can also function as a guide to soften the learning curve on actions; a lot can be learned from adding a template and then exploring the actions and tasks it created.

Please note that some templates require settings to be changed in the settings page; for example the 'Receiving Art-Net' template needs the DMX outlets to be set to outputs in order to achieve an Art-Net to DMX conversion.

Appendix C gives an overview of the available templates.

9.5 Variables

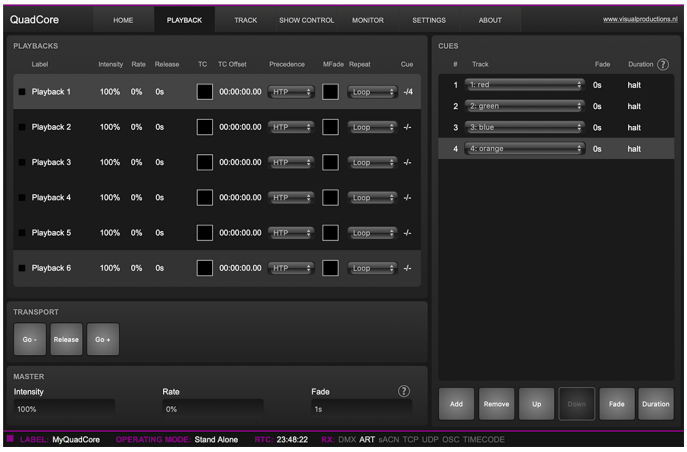
Variables are part of the show control system in the QuadCore. There are 10 variables and each can hold a value in the range of [0,255]. These values can be manipulated by tasks and can be used for advanced action programming. Variables can be added as sources in order to have actions triggered when a variable changes value.

You can see the status of the variables in the monitor page, as discussed on page 70.

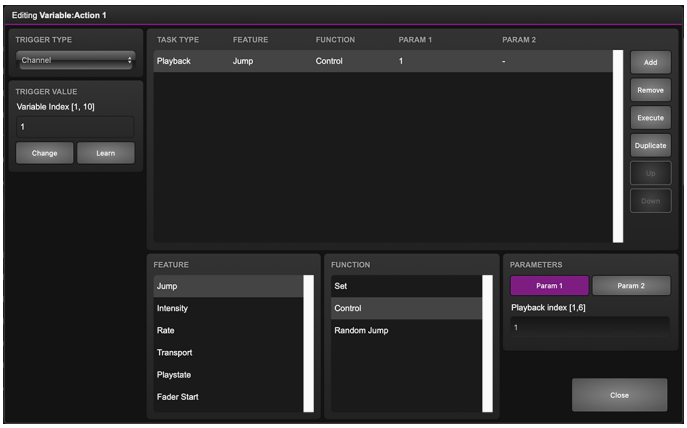
The values of the variables are stored in the same battery backed-up memory as the RTC; it will hold the values between power cycles, if the time between power-on does not exceed a few days.

To illustrate the use of variables, please see the following example in which a variable is used to keep track of the cue index between power-cycles. (By default, the QuadCore does not remember which playback or cue was active after a power-cycle.) For simplicity, we assume that an external system is selecting cues inside the QuadCore by sending UDP messages.

- Program a playback to contain 4 cues.

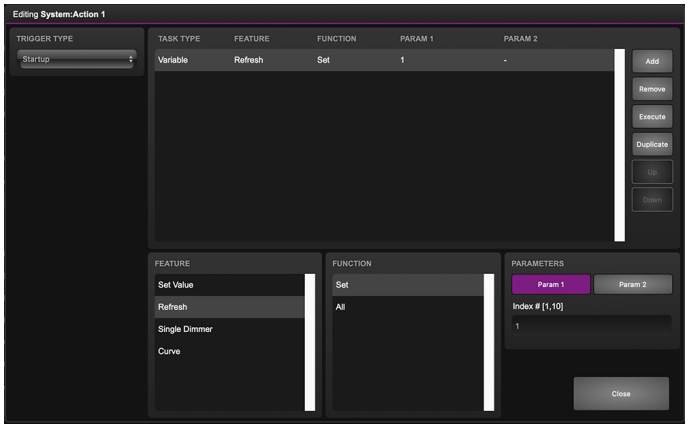


- Add the *Variable* source and insert one action. If the variable changes then jump to the appropriate cue by adding a *Playback* task to this action.



- Have the external system use the UDP API to set the variable values in order to select a cue. The appropriate API message is `core-va-1-set=<integer>` where `integer` is the cue index.

- Add the *System* source and insert one action. At start-up, trigger a variable 'change' in order to jump to the previously selected cue.

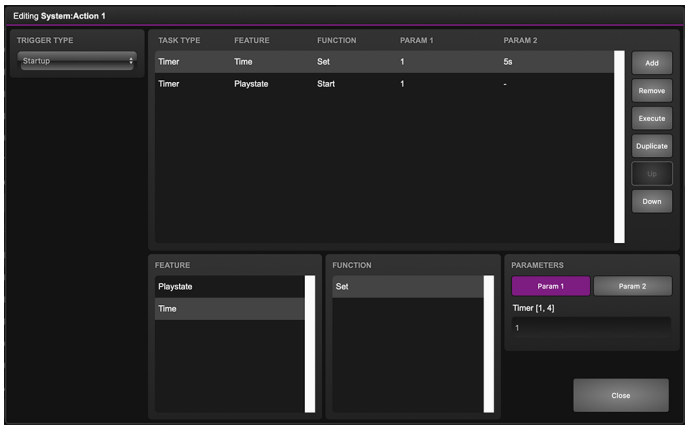


9.6 Timers

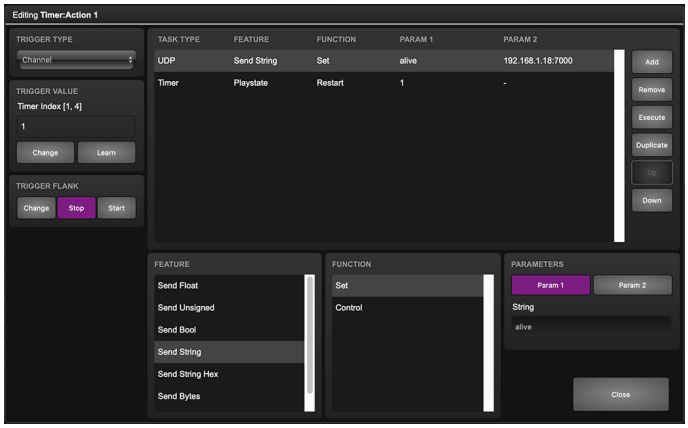
The show control system of the QuadCore features four internal timers. By using tasks, the timers can be set to certain durations and they can be started. Once started the timers will countdown to zero. When the timer reaches zero it will generate an event that can be captured by using the *Timer* actionlist. Please note that the timer values are not stored between power cycles. You can see the status of the four timers in the monitor page, under *Timers*.

The next example will show a timer being used to regularly send a UDP message to an external system in order to notify that the QuadCore is still 'alive'.

- Setup and start the timer at power-on. This is done by creating an action in the *System* actionlist.



- If the timer expires then send the UDP message and restart the timer. This is done by creating an action in the *Timer* actionlist.

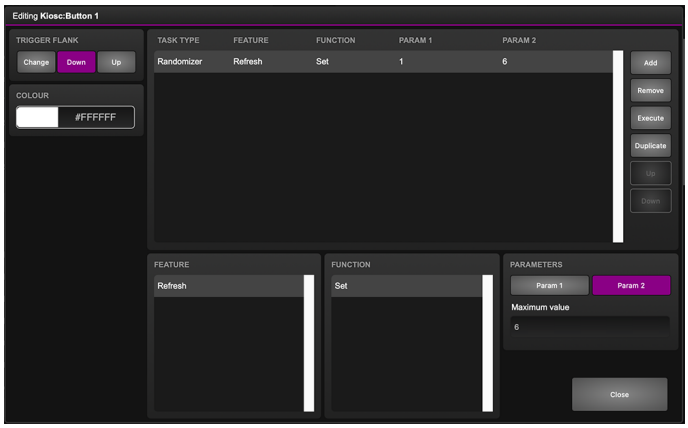


9.7 Randomizer

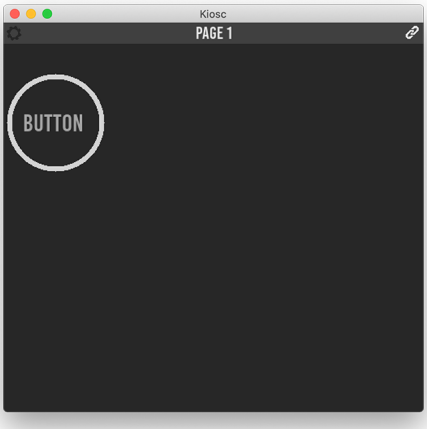
The *Randomizer* is an internal software feature that can generate a (pseudo-)random number. This is useful for having an event trigger a random lighting scene in a themed environment. The randomizer is activated by the Randomizer task. The result of the randomizer's calculation can be obtained by catching the event in the Randomizer actionlist.

The following example shows how to use a Kiosc button to trigger a random cue.

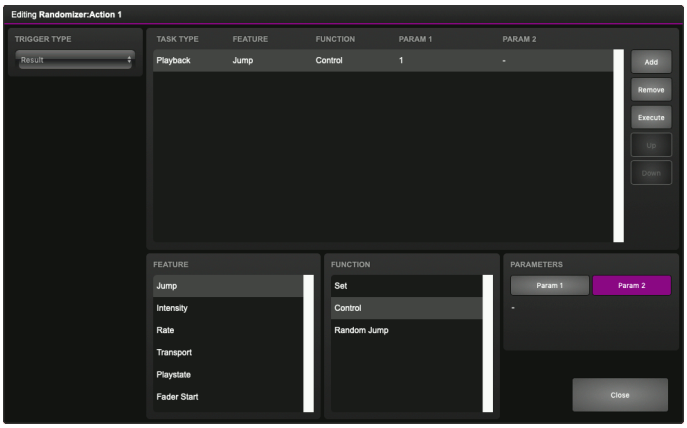
- Inside the Kiosc actionlist is a button-action. This triggers the task Randomizer, which is set to a range of between 1 and 6. (Parameters 1 and 2 of the task Randomizer)



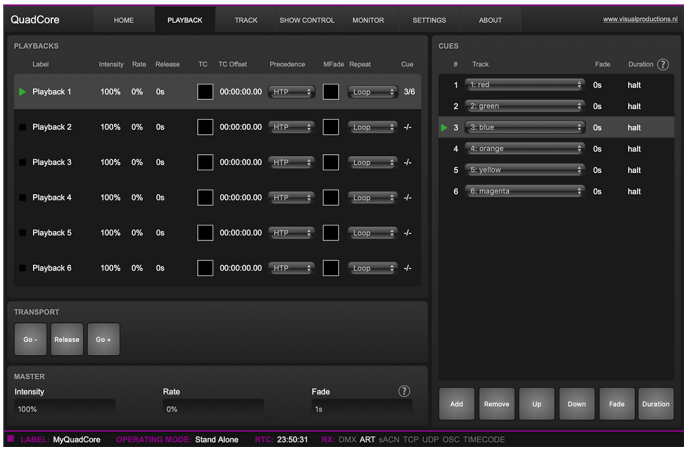
Kiosc will just simply show one button.



- Next in the actionlist Randomizer there is an action triggered by the Randomizer. The task Playback, controlled by the result of the Randomizer, jumps to a cue in Playback 1 (Parameter 1 of the task Playback).



- When the Kiosc button is pressed the Randomizer picks a number between one and six, as defined in the Randomizer task. The Playback task receives this number and triggers the corresponding cue.



9.8 User Lists

Normally a source is connected to a communication port, protocol or software feature. Actionlists will be triggered as soon as an event happens in a source. There is an exception; the User Lists are not connected to any source and therefor will not be triggered by any communication or other event. The purpose of User Lists is to have extra actionlists containing actions which can be triggered by explicitly linking to it. See page 105 for details of the *Link* feature in the *Action* task.

Chapter 10

Protocol Conversion

The QuadCore is fitted with several communication ports and additionally supports various Ethernet-based protocols. Although some protocols are predominantly used for triggering the internal playbacks (such as UDP, OSC, etc.) and some other protocols are mainly used for recording (such as DMX input, Art-Net and sACN) the QuadCore is capable of converting one protocol into another. This chapter provides an insight on which conversions are possible and how to set them up.

All possible conversions can be organised into two categories: Converting Control Protocols and Converting DMX Universe Protocols.

10.1 Converting Control Protocols

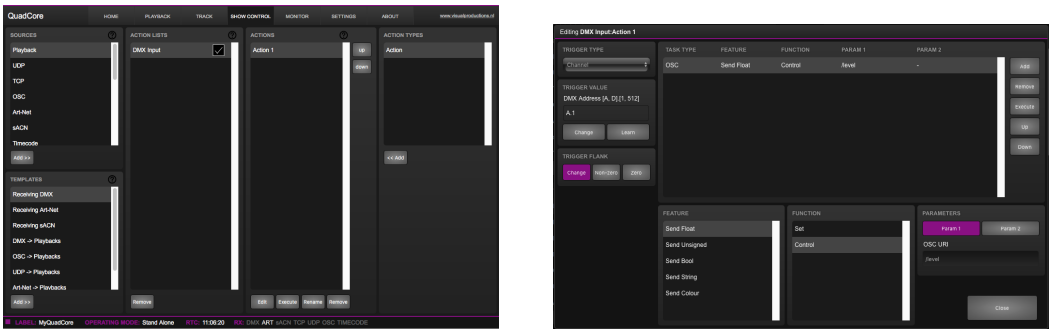
The first category of conversions comprise the protocols typically used for triggering or transporting one piece of information. The following table shows these protocols and what kind of information they are able to carry.

Control Protocols	Information
UDP	-
TCP	-
OSC	percentage [0%,100%], number, string, colour, On/Off
DMX	number [0,255]
Art-Net	number [0,255]
sACN	number [0,255]

Although DMX, Art-Net and sACN are dedicated lighting protocols and naturally fit in the next category, their individual channels lend themselves well for conveying control messages.

Setting up a conversion is done in the Show Control page. First add the incoming protocol from the 'Sources' table into the 'Action list' table. Then add an action to this new action-list. Inside this action the trigger-flank field (if visible) should be set to Change; as this action should be triggered every time the incoming signal changes. Furthermore, a task need to be added, the task-type determines which protocol is the output of our conversion. It is important that the 'function' in this task is set to 'Control'. This will make sure that the output is not a fixed value, rather it will output the information received from the incoming signal.

Please consider two examples. Figure 10.1 shows a conversion between DMX and OSC. This example assumes the DMX Port A is set to 'Input' on the Settings page.



(a) Step 1 (b) Step 2

Figure 10.1: Conversion from DMX to OSC

10.2 Converting DMX Universe Protocols

This category includes all protocols that carry a DMX Universe (a block of 512 DMX channels). These protocols are DMX, Art-Net, sACN and to some extend KiNet. The QuadCore is capable of receiving a complete DMX universe from one protocol and sending it out on a different protocol. Furthermore, it is able to merge DMX universes from multiple sources into one output protocol. All this is done with a minimal amount of configuration in the QuadCore. The following table lists examples of the conversions that can be made.

Example DMX Universe Conversions
DMX ->Art-Net
Art-Net ->DMX
DMX ->sACN
sACN ->DMX
DMX ->KiNet
Art-Net ->sACN

It is also possible to create combinations of the examples above. For instance you could set up a conversion from DMX to both Art-Net and sACN. Or merge incoming Art-Net and sACN together into the DMX output. Also, at any point it is possible to merge the incoming DMX data with the data generated by the internal playbacks.

To set up the conversion go to the Show Control page and choose the incoming protocol from the 'Sources' table and add it to the 'Action lists' table. Then add an action for each DMX Universe you wish to convert; e.g. when converting two DMX ports to Art-Net it requires two action to programmed. The trigger-type in the actions should be set to 'Universe' to make the QuadCore process the 512 channels as a whole rather then process individual channels. Each action should contain a DMX-task with the 'feature' set to 'Universe'; all DMX Universe data is first being copied into the QuadCore's internal DMX buffer. From this buffer it can be copied to the DMX outlet or the other protocols such as Art-Net and sACN. Figure 10.2 provides a schematic for this data flow.

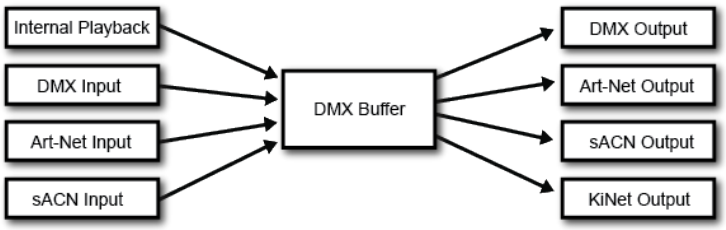


Figure 10.2: DMX merging data flow

The function needs to be set to *Control HTP*.

Function	
Control HTP	Highest Takes Precedence
Clear	

All channels are compared and the highest levels are used for the merged output (HTP precedence).

The additional 'Clear' function is not related to the data merging precedence; it is just a function to clear the whole universe to zero.

Please note that the 'Templates' table provides pre-programmed configurations for the most popular conversions.

A very typical conversion that can illustrate as an example is to convert Art-Net universes 0.0 and 0.1 to DMX output A and B respectively. Figure 10.3 shows action-list, figure 10.4 show the contents of 'Action 1' and figure 10.5 show the required configuration of the Settings page.

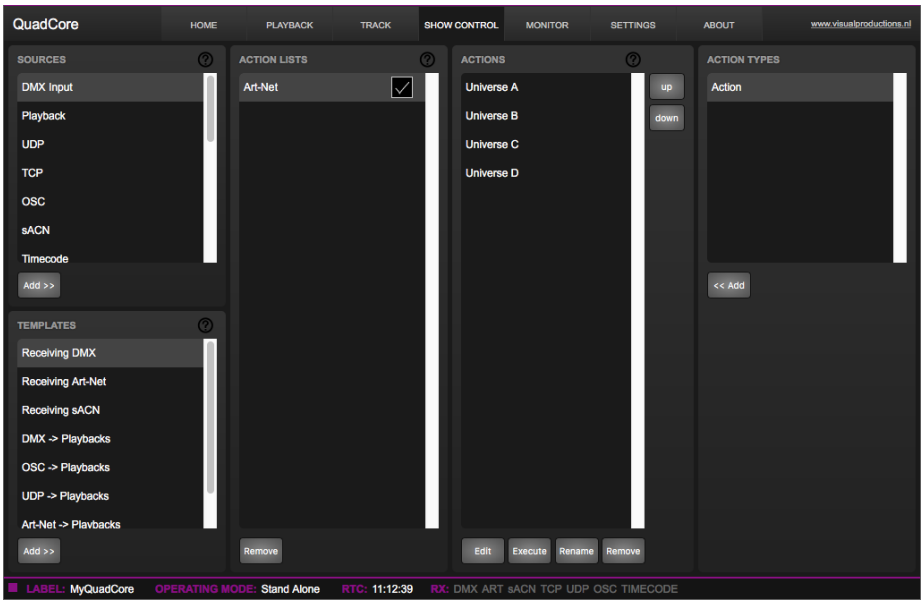


Figure 10.3: Converting Art-Net to DMX step 1

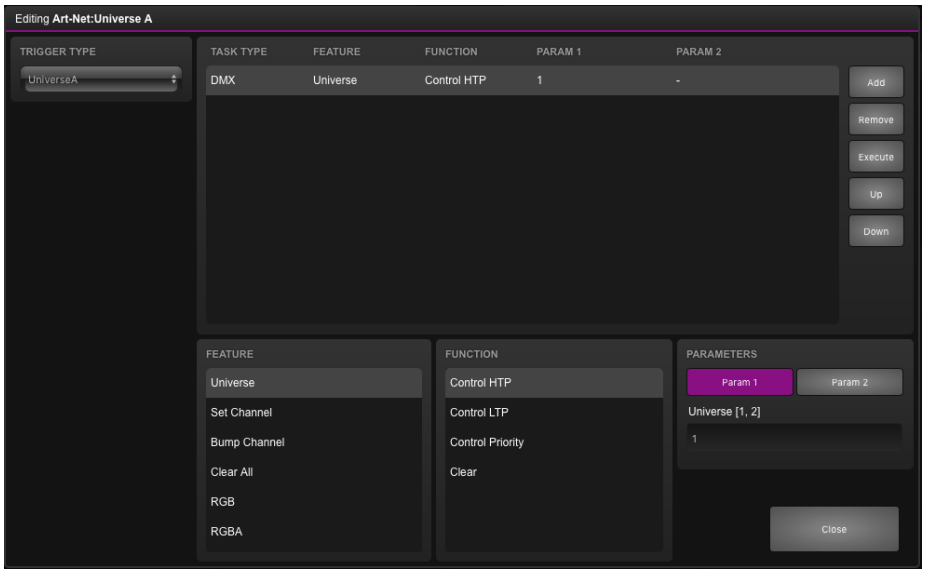


Figure 10.4: Converting Art-Net to DMX step 2

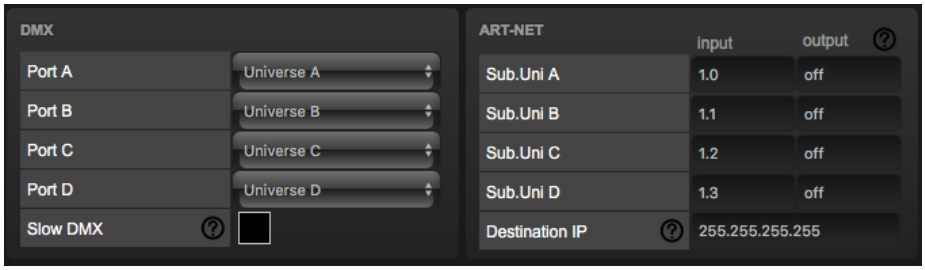


Figure 10.5: Converting Art-Net to DMX step 3

Chapter 11

Monitors

This page allows the user to inspect the incoming and outgoing data, both DMX-type data (See figure 11.1) as well as control messages (See figure 11.2). Monitoring incoming and outgoing data can help the user troubleshoot during programming.

Figure 11.1: DMX Monitor

In the Monitor page three different sources of input can be found (DMX, Art-Net and sACN), along with the control input and output sources (TCP, UDP and OSC). On the right side of the page there are the universes were the user can swap between the four of them or choose a preferable unit for displaying the requested information.

The Monitor page also presents an overview of timer and variable values. See figure 11.3

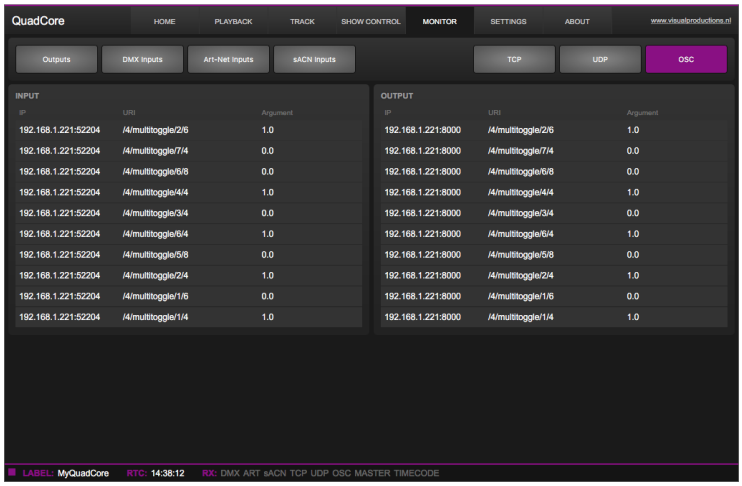


Figure 11.2: OSC Monitor

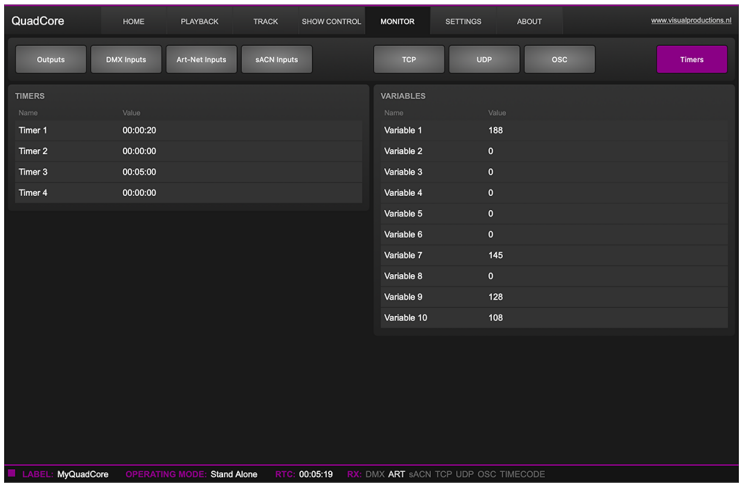


Figure 11.3: Timer & Variable Monitor

Chapter 12

Settings

The QuadCore's settings are organised into sections, see the Settings page figure 12.1. This chapter will discuss each section.

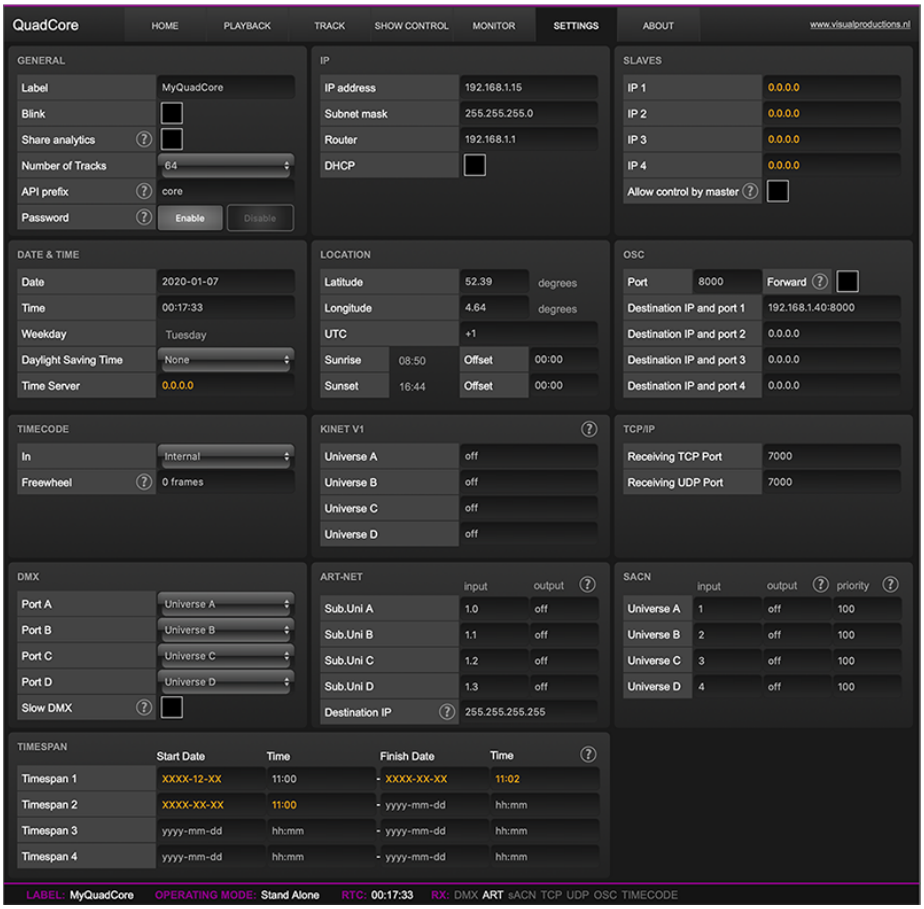


Figure 12.1: Settings page

12.1 General

You can change the QuadCore's label. This label can be used to distinguish the unit in a set-up with multiple devices.

By enabling the *Blink* checkbox the device's LED will blink to help to identify it amongst multiple devices.

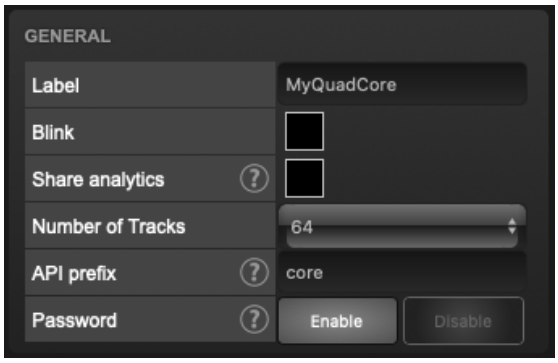


Figure 12.2: General Settings

The *Number of Tracks* drop-down determines the organisation of the Track memory. This is discussed on page 46. Changing the number of tracks will result in loosing the current content of the tracks.

The API commands discussed in appendix D start with a prefix that is set to *core* by default. When using multiple devices from Visual Productions it can be useful to assign unique labels to these prefix, especially when using broadcasted messages. Read more about feedback loops in paragraph D.4.1.

Unauthorised users can be prevented from making changes to the QuadCore by enabling the *Password* protection. Once enabled, the password can be disabled via the web-interface (using the *Disable* button) and the reset button (see figure 5.2). Long-press the reset button to disable the password protection; this will also revert the unit's static IP back to the default factory settings.

12.2 IP

The IP fields are for setting up the IP address and subnet mask of the QuadCore. The *Router* field is only required when Port Forwarding is used. You can also enable or disable the DHCP feature (For more information see chapter 5 at page 33).

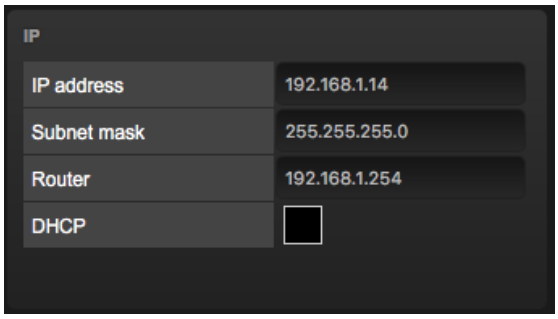


Figure 12.3: IP Settings

12.3 Slaves

This section enables the master-slave synchronisation.

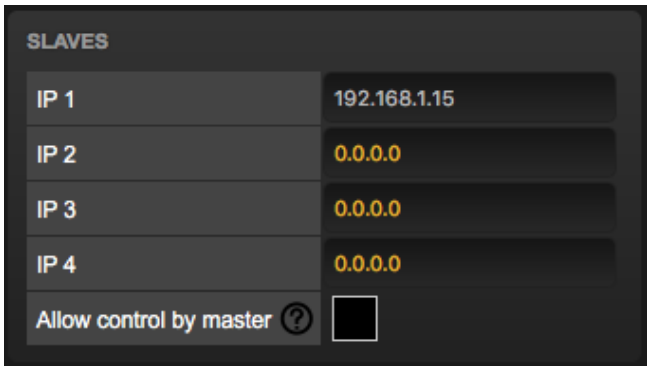


Figure 12.4: Slaves Settings

The master-QuadCore should specify the IP addresses of its slaves. When the IP is indicated in white then the master-slave connection is established, otherwise the IP is indicated in orange. For creating a system with more than four slaves, a broadcast IP can be set. A message sent to a broadcast address may be received by all network-attached hosts. A typical broadcast IP address is 192.168.1.255, however, this depends on the subnet used.

The slave-QuadCore units require the 'Allow control by master' checkbox to be enabled. Enabling 'Allow control by master' checkbox will cause playback data to be overwritten.

12.4 Date & Time

The date and time of the RTC can be set here. The clock has a back-up battery to keep the time during a power down. Daylight Saving Time (DST) is supported for the regions Europe and United States.

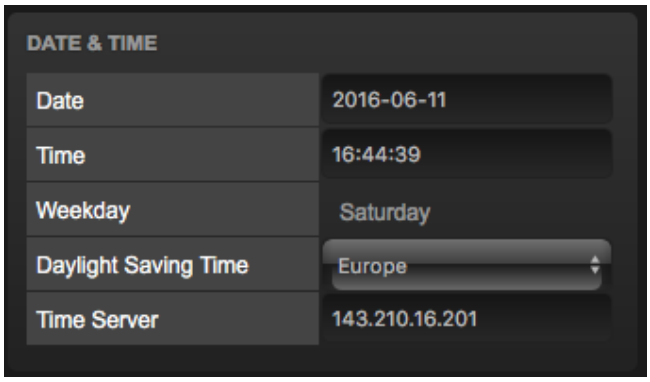


Figure 12.5: Date & Time Settings

The Time Server field allows a NTP server to specified. At start up, the QuadCore will fetch the time and date from this server. Additionally, an action can be used to fetch the time. The DST and the Coordinated Universal Time (UTC) are taken into account when obtaining the time for the NTP server.

The following table lists suggested NTP servers.

Continent	Server
North America	64.90.182.55
South America	201.49.148.135
Europe	216.239.35.8
Africa	196.23.245.74
Asia	133.100.9.2
Australia	137.92.140.80

12.5 Location

The astronomical clock in the QuadCore calculates the sunrise and sunset times based on day of the year, latitude, longitude and UTC. The latitude and longitude values define the position in the world and should be entered in degrees. The latitude value should be positive for North and negative for South, the longitude should positive for East and negative for West.

The website http://www.findlatitudeandlongitude.com/ can help discover the latitude and longitude values for the current location. The time-zone and perhaps daylight saving time of the current location is expressed in the UTC value. UTC is - in this context - equivalent to Greenwich Mean Time (GMT).

For example, Visual Productions' HQ is based in the city of Haarlem, the Netherlands. During the winter the UTC equals +1 and in the summer during day light saving time it is set to +2. So, the settings for the Visual Productions' HQ are shown in Figure 12.6.

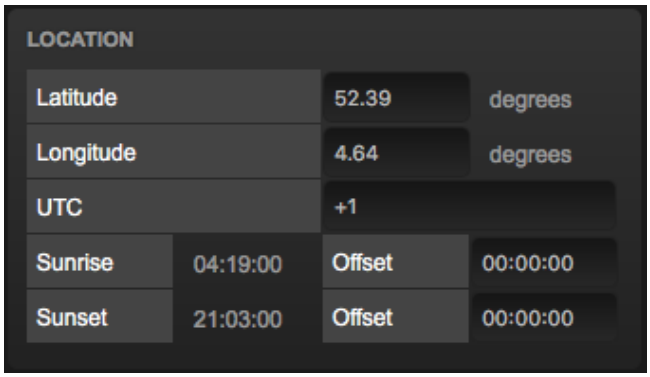


Figure 12.6: Location settings

The Offset fields allows to shift the sunrise and sunset triggers, both earlier and later. For example, to have a trigger half an hour before sunrise set the offset to -00:30.

12.6 OSC

External equipment sending OSC messages to the QuadCore need to be aware of the number specified in the 'Port' field. This is the port the QuadCore listens to for incoming messages.

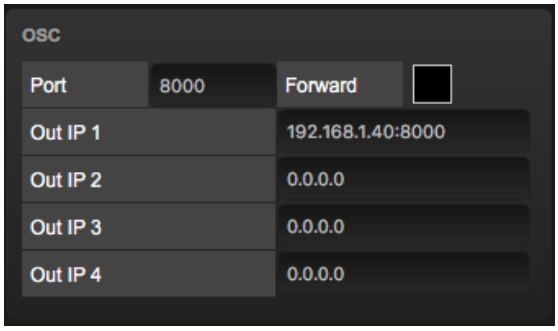


Figure 12.7: OSC Settings

The QuadCore will send its outgoing OSC messages to the IP addresses specified in the 'Out IP' fields. Up to four IPs can be specified here. Use the 'ip-address:port' format in these fields, e.g. "192.168.1.11:9000". If a field should not be used then it can be filled with IP 0.0.0.0:0. It is possible to enter a broadcast IP address like 192.168.1.255 in order to reach more than four recipients.

Enabling the *Forward* checkbox will have the QuadCore copy every incoming OSC message and send it the addresses specified in the 'Out IP' fields.

12.7 Timecode

The QuadCore can receive a timecode signal to trigger Actions or synchronise Playbacks. It supports Art-Net time-code signals. Also the QuadCore provides an 'Internal' timecode source. In this case the internal timing logic of the QuadCore is used.

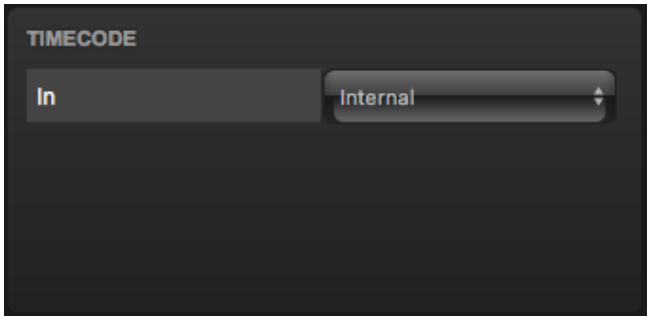


Figure 12.8: Timecode Settings

The timecode freewheel will allow continue for some frames after the timecode source stops. "Forever" can be entered to continue when no timecode is received.

12.8 TCP/IP

Defines the listening ports for TCP and UDP messages. External systems intending to send TCP or UDP message to the QuadCore need to know the unit's IP address and this port number. By default both ports are set to 7000.

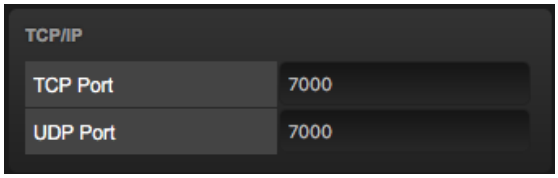


Figure 12.9: TCP/IP settings

12.9 DMX

The *DMX* settings specify whether the DMX port is an *Input* or *output*.

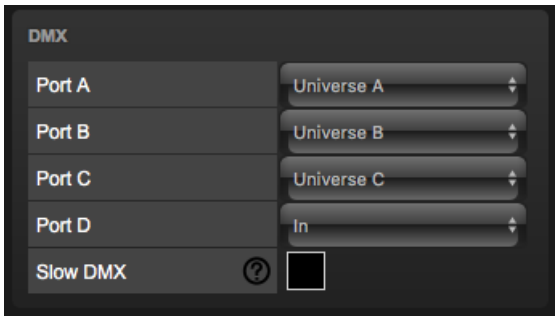


Figure 12.10: DMX settings

When the *Slow DMX* checkbox is enabled, the QuadCore will slowdown the rate at which it sends out DMX. This is done to facilitate DMX fixtures that have difficulties keeping up with the optimal DMX transmission rate.

RDM can be disabled by using the *Enable RDM* checkbox. This checkbox is repeated on the *Patch* page.

12.10 Art-Net

These universes can be mapped to any of the 256 available universes in the Art-Net protocol. The universe is entered in the 'subnet.universe' format, i.e. the lowest universe number is written as '0.0' and the highest universe number is denoted as '15.15'. The outgoing Art-Net transmission can be disabled by entering 'off' in the output fields.

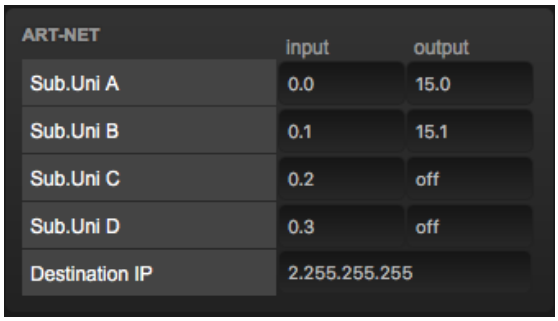


Figure 12.11: Art-Net settings

The *Destination IP* determines where the outgoing Art-Net data will be sent to. Typically, this field contains a broadcast address like 2.255.255.255 which will send the Art-Net data to the 2.x.x.x IP range. Another typical Art-Net broadcast address is 10.255.255.255. When using broadcast address 255.255.255.255 then all the devices on the network will receive the Art-Net data.

It is also possible to fill in a unicast address like 192.168.1.11; in this case the Art-Net data will be send to one IP address only. This keeps the rest of the network clean of any Art-Net network messages.

12.11 sACN

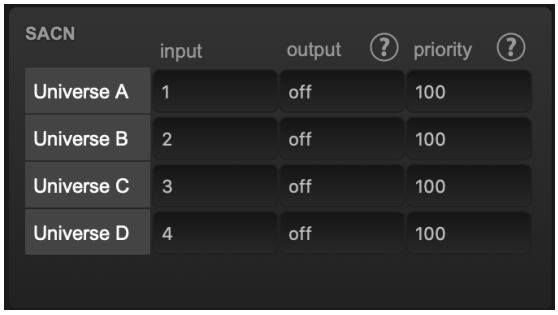


Figure 12.12: sACN settings

Set priority field to control the priority level of the output universe. The priority can be set from 0 (lowest) to 200 (highest).

12.12 KiNet v1

The QuadCore features transmission of DMX data via KiNet; it supports KiNet protocol version 1.

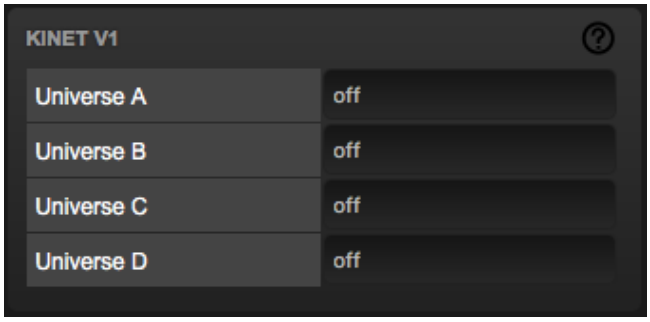


Figure 12.13: Kinet Settings

12.13 Timespan

The *Timespan* section allows for four periods of time to be defined. The beginning and end of the time periods will trigger actions in the show control page.

An advantage of *Timespans* over normal scheduler actions is that Timespans are power-cycle safe. When the QuadCore is without power during the moment of a normal scheduler trigger then it will miss that trigger. *Timespans*, however, will still receive the trigger once the power is restored after the moment has occurred.

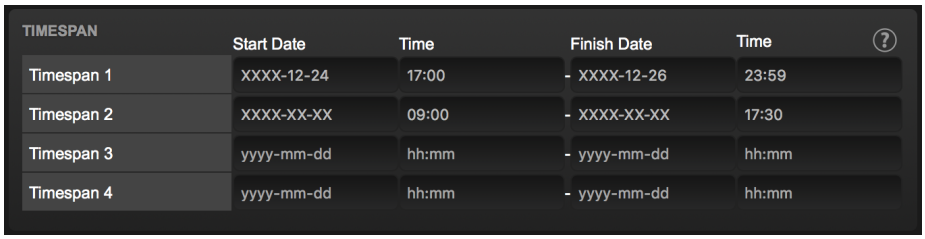


Figure 12.14: Timespan settings

Timespans are ideal for defining periods of time during a year (like Christmas or Easter) or periods during the day (like opening times).

When entering the date and time data, XX's can be used to signify a 'don't care'. This allows for easily programming of yearly, monthly or daily reoccurring events.

Chapter 13

vManager

A free-of-charge software tool called vManager has been developed to manage the devices. vManager allows for:

- Setup the IP address, subnet mask, router and DHCP
- Backup and restore the device's internal data and settings
- Perform firmware upgrades
- Set the real-time clock of the QuadCore (The computer's date and time will be used)
- Identify a specific device (in a multi device set-up) by blinking its LED
- Revert to factory defaults

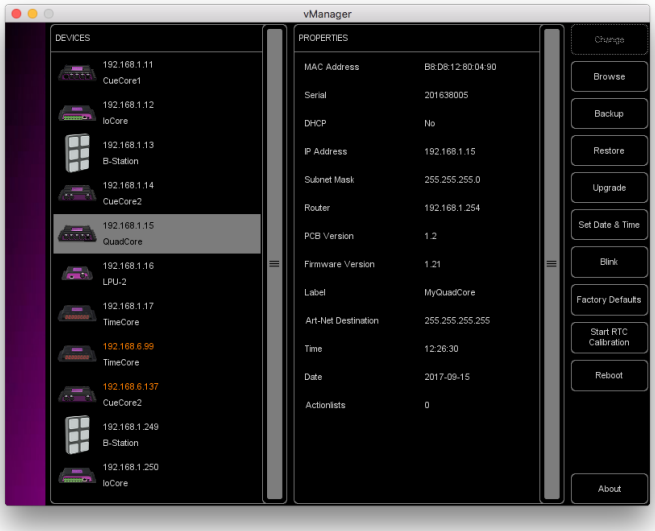


Figure 13.1: vManager

The following section explain the buttons in the vManager, as seen in figure 13.1.

13.1 Backup

Backups of all the programming data inside the device can be made. This backup file (an XML) is saved on the computer's hard-disk and can be easily transferred via e-mail or USB stick. The data of the backup can be restored via the *Restore* button.

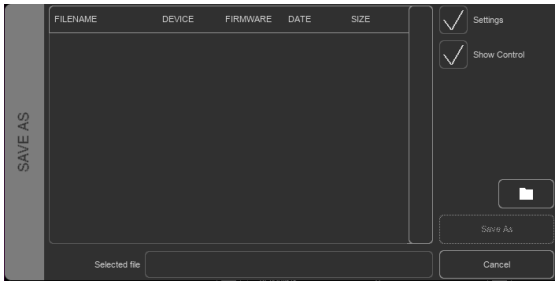


Figure 13.2: Creating a backup

Apps distributed by app stores are not allowed to access files outside this designated location. It is important to know where vManager is storing its files, in case you wish transfer a backup file to memory stick or dropbox.

The designated file location differs per operating system and is likely to be a long and obscure path. For this reason, vManager provides you with a shortcut to the correct file location. A *Folder* button can be found in the file related dialogs. Clicking this button will open a file browser at appropriate folder.

13.2 Upgrade Firmware

To upgrade the firmware, first select the device and press the *Upgrade* Firmware button. The dialogue allows for selecting from the list of firmware versions available.

Figure 13.3: Firmware upgrade

Warning: Make sure the power to the device is not interrupted during the upgrade process.

13.3 Set Date & Time

The computer's date and time can be quickly copied to the unit by selecting a device and clicking the *Set Date & Time* button. Not all Visual Productions devices feature an internal real-time clock.

13.4 Blink

The device's LED can be set to *Blink* fast for identifying the particular unit amongst multiple devices. The blinking is enabled by double-clicking on a device in the Devices list or by selecting a device and then clicking the Blink button.

13.5 Factory Defaults

All the user data like cues, tracks and actions are stored in the on-board flash memory. They will be completely erased and all settings will reverted to their defaults by pressing the *Factory Defaults* button. This action does not affect the device's IP settings.

13.6 RTC Calibration

The QuadCore features an internal real-time clock (RTC) that is used for generating scheduler triggers (date, time, sunrise, etc). In case that the clock is drifting, i.e. slowly falling behind or getting ahead of the real time, then it can be calibrated again using the vManager. The procedure is as follows:

1. Select the QuadCore
2. Click on the *Start RTC Calibration* button
3. wait approximately 30 minutes
4. Click on the *Stop RTC Calibration* button
5. Apply the recommended calibration value in the dialog (figure 13.4)

13.7 Reboot

The *Reboot* button allows you to remotely restart the device. This is useful for testing the unit's behaviour after a power-cycle.

13.8 Installing vManager

The vManager app is available on a wide range of operating systems, both mobile and desktop.

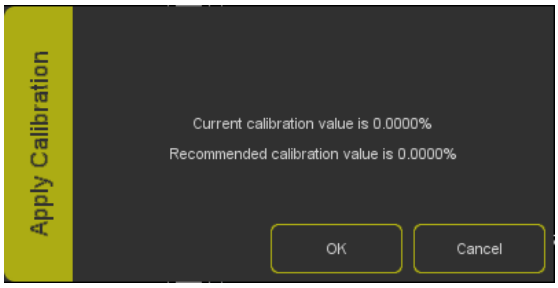


Figure 13.4: Apply calibration value

The softwares is distributed through app-stores to take advantage of receiving future software updates automatically.

13.8.1 iOS

vManager can be downloaded from the Apple iOS app-store at `https://itunes.apple.com/us/app/vman/id1133961541`.

13.8.2 Android

vManager can be found on the Google Play store at `https://play.google.com/store/apps/details?id=org.visualproductions.manager`.

Android 5.0 or higher is required.

13.8.3 Windows

Visit the Microsoft store at `https://www.microsoft.com/en-us/p/vmanager/9nblggh4s758`.

Windows 10 is required.

13.8.4 macOS

Visit the Apple macOS app store at `https://apps.apple.com/us/app/vmanager/id1074004019`.

macOS 11.3 is recommended.

13.8.5 Ubuntu

You can acquire the vManager from Snapcraft at `https://snapcraft.io/vmanager`.

Alternatively, it can be installed by using the command-line:

```
snap find vmanager
snap install vmanager
```

To update the apps later on via the command-line type:
`snap refresh vmanager`

Ubuntu 22.04 LTS is recommended. The software is only available for the amd64 architecture.

Chapter 14

Kiosc

Kiosc is an application for creating custom touch screen user-interfaces for the range of lighting controllers from Visual Productions. Kiosc is designed to have no editing capability, making it a fool-proof interface that can safely be presented to non-technical operators.

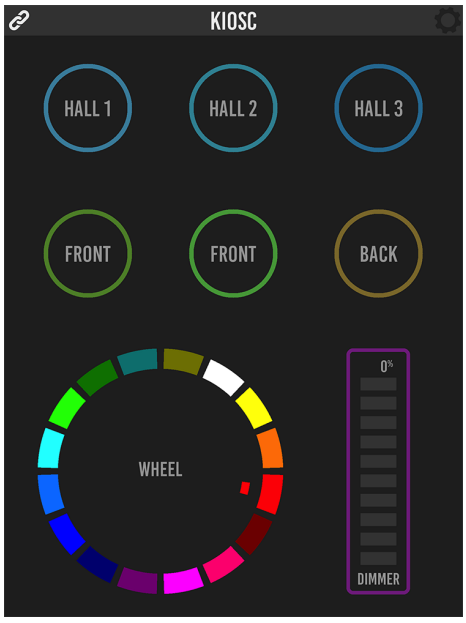


Figure 14.1: Kiosc

Kiosc is the ideal way to remote control our solid-state lighting controllers like CueluxPro, CueCore1, CueCore2, QuadCore, IoCore1, IoCore2, LPU-2, DaliCore, B-Station1 and the QuadCore. Kiosc enables you to choose scenes or presets, set intensity levels or choose RGB colours.

You can also use it to control third-party AV equipment. Kiosc speaks OSC, UDP and TCP.

Kiosc is available as software app and as a physical product. The hardware version of Kiosc is a wall-mount 7" touch screen with Kiosc pre-installed. It is powered by PoE and requires only a RJ-45 connection.

Figure 14.2: Kiosc

Please read the Kiosc manual, available from `http://www.visualproductions.nl/downloads` for more details.

Appendices

Appendix A

Trigger Types

The following tables list the different types of triggers that can be used in the QuadCore. The different types are accompanied with values and flanks.

A.1 DMX Input

Trigger Type	Trigger Value	Flank	Description
Channel	DMX Address	Change	Channel changes
Channel	DMX Address	Non-zero	Channel becomes non-zero
Channel	DMX Address	Zero	Channel becomes zero
UniverseA	-	-	A DMX level change in the first universe
UniverseB	-	-	A DMX level change in the second universe
UniverseC	-	-	A DMX level change in the third universe
UniverseD	-	-	A DMX level change in the forth universe
Receiving	-	Change	Start receiving or loose DMX signal
Receiving	-	Stop	Lost DMX signal
Receiving	-	Start	Start receiving DMX signal

A.2 Playback

Trigger Type	Trigger Value	Flank	Description
Active	Playback Index	Change	Playback starts or stops
Active	Playback Index	Released	Playback stops
Active	Playback Index	Start	Playback starts
Release	Playback Index	Change	Playback starts or finishes releasing
Release	Playback Index	Released	Playback finished releasing
Release	Playback Index	Release	Playback starts releasing
Released	Playback Index	Change	Playback starts or stops
Released	Playback Index	Playing	Playback starts playing
Released	Playback Index	Released	Playback finished releasing
Playing	Playback Index	Change	Playback starts or stops
Playing	Playback Index	Release	Playback starts releasing
Playing	Playback Index	Playing	Playback starts playing
Running	Playback Index	Change	Playback starts or pauses
Running	Playback Index	Paused	Playback pauses
Running	Playback Index	Playing	Playback starts playing
Intensity	Playback Index	Change	Playback intensity changes
Intensity	Playback Index	Non-zero	Playback intensity becomes >0%
Intensity	Playback Index	Zero	Playback intensity becomes 0%
End	Playback Index	-	Playback stops playing
CueChange	Cue Index	Change	Cue activated or deactivated
CueChange	Cue Index	Inactive	Cue becomes inactive
CueChange	Cue Index	Active	Cue becomes active
CueIndex	Playback Index	-	Active cue index changed
CueLabel	Playback Index	-	Label of the current Cue
TrackBegin	Playback Index	-	Track starts

A.3 UDP

Trigger Type	Trigger Value	Flank	Description
Message	String	-	Receive a message that matches the trigger-value
Receiving	-	-	Receive any message

The user can define his own string as the trigger value of a message. Please note that this string has a maximum length of 31 characters.

It is possible to pass a parameter along with a message. In order to do this use the syntax `trigger=value`. For example when the trigger type is set to 'message' and the trigger value is set to `intensity` then the transmitting equipment can pass an intensity level by sending `intensity=255`, where 255 can be any number in the range [0,255].

A.4 TCP

Trigger Type	Trigger Value	Flank	Description
Message	String	-	Receive a message that matches the trigger-value
Receiving	-	-	Receive any message

The user can define his own string as the trigger value of a message. Please note that this string has a maximum length of 31 characters.

A.5 OSC

Trigger Type	Trigger Value	Flank	Description
Message	URI	Change	Receive a message that matches the URI
Message	URI	Down	Receive a message that matches the URI and the value non-zero
Message	URI	Up	Receive a message that matches the URI and the value is zero
Receiving	-	-	Receive any message

The user can define his own URI as the trigger value of a message, however, the OSC specification dictate this string must start with a '/' sign. Please note that this string has a maximum length of 31 characters, including the '/'.

A.6 Art-Net

Trigger Type	Trigger Value	Flank	Description
Channel	DMX Channel	Change	Channel changes
Channel	DMX Channel	Non-zero	Channel becomes non-zero
Channel	DMX Channel	Zero	Channel becomes zero
UniverseA	-	-	A DMX level change in the first universe
UniverseB	-	-	A DMX level change in the second universe
UniverseC	-	-	A DMX level change in the third universe
UniverseD	-	-	A DMX level change in the forth universe
Receiving	-	Change	Start receiving or loose Art-Net signal
Receiving	-	Stop	Lost Art-Net signal
Receiving	-	Start	Start receiving Art-Net signal

A.7 sACN

Trigger Type	Trigger Value	Flank	Description
Channel	DMX Channel	Change	Channel changes
Channel	DMX Channel	Non-zero	Channel becomes non-zero
Channel	DMX Channel	Zero	Channel becomes zero
UniverseA	-	-	A DMX level change in the first universe
UniverseB	-	-	A DMX level change in the second universe
UniverseC	-	-	A DMX level change in the third universe
UniverseD	-	-	A DMX level change in the forth universe
Receiving	-	Change	Start receiving or loose sACN signal
Receiving	-	Stop	Lost sACN signal
Receiving	-	Start	Start receiving sACN signal

A.8 Timecode

Trigger Type	Trigger Value	Flank	Description
Time	Frame	-	Timecode frame
Receiving	-	Change	Start receiving or loose timecode signal
Receiving	-	Stop	Lost timecode signal
Receiving	-	Start	Start receiving timecode signal

A.9 Kiosc

Trigger Type	Trigger Value	Flank	Description
-	-	Change	Button/Fader goes up or down
-	-	Down	Button is pressed
-	-	Up	Button is released

When editing the Kiosc actionlist it will be possible to add different kind of actions such as Button, Fader and Colour Picker. These elements will be displayed in the Kiosc software.

More details on Kiosc can be found on page 86.

A.10 Scheduler

The scheduler depends on the *Date & Time* settings, as well as the *Location* settings that are located in the *Settings* page. These settings are discussed on page 74.

Trigger Type	Trigger Value	Flank	Description
WeekdayAndTime	-	-	Enable weekdays and specify a time (don't care 'X' can be used)
DateAndTime	-	-	Specify a specific date and time (don't care 'X' can be used)
Sunrise	-	-	When the sun rises in the morning
Sunset	-	-	When the sun goes down in the evening
DaylightST	-	Change	Daylight Saving Time period starts or ends
DaylightST	-	Stop	Daylight Saving Time period ends
DaylightST	-	Start	Daylight Saving Time period starts
Timespan	Index [1-4]	Change	Timespan starts or ends
Timespan	Index [1-4]	Start	Timespan starts
Timespan	Index [1-4]	Finish	Timespan ends

Timespans can be used to define periods of time. E.g. Christmas holiday or the daily opening hours of the venue. The timespan can be used for triggering in the show control page. If the QuadCore will be power cycled during the start or finish of a timespan, then the trigger will still be fired as soon as the unit powers up. Timespans are defined on the *Settings* page.

A.11 Randomizer

The *Randomizer* will generate a pseudo-random number, this functionality is discussed on page 62.

Trigger Type	Trigger Value	Flank	Description
Result	-	-	The Randomizer made a new value
Specific Value	Number in the range of [0,255]	-	The Randomizer made a value that matches

A.12 System

Trigger Type	Trigger Value	Flank	Description
Startup	-	-	The QuadCore has been powered up
Network Connection	-	Change	Network connection established or lost
Network Connection	-	Stop	Network connection lost
Network Connection	-	Start	Network connection established
ReleasedByMaster	-	Change	Master (e.g. CueluxPro) released or obtained connection
ReleasedByMaster	-	Stop	Master released connection
ReleasedByMaster	-	Start	Master obtained connection

Use *Startup* if something needs to happen as soon as the QuadCore is powered on. Please note that ethernet network might not be online yet.

Use the *Network Connection* trigger if something needs to happen as soon as the ethernet network becomes online after a power-cycle.

ReleasedByMaster is a trigger that will happen when the QuadCore is disconnected from CueluxPro.

A.13 Variable

Variables can contains a number between 0 and 255. This functionality is explained on page 59.

Trigger Type	Trigger Value	Flank	Description
Channel	Variable Index	-	The specified variable changes
Variable 1	Number [0,255]	Change	Variable 1 becomes = or # to the specified number
Variable 1	Number [0,255]	Equal	Variable 1 = number
Variable 1	Number [0,255]	Stop Equal	Variable 1 stops to be = to number
Variable 2	Number [0,255]	Change	Variable 2 becomes = or # to the specified number
Variable 2	Number [0,255]	Equal	Variable 2 = number
Variable 2	Number [0,255]	Stop Equal	Variable 1 stops to be = to number
Variable 3	Number [0,255]	Change	Variable 3 becomes = or # to the specified number
Variable 3	Number [0,255]	Equal	Variable 3 = number
Variable 3	Number [0,255]	Stop Equal	Variable 1 stops to be = to number
...			
Variable 9	Number [0,255]	Change	Variable 9 becomes = or # to the specified number
Variable 9	Number [0,255]	Equal	Variable 9 = number
Variable 9	Number [0,255]	Stop Equal	Variable 9 stops to be equal to number
Variable 10	Number [0,255]	Change	Variable 10 becomes = or # to the specified number
Variable 10	Number [0,255]	Equal	Variable 10 = number
Variable 10	Number [0,255]	Stop Equal	Variable 10 stops to be = to number

A.14 Timer

There are four timers in the QuadCore. This functionality is explained on page 61.

Trigger Type	Trigger Value	Flank	Description
Channel	Timer Index	Change	The timer starts or stops
Channel	Timer Index	Stop	The timer stops
Channel	Timer Index	Start	The timer starts
Time	Timer Index	-	A Stream of the current time of that timer

The timer will automatically stop when it reaches 00:00.0. An *Action* needs to be programmed in order to restart the timer and make it loop.

A.15 Actionlist

Actionslists are discussed in detail on page 56. The Actionlists have a checkbox for enabling/disabled. Changing this checkbox generates a trigger.

Trigger Type	Trigger Value	Flank	Description
-	Actionlist Index	Change	The actionlist enable checkbox changes
-	Actionlist Index	Down	The actionlist is enabled
-	Actionlist Index	Up	The actionlist is disabled

A.16 User List (1-4)

User lists have no triggers. Actions inside user lists can only be activated by other actions through *Action* task using the *Link* feature.

Appendix B

Task Types

Tasks allow you to automate the functionality in the QuadCore. All this functionality is categorized in task-types. This appendix provides a listing of the various task-types. The tables present an overview of all available features and functions per task-type.

B.1 Playback

Manipulate one of the 6 playbacks.

Feature	Function	Parameter 1	Parameter 2
Jump	Set	Playback Index	Cue index
Jump	Control	Playback Index	-
Jump	Random Control	Playback Index	-
Intensity	Set	Playback Index	percentage [0%,100%]
Intensity	Control	Playback Index	-
Intensity	Inverted Control	Playback Index	-
Intensity	Increment	Playback Index	percentage [0%,100%]
Intensity	Decrement	Playback Index	percentage [0%,100%]
Intensity	Stop Continuous	Playback Index	-
Intensity	Increment Continuous	Playback Index	percentage [0%,100%]
Intensity	Decrement Continuous	Playback Index	percentage [0%,100%]
Set Rate	Set	Playback Index	percentage [-100%,100%]
Set Rate	Control	Playback Index	-
Transport	Pause	Playback Index	-
Transport	Release	Playback Index	-
Transport	Go+	Playback Index	-
Transport	Go-	Playback Index	-
Transport	Jump	Playback Index	Cue number[1,32]
Transport	Solo	Playback Index	-
Transport	Random Solo	Playback Index	-
Play State	Toggle	Playback Index	-
Play State	Control	Playback Index	-
Play State	Inverted Control	Playback Index	-
Fader Start	Toggle	Playback Index	-
Fader Start	Control	Playback Index	-
Fader Start	Inverted Control	Playback Index	-

The *Fader Start* feature controls the playback's intensity while at the same time start the playback when the intensity is higher than 0% and release when the intensity reaches 0%.

B.2 Playback Master

Manipulate the master settings on the Playback page.

Feature	Function	Parameter 1	Parameter 2
Intensity	Set	-	percentage [0%,100%]
Intensity	Control	-	-
Intensity	Inverted Control	-	-
Intensity	Increment	-	percentage [0%,100%]
Intensity	Decrement	-	percentage [0%,100%]
Intensity	Stop Continuous	-	-
Intensity	Increment Continuous	-	percentage [0%,100%]
Intensity	Decrement Continuous	-	percentage [0%,100%]
Set Rate	Set	-	percentage [-100%,100%]
Set Rate	Control	-	-
Fade time	Set	Time	-
Fade time	Control	-	-
Release	All	-	-

B.3 Track

Manipulate the settings on the Track page or record a snapshot.

Feature	Function	Parameter 1	Parameter 2
Program	Stop	-	-
Program	Record	Track Index	-
Program	Erase	Track Index	-
Intensity Map	Clear	-	-
Intensity Map	Capture DMX	-	-
Intensity Map	Capture Art-Net	-	-
Intensity Map	Capture sACN	-	-
Snapshot	Clear	-	-
Snapshot	Capture DMX	-	-
Snapshot	Capture Art-Net	-	-
Snapshot	Capture sACN	-	-

Please refer to page 47 for more details on *Intensity Map*.

The *Snapshot* feature records a static scene from an incoming protocol like DMX, Art-Net or sACN.

B.4 UDP

Send a UDP message via the network. Specify the recipient in Parameter 2. For example "192.168.1.11:7000".

Feature	Function	Parameter 1	Parameter 2
Send Float	Set	floating point number	IP address & port
Send Float	Control	-	IP address & port
Send Unsigned	Set	positive number	IP address & port
Send Unsigned	Control	-	IP address & port
Send Bool	Set	true or false	IP address & port
Send Bool	Control	-	IP address & port
Send String	Set	text string	IP address & port
Send String	Control	-	IP address & port
Send String Hex	Set	text string	IP address & port
Send String Hex	Control	-	IP address & port
Send Bytes	Set	Hex string	IP address & port
Wake On Lan	Set	MAC Address	IP address & port

Please note that string in parameter 1 has a maximum length of 31 characters.

The *Send Bytes* enables you to send a string of characters, where the characters are specified by their hexadecimal ASCII*ASCII* code. This allows for non-printable characters - e.g. NULL (0x00) and CR (0x0D) to be sent. For example to send the string `Visual` with an added CR and LF enter `56697375616C0D0A` in parameter 2.

When using the Wake On Lan feature parameter 1 should contain the MAC Address of system's NIC (Network Interface Controller) you wish to wake up. The recommended value for parameter 2 is `255.255.255.255:7`. This broadcasts the message to the whole network at port 7 which is most commonly used for Wake On Lan.

B.5 OSC

Send an OSC message via the network. The OSC recipients are specified in the Settings page.

Feature	Function	Parameter 1	Parameter 2
Send Float	Set	URI	floating point number
Send Float	Control	URI	-
Send Unsigned	Set	URI	positive number
Send Unsigned	Control	URI	-
Send Bool	Set	URI	true or false
Send Bool	Control	URI	-
Send String	Set	URI	String of characters
Send String	Control	URI	-
Send Colour	Set	URI	RGB colour
Send Colour	Control	URI	-

Please note that string in parameter 1 has a maximum length of 31 characters, including the compulsory leading '/' sign.

B.6 DMX

Manipulate the DMX levels. These are the levels that can also be sent out via Art-Net or sACN.

Feature	Function	Parameter 1	Parameter 2
Universe	Control HTP	Universe #	-
Universe	Clear	Universe #	-
Channel	Set	DMX Channel	DMX Value
Channel	Toggle	DMX Channel	-
Channel	Control	DMX Channel	-
Channel	Inverted Control	DMX Channel	-
Channel	Decrement	DMX Channel	-
Channel	Increment	DMX Channel	-
Bump	Set	DMX Channel	DMX Value
Bump	Control	DMX Channel	-
Clear	All	-	-
RGB	Set	DMX Address	RGB Colour value
RGB	Control	DMX Address	-
RGBW	Control	DMX Address	-
XY	Control	DMX Address	-
XxYy	Control	DMX Address	-
Ii	Set	DMX Address	Intensity Value
Ii	Control	DMX Address	-
Block	Control	DMX address and footprint	Fixture count
Block RGBW	Control	DMX address and footprint	Fixture count
Block CW	Control	DMX address and footprint	Fixture count
Block CW	Inverted Control	DMX address and footprint	Fixture count

The *Bump* feature will momentarily set the value of a DMX channel and then take it back to 0, effectively creating a pulse.

XY will take an incoming position and translate it to two DMX channels. *XxYy* will translate it to four DMX channels; a 16-bit pan and a 16-bit tilt level.

The *Ii* feature can take an incoming value and translate it to two-channel DMX 16-bit value.

Block can set a number of DMX channels to the same value. Parameter 1 will denote the starting channel and parameter 2 determines how many channels will be set. The footprint field in parameter 1 offers the possibility to have 'space' between the channels.

The *Block RGBW* feature is similar to the normal *Block*. The difference is that it takes RGBW colour which will be converted to four DMX channels. Therefor, the footprint should at least be set to 4.

The *Block CW* feature is similar to the normal *Block*. The difference is that it takes a level value which will be converted to two DMX channels. Therefor, the footprint should at least be set to 2.

B.7 Time Server

Fetch a time from the NTP time server. Please refer to the Settings on page 74.

Feature	Function	Parameter 1	Parameter 2
Refresh	Set	-	-

B.8 Variable

Manipulate one of the 10 variables.

Feature	Function	Parameter 1	Parameter 2
Set Value	Set	Variable #	Number in the range of [0,255]
Set Value	Toggle	Variable #	Number in the range of [0,255]
Set Value	Control	Variable #	-
Set Value	Inverted Control	Variable #	-
Set Value	Decrement	Variable #	-
Set Value	Increment	Variable #	-
Set Value	Stop Continuous	Variable #	-
Set Value	Continuous Decrement	Variable #	Delta
Set Value	Continuous Increment	Variable #	Delta
Set Value	Control Scaled	Variable #	-
Set Value	Control Offset	Variable #	-
Refresh	Set	Variable #	-
Single Dimmer	Set	Variable #	Delta
Curve	Control	Variable #	Curve
Curve	Inverted Control	Variable #	Curve

Variables are further explained on page 59.

The Single Dimmer feature is used to increase or decrease a level by using only one switch. When controlling this task through a GPI action (via an IoCore), then closing the GPI will increase or decrease the level. Opening the GPI port will freeze on the current level. This feature is useful for controlling an intensity will just one button.

B.9 System

Miscellaneous tasks.

Feature	Function	Parameter 1	Parameter 2
Blink	Set	On or Off	-
Blink	Toggle	-	-
Blink	Control	-	-
Blink	Pulse	Seconds	-

The Blink feature controls the LED on the unit as seen in figure 5.2.

B.10 Action

Use the Link feature to have one action trigger another action.

Feature	Function	Parameter 1	Parameter 2
Link	Set	Action	-

B.11 Actionlist

Manipulate an actionlist.

Feature	Function	Parameter 1	Parameter 2
Enable	Set	Actionlist	On or Off
Enable	Toggle	Actionlist	-
Enable	Control	Actionlist	-
Enable	Inverted Control	Actionlist	-

B.12 Randomizer

Trigger the Randomizer to generate a new random number.

Feature	Function	Parameter 1	Parameter 2
Refresh	Set	Minimum value	Maximum value

The Randomizer functionality is discussed on page 62.

B.13 Timer

Manipulate on of the four internal timers.

Feature	Function	Parameter 1	Parameter 2
Playstate	Start	Timer #	-
Playstate	Stop	Timer #	-
Playstate	Restart	Timer #	-
Time	Set	Timer #	Time

B.14 Timecode

Manipulate the internal timecode generator.

Feature	Function	Parameter 1	Parameter 2
Playstate	Start	-	-
Playstate	Stop	-	-
Playstate	Restart	-	-
Playstate	Pause	-	-
Time	Set	-	Time

Appendix C

Templates

This appendix discusses the templates provided in the Show Control page.

Template	Description
Receiving DMX	Receiving DMX on all ports. DMX properties in the Settings page have to be configured accordingly.
Receiving Art-Net	Receiving DMX on all universes. Art-Net properties in the Settings page have to be configured accordingly.
Receiving sACN	Receiving sACN on all universes. sACN properties in the Settings page have to be configured accordingly.
DMX ->Playbacks	DMX port A (channel 1-6) will control the intensity of all six playbacks. When a channel is >0% it will activate the playback, when set 0% it the playback will be released.
OSC ->Playbacks	A selection of OSC Commands will control Go+, Go-, Release, Intensity and Rate for all six playbacks.
UDP ->Playbacks	A selection of UDP Commands will control Go+, Go- and Release for all six playbacks.
Art-Net ->Playbacks	Art-Net input universe A will control the intensity of all six playbacks. When a channel >0% it will activate the playback, when set 0% it the playback will be released.
Kiosc ->Playbacks	Creates a Kiosc layout with buttons and sliders to operate the six playbacks.

Appendix D

API

The QuadCore is pre-programmed to make its internal functionality available via OSC, TCP, UDP and HTTP. There is a simple API implemented for each protocol. Notwithstanding these API's, it is possible to create your own OSC, TCP and UDP implementation in the Show Control page.

The API is originally designed for external equipment to control the QuadCore, however, it is also capable to send information back. This feedback mechanism is discussed at the end of this chapter, on page 115.

D.1 OSC

The following table uses playback #1 as an example. The number '1' can be replaced by any number in the range of [1,6].

URI	Parameter	Description
/core/pb/1/go+*	-	Jump to the next cue in playback #1
/core/pb/1/go-	-	Jump to the previous cue in playback #1
/core/pb/1/jump	integer	Jump to a specific cue in playback #1
/core/pb/1/release	-	Release the playback
/core/pb/1/intensity	float	Set the playback's intensity
/core/pb/1/rate	float	Set the playback's intensity
/core/pb/release	-	Release all playbacks
/core/pb/intensity	float	Set the master intensity
/core/pb/rate	float	Set the master rate
/core/pb/fade	string	Set the master fade time
/core/pb/solo	integer	Start a solo playback

* This command does not work when the parameter is off or a 0 value.

The following table uses track #1 as an example. The number '1' can be replaced by any number in the range of [1,128].

URI	Parameter	Description
/core/tr/select	integer	Select a track
/core/tr/erase	-	Erase the selected track
/core/tr/record	-	Start recording the selected track
/core/tr/stop	-	Stop recording
/core/tr/1/erase	-	Erase track #1
/core/tr/1/record	-	Start recording track #1
/core/tr/snapshot/dmx	-	Record a snapshot of the current DMX input
/core/tr/snapshot/artnet	-	Record a snapshot of the current Art-Net input
/core/tr/snapshot/sacn	-	Record a snapshot of the current sACN input

The following table only applies when Internal is selected as timecode in at the settings page.

URI	Parameter	Description
/core/tc/start	-	Start the internal timecode
/core/tc/stop	-	Stop the internal timecode
/core/tc/restart	-	Restart the internal timecode
/core/tc/pause	-	Pause the internal timecode
/core/tc/set	string	Set the internal timecode

The following table uses actionlist #1 as an example. The number '1' can be replaced by any number in the range of [1,8]. The table also uses action #2 as an example. The number '1' can be replaced by any number in the range of [1,48].

URI	Parameter	Description
/core/al/1/2/execute	bool/float/integer	Execute action #2 inside action list #1
/core/al/1/enable	bool	Set the 'enable' checkbox for action list #1

The following table uses timer #1 as an example. The number '1' can be replaced by any number in the range of [1,4].

URI	Parameter	Description
/core/tm/1/start	-	Start timer #1
/core/tm/1/stop	-	Stop timer #1
/core/tm/1/restart	-	Restart timer #1
/core/tm/1/pause	-	Pause timer #1
/core/tm/1/set	time-string	Set timer #1 at the time-string

The following table uses variable #1 as an example. The number '1' can be replaced by any number in the range of [1,8].

URI	Parameter	Description
/core/va/1/set	integer	Set the value of variable #1
/core/va/1/refresh	-	Refresh variable #1; a trigger will be generated as if the variable changed value
/core/va/refresh	-	Refresh all variables; triggers will be generated

The following table shows how to active miscellaneous functions.

URI	Parameter	Description
/core/dmx/1	integer	Set the value of a DMX channel
/core/blink	-	Momentarily flashes the QuadCore's LED
/core/hello	-	The unit will reply with the same Hello message

D.2 TCP & UDP

TCP (Transmission Control Protocol) is a protocol for sending messages across an Ethernet network. TCP provides reliable, ordered and error-checked delivery of messages between programs running on computers connected to a local area network, intranet or the public Internet.

UDP (User Datagram Protocol) is a simple protocol for sending message across the network. It does not provide any error checking. Although UDP is a bit faster than TCP, it is less secure.

Typically either TCP or UDP is supported by various media devices like video projectors and show controllers.

The functionality within the QuadCore can controlled by using the following ASCII strings (human readable text) messages:

The following table uses playback #1 as an example. The number '1' can be replaced by any number in the range of [1,6].

String	Description
core-pb-1-go+	Jump to the next cue in playback #1
core-pb-1-go-	Jump to the previous cue in playback #1
core-pb-1-jump=<integer>	Jump to a specific cue in playback #1
core-pb-1-release	Release the playback
core-pb-1-intensity=<float>	Set the playback's intensity
core-pb-1-rate=<float>	Set the playback's intensity
core-pb-release	Release all playbacks
core-pb-intensity=<float>	Set the master intensity
core-pb-rate= <float>	Set the master rate
core-pb-fade=<text>	Set the master fade time
core-pb-solo=<integer>	Start a solo playback

The following table uses track #1 as an example. The number '1' can be replaced by any number in the range of [1,128].

String	Description
core-tr-select=<integer>	Select a track
core-tr-erase	Erase the selected track
core-tr-record	Start recording the selected track
core-tr-stop	Stop recording
core-tr-1-erase	Erase track #1
core-tr-1-record	Start recording track #1
core-tr-snapshot-dmx	Record a snapshot of the current DMX input
core-tr-snapshot-artnet	Record a snapshot of the current Art-Net input
core-tr-snapshot-sacn	Record a snapshot of the current sACN input

The following table only applies when Internal is selected as timecode in at the settings page.

String	Description
core-tc-start	Start the internal timecode
core-tc-stop	Stop the internal timecode
core-tc-restart	Restart the internal timecode
core-tc-pause	Pause the internal timecode
core-tc-set= <string >	Set the internal timecode

The following table uses actionlist #1 as an example. The number '1' can be replaced by any number in the range of [1,8]. The table also uses action #2 as an example. The number '1' can be replaced by any number in the range of [1,48].

String	Description
core-al-1-2-execute=<arg>	Execute action #2 inside action list #1
core-al-1-enable=<bool>	Set the 'enable' checkbox for action list #1

The following table uses timer #1 as an example. The number '1' can be replaced by any number in the range of [1,4].

String	Description
core-tm-1-start	Start timer #1
core-tm-1-stop	Stop timer #1
core-tm-1-restart	Restart timer #1
core-tm-1-pause	Pause timer #1
core-tm-1-set=<text>	Set timer #1 at the time-string

The following table uses variable #1 as an example. The number '1' can be replaced by any number in the range of [1,8].

String	Description
core-va-1-set=<integer>	Set the value of variable #1
core-va-1-refresh	Refresh variable #1; a trigger will be generated as if the variable changed value
core-va-refresh	Refresh all variables; triggers will be generated

The following table shows how to activate miscellaneous functions.

String	Description
core-dmx-1=<integer>	Set the value of a DMX channel
core-blink	Momentarily flashes the QuadCore's LED
core-hello	The unit will reply with the same Hello message

D.3 HTTP

HTTP (Hyper Text Transfer Protocol) is the standard protocol to access web pages. It can also be used to control the QuadCore, using the URLs listed below.

The following table uses playback #1 as an example. The number '01' can be replaced by any number in the range of [01,06].

Description	URL	Parameter Range	Example
Playback Go Forward	/ajax/pbXX/go+	-	http://192.168.1.10/ajax/pb01/go+
Playback Go Back	/ajax/pbXX/go-	-	http://192.168.1.10/ajax/pb01/go-
Playback Jump	/ajax/pbXX/jmp	[1, 32]	http://192.168.1.10/ajax/pb01/jmp=3
Release playback	/ajax/pbXX/rel	-	http://192.168.1.10/ajax/pb01/rel
Set playback intensity	/ajax/pbXX/int	[0.0, 1.0]	http://192.168.1.10/ajax/pb01/intensity=0.55
Set playback rate	/ajax/pbXX/rat	[-1.0, 1.0]	http://192.168.1.10/ajax/pb01/rate=0.55
Release all playbacks	/ajax/rel	-	http://192.168.1.10/ajax/rel
Set master intensity	/ajax/int	[0.0, 1.0]	http://192.168.1.10/ajax/intensity=0.55
Set master rate	/ajax/rat	[-1.0, 1.0]	http://192.168.1.10/ajax/rate=0.55
Set master fade	/ajax/fad	text	http://192.168.1.10/ajax/fade=3s
Start a solo playback	/ajax/pb/sol	[1, 6]	http://192.168.1.10/ajax/pb/sol=3

Description	URL	Parameter Range	Example
Record a snapshot of DMX	/ajax/tr/snapshot/dmx	-	http://192.168.1.10/ajax/track/snapshot/dmx
Record a snapshot of Art-Net	/ajax/tr/snapshot/dmx	-	http://192.168.1.10/ajax/track/snapshot/artnet
Record a snapshot of sACN	/ajax/tr/snapshot/dmx	-	http://192.168.1.10/ajax/track/snapshot/sacn

The following table uses actionlist #1 as an example. The number '01' can be replaced by any number in the range of [01,08].

Description	URL	Parameter Range	Example
Execute action	/ajax/alXX/2/exe	-	http://192.168.1.10/ajax/al01/2/exe=true
Enable actionlist	/ajax/alXX/ena	true/false	http://192.168.1.10/ajax/al01/enable=false

Description	URL	Parameter Range	Example
Blink LED	/ajax/bli	-	http://192.168.1.10/ajax/blink

You can send your HTTP GET requests to port 80.

D.4 Feedback

The QuadCore is able to send feedback to external equipment using its API, so called 'clients'. The QuadCore keeps a memory of the last four OSC clients and last four UDP clients. The clients will automatically receive updates on several playback related state changes.

Below is a table listing the messages the QuadCore will send back to its clients.

The `hello` command is ideal for polling the device; it allows you to verify that the QuadCore is online at the IP address and port that you expect.

A power-cycle will clear the internal client lists. Send `/core/goodbye` or `core-goodbye` to explicitly be removed from the client list.

Consider programming custom action in the show control when additional feedback functionality is required.

D.4.1 Preventing a feedback loop

Feedback is automatically send to a device which uses the OSC or UDP API. If the external device is also a Visual Productions unit then the feedback message could be interpreted by the external unit a new command. This can result in another feedback message being generated. An endless stream of feedback messages can stall the units involved.

This feedback loop can be prevented by assign a unique label the device's API prefix. This setting is discussed on page 73.