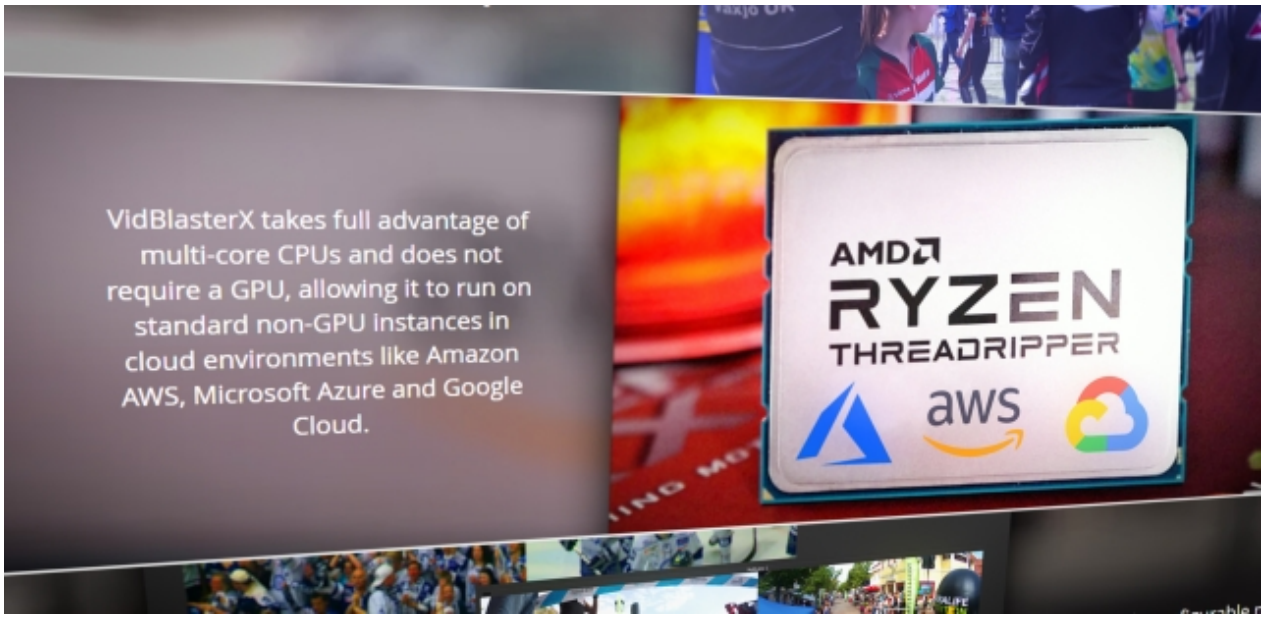
VidBlasterX

Table of contents

Welcome!

Welcome to VidBlasterX, the easy to use video production tool to create anything from a single camera recording to a large multi camera television broadcast in 4K. VidBlasterX is a versatile and powerful ultra-low latency vision mixer and video router with built-in scalers, time base correctors, frame synchronisers, IP video encoders and decoders, multiview, native NDI support, video players, recorders, keyers, effects, audio support and much more. VidBlasterX takes full advantage of multi-core CPUs and does not require a GPU, allowing it to run on standard non-GPU instances in cloud environments like Amazon AWS, Microsoft Azure and Google Cloud and achieve very low latency. VidBlasterX will put live video production right at your fingertips!



Offline viewing & printing

For offline viewing and/or printing a pdf version of this help is available.

Typographic styles

Older references to text you see on screen or hardware devices are in *italics*, newer references `look like this`. API code `looks like this`. Links to external pages look like this.

Editions

VidBlasterX6 is available in a trial and 3 licensed editions: Studio, Broadcast & Broadcast 4K. The Studio edition can have up to 25 modules per profile, the Broadcast editions 100. All editions support up to full HD (1080p) output, the Broadcast 4K edition also supports UHD & 4K video output resolutions. The (free) trial edition is equal to the Broadcast 4K edition with a watermark added to all output channels. All editions come with community support, the Broadcast editions also come with direct email support from the developer. Desktop licences for the Studio and Broadcast editions can be purchased online. Floating licence seats for the Broadcast 4K edition can be ordered by sending an email to support@vidblasterx.com.

Feature	Studio	Broadcast	Broadcast 4K
Max. modules per profile	25	100	100+
Max. internal video resolution	full HD	full HD	UHD/4K
JLCooper I/O module		supported	supported
Advanced audio channel mapping		supported	supported
Licence type	desktop	desktop	floating
Support	community	community & email	community & email

Installing

VidBlasterX is a native 64 bit program and requires a 64 bit version of Windows 7 or higher. Installing the program is straightforward and should not pose any problems. After installing a reboot may be required but only if you are prompted to do so. VidBlasterX can coexist with earlier **major** versions of the program, each will have their unique own folder and shortcut, enabling you to trial a new major version while continuing to use your current production version. Note that minor version updates **will** overwrite your current version.

Resetting the profile
Changes in Windows, in device drivers or hardware may prevent VidBlasterX from starting a (new) profile properly. If you hold down the left *Shift* & *Ctrl* keys while starting the program it will silently load the (empty) default profile instead of the last used profile. As a backup the old profile will be saved with the current timestamp.

Licensing

VidBlaster is free to try for as long as you want, you may even use it for non-commercial productions and educational purposes. Licensing VidBlaster will allow you to use it for commercial productions and will remove the trial logo.

Product philosophy

Some video software products have a limited range of configuration options and have quite rigid hardware requirements – often based on what was available at the time the product was being designed. VidBlaster is different in offering a truly modular design which creates a vast number of permutations in the way it can be configured, allows connection to a wide variety of input and output hardware, and processes a range of frame rates and image sizes from sub-SD up to UHD/4K. The result of this forward-thinking approach is that the VidBlaster software is continually pushing past the boundaries of what is theoretically possible using today's

PC hardware. The implication of that, at any point in time, is that not all combinations of features and settings are going to be possible on current PC platforms, whilst we wait for the hardware to “catch up” with the software. It is therefore essential to take advantage of the Trial edition so as to ensure that VidBlasterX is suitable for your particular application before purchasing. The process of buying a licence does not include delivery of any physical media, so you should first download a copy of the software from the web site. It will run in trial mode, with an on-screen "watermark" graphic, until you buy a licence and enter the Licence Key.

The licence

The *Studio* and *Broadcast* editions come with a desktop licence. The desktop licence is for a single (virtual) PC, i.e. it should not be installed on multiple machines, and allows multiple non-simultaneous users. If you need to run VidBlaster on multiple (virtual) desktops, you will need to buy a licence for each one. *VidBlasterX Broadcast 4K* comes with a floating licence for one or more seats. You may install the software on as many (virtual) PCs as you like, but only the number of seats ordered is allowed to run simultaneously.

Obtaining and entering a licence key

A licence key is valid for one year. Each time you extend your licence with another year you will need to purchase a new licence key. The licence is delivered by e-mail. It is primarily your responsibility to take care of it and not to lose it. Copy and paste the licence key into the Licence Key input dialog (`Help` > `Licence Key`). A notification will inform you if activation of the licence is successful.

Seats

When you purchased your VidBlasterX Broadcast 4K licence you will have specified the number of seats you wish to include. Every time you start the licensed application it will automatically check if seats are still available and if so, awarded. If no seats are left you will be notified of this and another application will need to be closed to continue. Note that licensed applications that are not reported as closed will automatically give up their seat within an hour.

Support

Support communities have been established for your convenience on both Facebook and LinkedIn. Email support is available from your reseller. If you bought your licence directly from the developer CombiTech then email support is available via support@vidblasterx.com for **Broadcast licences only**.

About CombiTech

CombiTech is a Netherlands based company owned by Mike Versteeg, developing software for over 30 years. Programs like Mscan Meteo, Mscan SSTV, StudioRack, CastBlaster, WinPodder and of course VidBlaster all have high quality, originality and ease of use in common. Mike can be found on LinkedIn, Instagram (behind-the-scenes) and Facebook (private).

SyncLok

SyncLok is a proprietary technology developed by CombiTech. Professional (broadcast) vision mixers use genlock to ensure proper synchronisation between sources. This requires expensive cameras, video players, IP and other video sources. Not all VidBlasterX users will have a budget for this, so a different solution is required. Many (software) manufacturers "fix" this issue by using one master clock, and more or less

synchronise video sources to that. This will not work for equipment that provides its own clock, like cameras and incoming IP transmissions, nor for setups with multiple video streams. It also increases latency as buffering is required. SyncLok overcomes these issues and in most cases works just as well as the original genlock. SyncLok carefully examines the timing of all video input signals and their internal routing, and dynamically adjusts clocks in real time so they are properly synchronised.

Performance

To get the best performance from VidBlasterX a lot of factors need to be considered, but probably the most important factor is the CPU. VidBlasterX makes extensive use of multi-threading, and as a result has superb multi core performance. So besides clock frequency, the number of cores is important as well. The picture below demonstrates how well VidBlasterX is able to do an even load distribution on a 32 core (64 virtual cores) Xeon 8375C CPU ingesting 22 UHD NDI streams.

Here are some more pointers to get the best performance out of your live production setup.

Video scaling
If the resolution (aka size) of video frames do not match then they will be automatically scaled. Scaling introduces scaling artefacts as well as consumes extra resources. For best performance make sure all video streams, from in- to output, maintain the same video resolution.

Video frame rate conversion
If the frame rate of an incoming video stream does not match the main frame rate, or if the outgoing video frame rate does not match that of the target device, the frame rate of the video stream will be automatically converted. Frame rate conversion almost always introduces artefacts in the form of small interruptions or jumps in the video. For best performance try to maintain a uniform video frame rate throughout your entire setup.

Submitting Bug Reports

Bug reports are appreciated and can be posted in the VidBlasterX Community. To keep the community organised, before posting please make sure you have read and followed the instructions below.

1. Confirm the problem is still present in the latest beta release, which can be found in the VidBlasterX Community. The latest beta version number can also be found by selecting *Help* > *Check for Updates* in VidBlasterX's main menu.

2. Verify the problem is not caused by hardware limitations by reducing the main video resolution and frame rate.

3. Try to pinpoint the problem as much as possible by gradually removing modules from your profile.

4. With your bug report include a screen shot of CPU core usage (Windows *Task Manager* > *Performance* > *Change graph to logical processors*), the CPU type, RAM speed and motherboard type.

5. When possible include steps to reproduce the issue.

When properly reported, rest assured bugs will be fixed as soon as possible.

The Main Window

One of the most powerful features of VidBlasterX is its modularity. By choosing the right building stones you can build the application that best suits your needs. These building stones are called modules, and can represent cameras, players, monitors etc. To add a module click *Modules* > *Add* in the main menu and select the module you want to add. Grab the title bar of the module to drag it to the required position. Grab the right or bottom border of the display to drag it to the required size. The collection of all modules, their positions and all their settings is called a profile, which can be saved and loaded to enable simple configuration ("application") switching.

The following commands are available in the main menu.

File > Load Profile
Loads previously saved profile from disk.

File > Save Profile
Saves current profile to disk.

File > Clear Profile
Removes all modules.

File > Lock Profile
When the profile is locked the main video resolution and frame rate cannot be changed and all modules are locked in place.

File > Exit
Exits program.

View > Macros
Opens the *Macros* editor.

View > Playout Controller
Opens the *Playout* controller.

View > Appearance > Full Desktop
Toggle between windowed and full desktop (all monitors) mode.

View > Appearance > Full Screen
Toggle between windowed and full screen (current monitor only) mode.

View > Advanced > API Command Stack
Used for debugging purposes. Opens the *API* window, showing the API command stack. All internal and

external commands sent through the API can be monitored here. To prevent unnecessary delays, the window is only updated when visible.

View > Advanced > MIDI Event Log
Used for debugging purposes. Opens the *MIDI* event log, showing detected midi devices and notes/control codes received. To prevent unnecessary delays, the window is only updated when visible.

View > Advanced > VidBlasterX Log
Used for debugging purposes. Opens VidBlasterX's internal log.

View > Advanced > X-keys Event Log
Used for debugging purposes. Opens the X-keys event log, showing detected X-keys devices and data received. To prevent unnecessary delays, the window is only updated when visible.

Modules > Add
Select a module to add to the profile.

Modules > Remove
Select a module to remove from the profile. Right-clicking the module and selecting *Remove Module* has the same effect.

Modules > Grid Size
Change the size of the grid. When moving modules they will snap to this grid when released.

Modules > Properties > High Quality Scaling
Set/clear the *High Quality Scaling* property of all/selected video modules. When set, scaling down of video frames is drastically improved, but at the expense of resources (both memory bandwidth and CPU).

Settings > Video Resolution
Selects the main video resolution, i.e. the video resolution used internally by all modules (except those that have a setting that overwrites this locally). Note you should not change this setting while broadcasting, recording or streaming, as it will cause an interruption.

Settings > Video Frame Rate
Select the main video frame rate, i.e. the frame rate used internally by all modules (except those that have a setting that overwrites this locally). Note you should not change this setting while broadcasting, recording or streaming, as it will cause an interruption.

Settings > Audio Standard
Select the audio standard (*EBU* or *SMPTE*) used to convert analog audio levels to digital.

Settings > Audio Buffer
Select the size of internal audio buffers in frames. Lower this setting to lower overall latency, but depending on system performance this may also introduce audio interruptions. Default setting is 2 frames.

Settings > ASIO Support
Enable this setting if you need access to a device that only offers an ASIO driver. In all other case ASIO drivers offer lower latency at best, or create system timing interference or even instability at worst. As video input and output typically have a 1 frame latency, a much lower audio latency has no value.

Help > Help
Opens the VidBlasterX help site in the default web browser.

Help > About
Displays the *About* window showing the program's version number and copyright.

Help > Licence Key
Opens a dialog to enter your licence key.

Modules

VidBlasterX has a unique modular design that allows you to configure the program exactly to your needs, both from a technical and from an ergonomic perspective. Modules can be roughly divided into 2 types: video and control modules. Video modules always include a display and perform video related functions. Control modules have no display and are used for other (control) purposes.

All modules can be easily positioned by grabbing their title bar and dragging them to the required position. Upon release they will snap to the grid set in the main menu (*Modules > Grid Size*). To resize a video module hover the mouse over the right (horizontal resize) or bottom (vertical resize) edge of the video display until the mouse cursor changes. Then hold down the left mouse button and drag the module to the desired size. The module will resize so that the display always keeps a constant aspect ratio. Upon release the right edge will snap to grid. Modules can have one or two tally lights: a program (red) / preview (green) / fx (white) tally on the left and a status tally (often used to indicate on/off status) on the right.

Each module has a popup menu, aka a context menu, which can be made visible by right clicking the module. Menu items that are common to most modules are listed and explained below, more specific items are listed on the respective module's page.

On
Activates the module's main function. Some modules automatically restore their last status.

Off
Deactivates the module's main function.

Video Source
The module takes its video from this source.

Audio Source
The module takes its audio from this source. By default this is set to either the default Windows recording device or, when present, the output of Audio Mixer 1. If no source is available, or the previously selected source is no longer available, this setting will default to *No sound*.

Audio Channels
The number of audio channels to use or *Auto* for automatic detection.

Settings
A *Settings* dialog will appear with settings related to this module.

Action > On Click > On/Off
When this flag is set, clicking the display has the same function as toggling on/off.

Action > On Click > Play/Pause
When set clicking the display has the same function has clicking the *Play* button or *Stop* button.

Action > On Click > Preview
When set clicking the display has the same function as selecting this module as source on the PVW 1 bus.

Action > On Click > Program
When set clicking the display has the same function as selecting this module as source on the PGM 1 bus.

Properties > Alias
Opens a dialog allowing you to enter a module's *Alias*, which can be a more descriptive or shortened version of the module's name. E.g. for a *Camera* module this could be "CAM1" or the name of the camera operator, for an *NDI Output* module this could be the name of the actual source (e.g. "PGM").

Properties > Alpha
VidBlasterX internally uses premultiplied alpha. Preferably all video streams from external sources that carry an alpha channel have a premultiplied alpha channel. If this is not possible, set this option to *Straight* to let

VidBlasterX do the required conversion (note this requires additional resources). Select *Ignore* (default) if VidBlasterX should ignore the alpha channel entirely.

Properties > Audio Meter
Allows you to select if the audio meter is visible. The audio meter has a range from -52 to 0 dBFS, divided in 3 segments: green from -52 to -20 dBFS (SMPTE) or -18 dBFS (EBU), yellow up to -10 dBFS and red up to 0 dBFS. The channels indicate the audio peak value in dBFS, with a peak-hold line indicating the highest peak value in the last 750 ms. By default the meter is visible.

Properties > Auto Scale
Almost all video and graphics are automatically scaled, centered and pillar/letterboxed throughout the program. In some cases however, like in the Still Store module, automatic scaling can be undesirable and this flag can be unchecked to disable this feature.

Properties > Collapsed
To conserve space most modules can be collapsed by double clicking the title bar or setting this flag. Double click the title bar again or clear this flag to return the module to its original state. Note a collapsed module still consume resources as it continues to function.

Properties > Controls
Allows you to select which controls, e.g. buttons and selection lists, are visible. The module will automatically resize to accommodate for the visible controls. Controlless modules remain fully operational through their popup menu and API.

Properties > Dock
When a module is docked to another module, dragging either module will make both modules move in unison. This feature is often used to allow grouping of, e.g. switcher, modules so they can be easily repositioned.

Properties > High Quality Scaling
When set, scaling down of video frames is drastically improved, but at the expense of resources (both memory bandwidth and CPU).

Properties > OSD
Text display over the video in the display is called *OSD*. This can be switched *Off*, it can be *Local* (i.e. from the module itself) or coming from the *Source* module. Normally OSD text is an overlay and not part of its video output. Set the *Embedded* flag to make the OSD text part of the video stream itself, which can be useful for e.g. remote monitoring or recording.

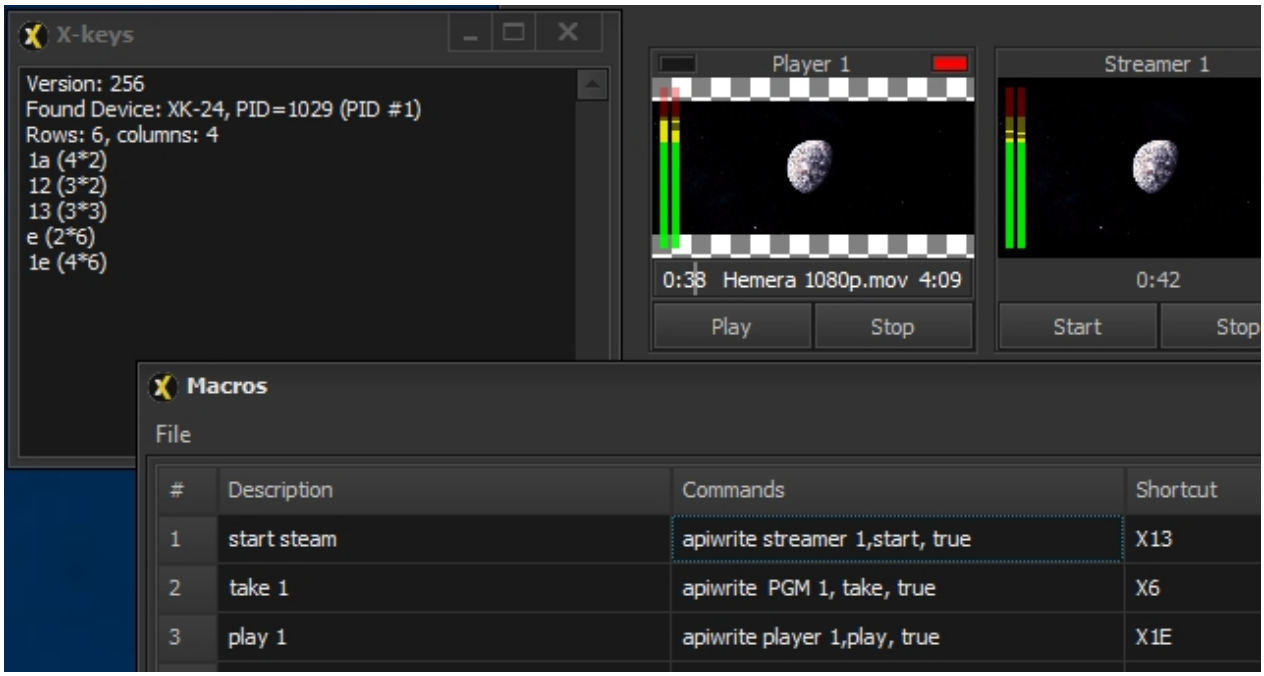
Properties > Size
Sets the module's display size as percentage of the native video resolution.

Clear Module
Clears the module's input (file or source) and display, leaving the module to output (transparent) black video.

Remove Module
Clears and removes the module.

Macros

Click *View > Macros* from the main window to open the *Macros* window. Macros form a very powerful tool to execute one or more tasks with a single command, like pressing a button on a keyboard or external controller. Macros can also be triggered (remotely) using the API. Macros can be cascaded, where one macro executes one or more other macros. The macro language is in plain English and identical to the API commands. Each macro has a serial number, an optional description, one or more commands and an optionally assigned shortcut. Each item can be viewed and edited inline by clicking it twice (once to select and once to start the editor). A macro can be easily tested using the popup menu's *Execute* command. Shortcuts can either be entered manually or automatically detected.



The Macros window consists of one or more tabs, each holding up to 100 macros. Each macro is defined by its number, a description, one or more commands and a shortcut.

#
The macro number, used when referencing macros. Each tab can hold up to 100 macros.

Description
Optional description of the macro.

Commands
One or more commands to be executed. Available commands are *delay* (e.g. *delay 1000* to add a 1 s delay), *loop* (loops back to start of macro) and the full API command set (e.g. to start the first player use *apiwrite player 1, play, true*). Individual commands are separated by a semicolon.

Shortcut
To execute a macro outside the API it requires a shortcut to be assigned to it. Typically this will be a function key combined with *Shift*, *Ctrl* and/or *Alt* (e.g. *Shift+Ctrl+F1*). Commands from midi input devices are entered as M<deviceid>:<channel><key>, e.g. *M0:0,1* for device #0, channel 1 and key code 1. Midi commands that have a variable third parameter, like rotary knobs and faders, can map this value to the command string using the %midi% placeholder. Commands from x-keys input devices are entered as X<keynumber>, where the key number is hexadecimal and starts in the top-left corner of the controller. Shortcuts can be viewed and edited as text inline, or you can "record" a shortcut by placing the shortcut in editor mode and pressing the desired key on the keyboard or button on the midi device.

The Macros window's main menu has the following commands.

File > Load
Loads new macros. Note this replaces all macros on all tabs, if you wish to add a set of macros use *Tab > Add From File*.

File > Save As
Saves all macros and tabs.

File > Clear
Erases all macros and removes all tabs but the *General* tab.

Tab > Add Empty
Adds new tab with specified name.

Tab > Add From File
Adds new tab with macros from file. All macros and macro references are renumbered if required.

Tab > Remove
Removes currently selected tab.

Tab > Rename
Renames currently selected tab.

Tab > Save As
Saves currently selected tab and al its macros to specified file.

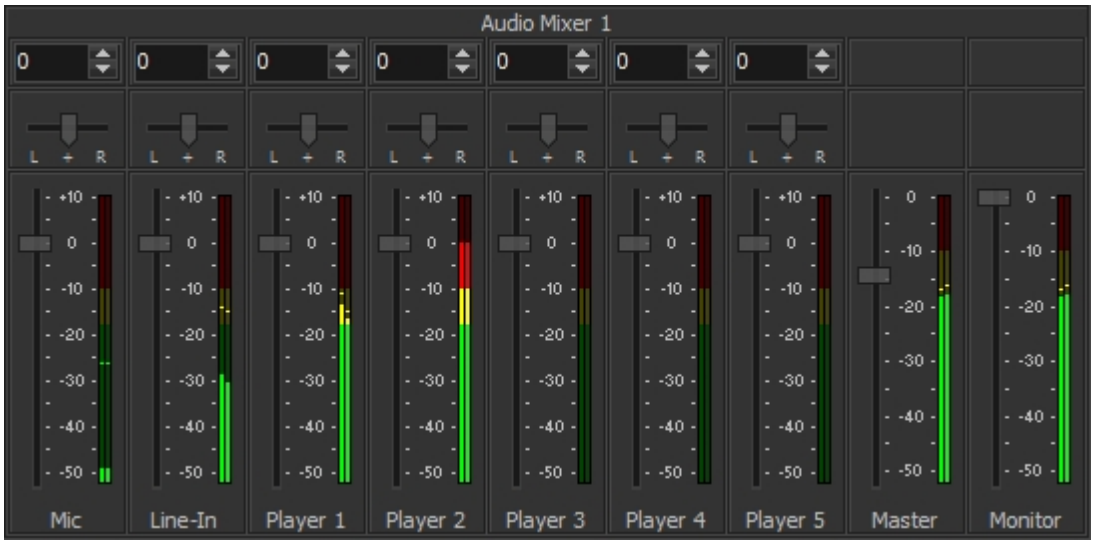
Related videos

Audio Overview

VidBlasterX has a flexible audio system built around one or more Audio Mixer modules, offering high quality, low latency multichannel audio mixing and routing. Audio can be ingested from internal modules or external devices, mixed, and output to modules like the Recorder and Streamer and/or to an external audio device.

The Audio Mixer Module

The Audio Mixer module can best be described as a modular digital audio workstation. Several channel strips can be combined to create an audio mixer with the desired number of inputs and outputs. A channel strip typically consists of a fader to control the channel's volume, an audio peak meter to monitor the channel's (pre-fader) audio levels in dBFS and an input gain and pan setting. The module makes dealing with various audio sources transparent as different sources, internal and external, can be used interchangeably. Audio from each source is converted to a high quality (floating point) internal audio format. Rate matching is applied to keep latency low and constant, both per source and overall.



Right click a channel strip to open its popup menu. The following additional menu entries are available.

<input> Audio Follows
Select a module here which tally status will control the volume: if the module's tally is selected (white) or program (red) the fader will go up to its previous position, if it's selected for preview or not at all the fader will go down to mute the channel. If the channel strip's input is a module and a bus is selected here, the status of the corresponding video input will control the volume (active is up, not active is muted). Default this is off (*None*).

<input> Channel Mapping
By default an input is always mapped 1:1 to the output. If the input signal is stereo then this menu enables you to instead map its left or right channel to all outputs. If the input has more channels than the mixer's output then this menu enables you to select the first output channel and map a subset to the output. For more advanced mapping, select the *Advanced Channel Mapping* menu entry.

<input> Default
Set all controls on the channel strip to their default values.

<input> Rename
Enables you to change the channel strip's bottom caption.

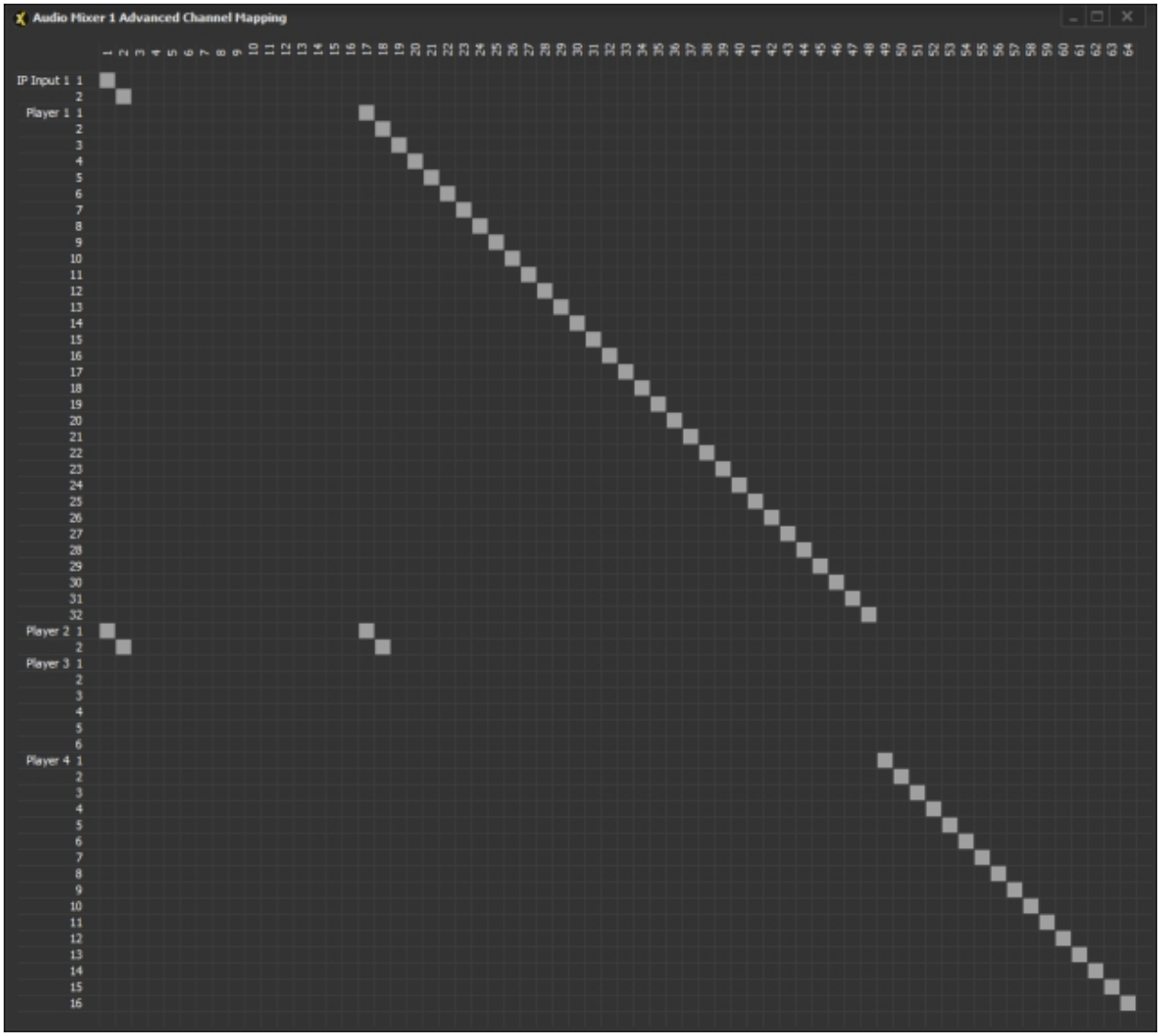
<input> Remove
Removes the channel strip.

Monitor Output
This option is only available when right clicking the *Monitor* channel strip. Select the external audio device that will be used to monitor audio, typically this will have monitor speakers or headphones connected. If output audio is monitored downstream, select No sound and optionally choose the required number of output channels (see *Output Channels*).

Output Channels
Normally the number of audio output channels is dictated by the output device selected in *Monitor Output*. If no device is selected (*No sound*), this is where you can set the number of audio output channels.

Add Channel Strip
Adds a new channel strip for the selected input. Audio can come from internal *Modules* and external *Audio Devices*.

Advanced Channel Mapping[1]
Opens a window with an input/output matrix. Maps any number of input channels to any number of output channels. Copies and mixes are created fully automatic.



Properties > Gain
Gain can be set from -99..0 dBFS. This setting can be used to attenuate audio from modules, to control the Windows level setting for external devices or even control a hardware preamp for external devices (determined by its driver). Default setting is 0 dBFS.

Appearance > Pan
Allows panning of the signal, i.e. change the distribution between left & right channels.

[1] Available in Broadcast edition only

Video Sources

VidBlasterX supports just about every live video source imaginable: webcams, video cameras, studio cameras, video capture devices and sources that are connected via IP, be it a local network or the internet. Through screen capture any part of the desktop can also be made available as video source, allowing you to show videos from websites like YouTube, capture video from programs like Skype or Google Hangouts, stream games or include spreadsheets or powerpoint presentations. Each video source can be accessed in VidBlaster by one or more *Camera*, *DeckLink Input*, *IP Input*, *NDI Input*, *Screen Capture* or *Web Input* modules.

The Camera Module

The *Camera* module is used to ingest a video stream from a video camera, often through the use of a (DirectShow compatible) video capture device. Right click the module to open its popup menu. The following additional menu entries are available.

Video Device
A list of all video devices available, typically video capture cards and webcams.

Video Input
If the video device has more than one input (e.g. a video capture device with HDMI and SDI inputs), you can select the active input here.

Video Standard
If the video device supports analog video standards (like PAL and NTSC), you can select the desired standard here.

Video Resolution
This submenu enumerates all video resolutions that are supported by the video device. *Auto* is the default setting where the setting that best matches the main video resolution will be automatically chosen. Note some capture cards only offer the correct video resolution after a video signal is connected. Other video cards, like those from Blackmagic Design, are even more demanding and will only work if you select the exact same video resolution and frame rate as that of the applied video signal.

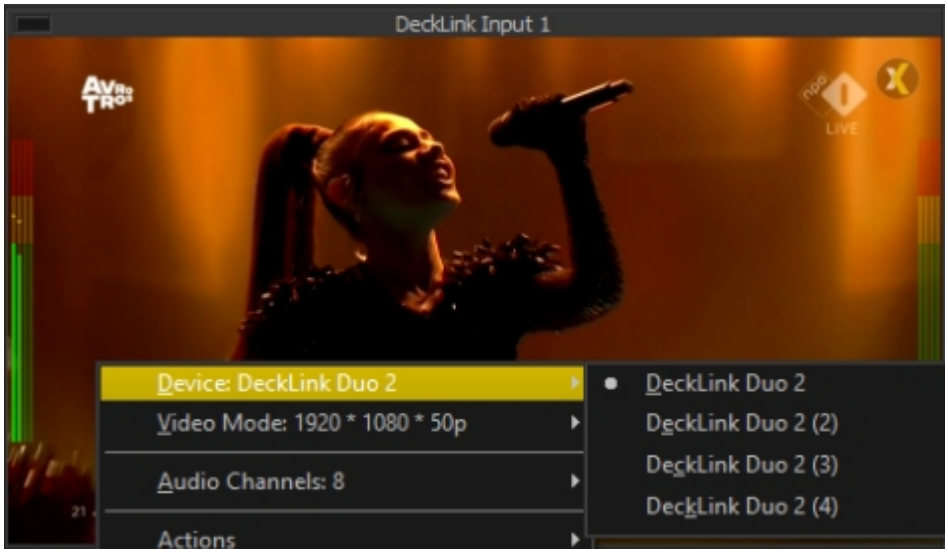
Video Frame Rate
Set the video device frame rate. When *Auto* is selected (default), the main frame rate will be used. Note a camera or video capture device may only support one specific frame rate, either because it is native to its design or because it cannot convert the source video signal's frame rate. In this case, selecting a different frame rate may either have no effect or result in no video. Typical examples are webcams, which often use either a fixed frame rate or adjust their frame rate (exposure) according to lighting conditions. Selecting a different frame rate for these webcams has no effect.

Video Frame Rate Multiplier
Converts the device's frame rate. This can be useful in case of driver errors, or when the capture card works at double the desired frame rate. Usually the default *Auto* settings works best.

Properties > Advanced Device Settings
Opens the DirectShow driver's property panel for the selected video capture device. Note not all drivers support this. The property panel is usually intended for advanced users only.

The DeckLink Input Module

The *DeckLink Input* module is used to ingest audio and video from a Blackmagic Design DeckLink video capture device directly through the DeckLink API giving the best possible performance.



Right click the module to open its popup menu. The following additional menu entries are available.

Device
A list of all available DeckLink devices.

Video Mode
Some DeckLink devices, especially the older generations, will only work if you manually select the signal format here that is applied to the device. Newer cards feature an automatic video mode detection enabling video mode selection to be done fully automatic, this sub menu will be disabled for these devices.

The IP Input Module

The *IP Input* module is used to process and decode audio and video from incoming streams using HTTP, RTMP, RTSP, SRT, TCP or (multicast) UDP network protocol. Popular codecs like H.264, MJPEG and MPEG-TS are all supported. Right click the module to open its popup menu. The following additional menu entries are available.

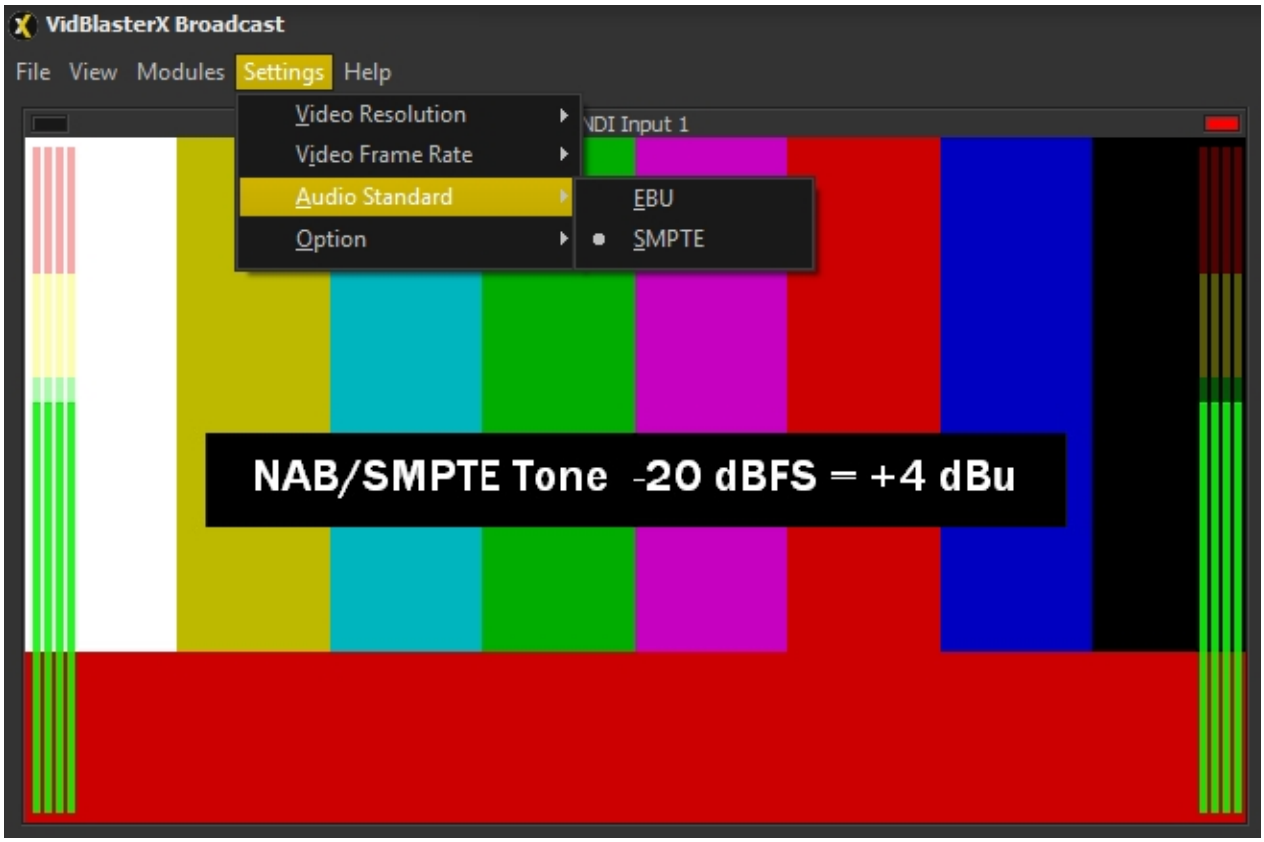
Open
Displays a dialog to enter the URL of the stream. Login details can be passed in the URL by inserting *<user>:<password>*@ before the IP address, e.g. rtsp://user:password@192.168...

Open Recent
Displays a list of recently opened URLs.

Properties > Auto Reconnect
Automatically tries to re-establish the connection when lost.

The NDI Input Module

The *NDI Input* module is used to ingest a video stream from an NDI compatible source connected to a local network. NDI™ (Network Device Interface) is a standard created by NewTek to make it easy to develop video-related products that share video on a local Ethernet network. NDI is natively supported by VidBlasterX, no further software needs to be installed. Right click the module to open its popup menu. The following additional menu entries are available.



Source
A list of all NDI sources found on the network, grouped per machine. If the list is not up-to-date then wait a few seconds and open this menu again.

Properties > Audio Only
If many NDI audio streams need to be received, setting this flag will save resources by stopping video decoding and display.

Properties > Frame Buffer Size
Because NDI does not offer a steady stream of video frames (networks don't guarantee that) a PLL is used to retrieve a stable clock. The size of its buffer is selected here (default 2 frames). The trade-off here is frame dropping vs. latency and the best setting will depend on the quality of your network. If it is not possible to use flow control in your network, a buffer size equivalent to 300 ms may be necessary to overcome TCP ACK timeouts.

Properties > Hold Last Frame
By default NDI streams are assumed to be clocked video streams and video will turn to black if sufficient frames are lost. By enabling this option the last frame will remain displayed indefinitely. This option is required when using NDI tools like Scan Converter which only send intermittent video frames.

The Screen Capture Module

The *Screen Capture* module is specifically designed to capture (part of) the Windows desktop. This is often used to incorporate spreadsheets, websites or video conversations in a production. Right click the module to open its popup menu. The following additional menu entries are available.

On
Starts screen capture. Due to the nature of graphics cards, screen capture (reading data back from the graphics card) is very inefficient and can quickly introduce interrupts in system timing. Only run screen capture modules that are actually being used.

Off
Stops capture.

Capture > Window
The window to capture.

Capture > Display
The display to capture, also referred to as screen capture in case of a single display setup.

Capture > Rectangle > Fixed
Shows the *Capture Rectangle* which can be dragged to the desired position. The size of the captured part of the screen equals the main video resolution, ensuring 1:1 pixel mapping for best capture quality.

Capture > Rectangle > Scalable
Shows the *Capture Rectangle* which can be dragged to the desired position and resized. The aspect ratio will remain equal to that of the main video resolution, ensuring the entire video frame will be filled by the (scaled) captured part of the screen.

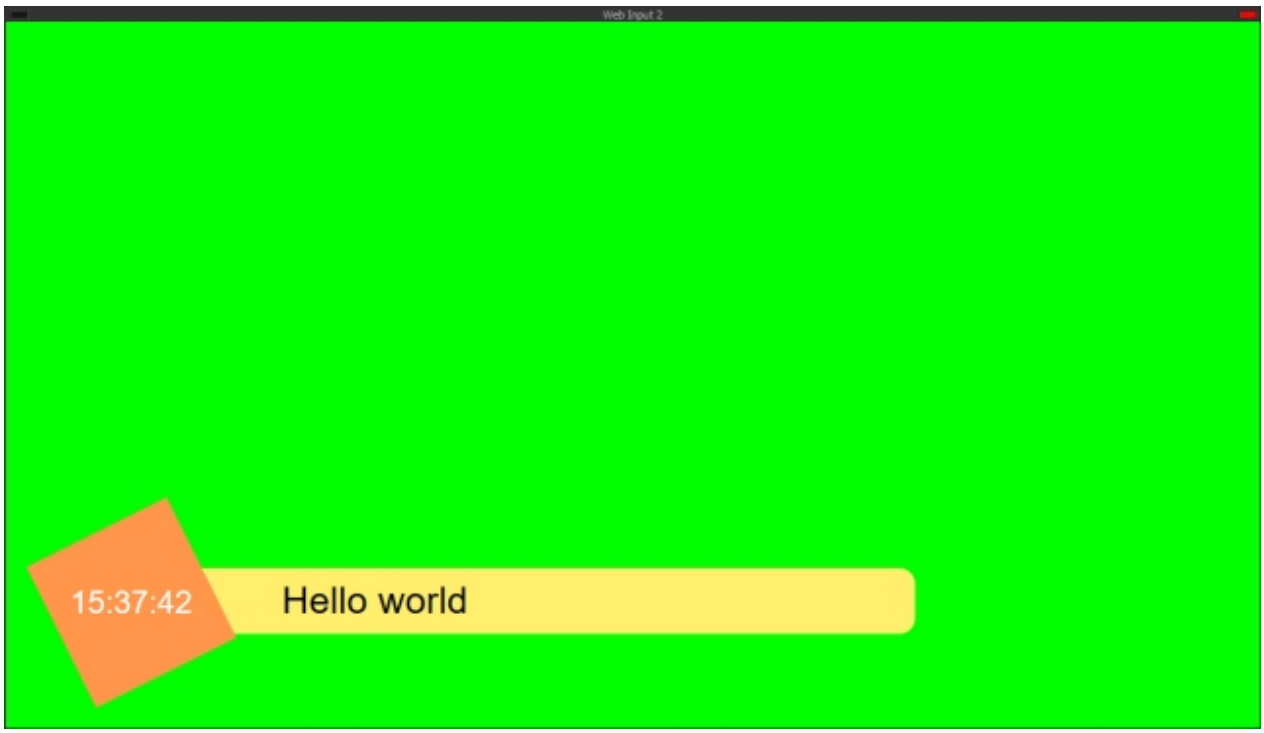
Capture > Rectangle > Freeform
Shows the *Capture Rectangle* which can be dragged to the desired position and resized at will. The captured part of the screen will be automatically scaled and centered in the video frame.

Properties > Mouse Cursor
Check this flag if you want to add a (simulated) mouse cursor to the capture.

Related videos

The Web Input Module

The *Web Input* module is used to capture (dynamic) web content. The module is typically used to overlay information or scoreboards generated from a local or remote server. Like any input module it is not interactive, if this is required use a browser and the *Screen Capture* module. The following additional menu entries are available.



Open
Displays a dialog to enter the URL of the web page.

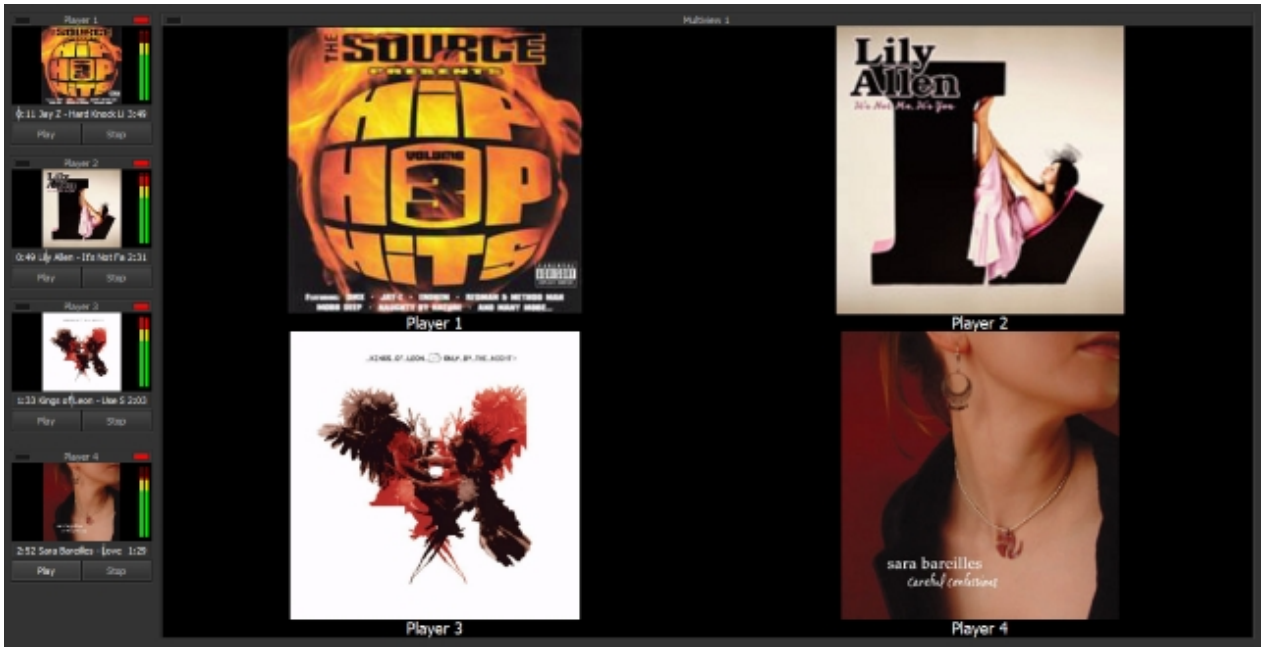
Open Recent
Displays a list of recently opened URLs.

Adding Video, Stills & Text

VidBlasterX supports many video formats, including Windows and Apple native formats as well as more advanced broadcast formats asl well as audio files. The files are loaded into a *Player* module which can be used to play the video and/or audio. Still graphics can be grabbed from a video stream, or loaded from disk, and subsequently "played" using the *Still Store* module. Text can be added using the Character Generator. If Microsoft Office is installed VidBlasterX can also play powerpoint presentations as a basic slide show. Finally the *Playout* controller can be used to fully automate the process of playing video, audio and execute macros and API commands. Although best results will be achieved if all material is prepared in the correct resolution and frame rate, all modules take care fully automatically of any required scaling and/or frame rate conversion.

The Player Module

The *Player* module is used to play a multitude of video formats with support for transparency. Note transparent video can have both straight or premultiplied alpha, but for performance reasons premultiplied is preferred. Interlaced video is automatically detected and converted to deinterlaced video at twice the original frame rate. Player can also play MP3 files and, when present, will extract and display the embedded "album art" graphic.



Right click the module to open its popup menu. The following additional menu entries are available.

Open
Displays the *Open* dialog to load a file. Instead of using this command, you can also drag & drop a file on the module.

Play
Starts or continues playing the file. If the file is pausing at the last frame, this command will cause a rewind (cue).

Stop
Pauses the player. If the player was already paused, this command will cause a rewind. To stop and rewind a playing file click *Stop* twice.

Action > On Program > Autoplay
Select this flag to automatically start the player when it is selected in the program bus followed by an automatic transition at the end of the file.

Action > On Program > Play
Select this flag to automatically start the player when it is selected in the program bus. The player will continue to play until it reaches end of file.

Properties > Loop
When this flag is set the video is looped, i.e. plays continuously. Audio will be disabled.

Properties > Timer
When checked (default) the player's time index is shown in the display (OSD).

Related videos

The Still Store Module

The *Still Store* module facilitates grabbing of a still, i.e. a single frame of video, and subsequently making this available as source and for saving to disk. It can also be used to "play" stills loaded as graphic files from storage devices. As this module can also hold transparent images, it is often used together with an Effects bus to create video overlays. One or more *Still Store* modules can be added by selecting *Modules* > *Add* > *Still Store*. Right click the module to open its popup menu. The following additional menu entries are available.

Grab
Grabs a still from the selected video source. Instead of using this command, you can also drag & drop a file on the module.

Open
Opens a dialog to load a graphic from disk. Note images with transparency (png with alpha) are not scaled to preserve quality.

Save
Saves the current still to disk. If the *Auto Save* flag is set the still will be saved without further interaction, if it is off a *Save As* dialog will open.

Properties > Auto Save
When this flag is checked the usual *Save file* dialog after clicking the *Save* button will be suppressed and the file will be saved without further interaction. The file name and location is derived from the last manual save, where the name is extended with an underscore followed by a 3 digit serial number (e.g. lastname_001.jpg). Numbering will reset to 1 after each startup and existing files will be overwritten without warning. If any errors occur during saving (e.g. rights issues) they are suppressed and the file is not saved. If a still has not yet been saved manually, and as a result no file name is yet available, this menu item will be disabled and the flag ignored.

The Character Generator Module

The *Character Generator* module, or *CG*, places anti-aliased text on a transparent background, that can then be keyed over other sources using the Effects bus. It is often used in combination with the Still Store module to add a background. Right click the module to open its popup menu. The following additional menu entries are available.

Font
Displays Font dialog to select font properties.

Related videos

The Powerpoint Module

One or more *Powerpoint* modules can be added by selecting *Modules* > *Add* > *Powerpoint*. This module uses Microsoft PowerPoint (required install) to convert ppt(x) files to their individual slides, which can then be easily browsed in this module. Right click the module to open its popup menu. The following additional menu entries are available.

Open
Displays the Open dialog to load a file. Instead of using this command, you can also drag & drop a file on the module or use the drop down list control.

Previous Slide
Select previous slide.

Next Slide
Select next slide.

First Slide
Select first slide.

Properties > Auto Slide Advance
Slides will automatically advance if an interval is set here.

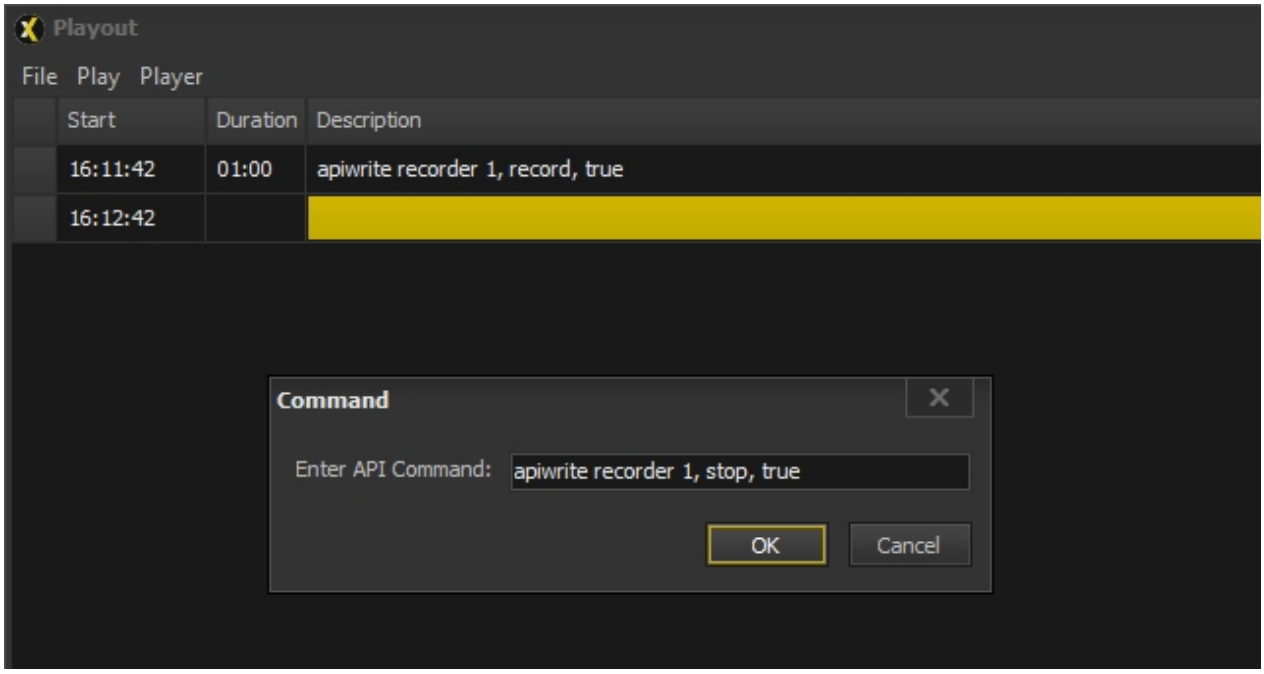
Properties > Loop
If the last slide is currently presented the next slide will be the first slide if this flag is set.

Related videos

Playout Controller

The *Playout* controller, which can be accessed from the main menu by selecting *View* | *Playout*, is used to automatically play video files and/or execute API commands and/or macros in a specific sequence. As a result, it allows you to fully automate content delivery to your viewers.

Here's an example of creating a timed recording, that automatically stops after 1 minute.



A description of all controls, commands and options is listed below (right click the module to get access to its popup menu).

Playlist
Holds all entries in the playout controller. Each entry consists of the following items:

Status
This column shows the status of each entry in the playout list, where each status is represented by the following symbols. Note non-active entries can be clicked to select the required status.

→	Active entry, e.g. file playing.
X	Entry to be skipped.
↑	Loop back to the start of the playlist.

Start
The start time of the entry. Note this is automatically computed using either the clock (first entry) or the start time of the previous entry increased by its duration.

Duration
The duration of the entry. Note this can be edited. By default commands have zero duration.

Description
The description of the entry. By default this will be the file name or command but it can be edited for clarification.

Note entries can be moved up and down using drag & drop.

Add Command
Opens command dialog and adds API command to the end of the playlist.

Add File
Opens file dialog and adds file to the end of the playlist. Files can also be added by dragging them from other applications and dropping them on the playlist.

Add Folder
Opens folder dialog and adds all files in the selected folder to the end of the playlist.

Clear
Clears the playlist.

Delete
Delete the selected entry from the list.

File | Load
Loads new playlist, overwriting the current one.

File | Save
Saves current playlist.

Insert Command
Opens command dialog and inserts API command before selected entry.

Insert File
Opens file dialog and inserts file before selected entry.

Play | ***Play***
Starts playout.

Play | ***Stop***
Stops playout.

Player
Player used to play video (or audio) files in the playlist.

Monitoring Video

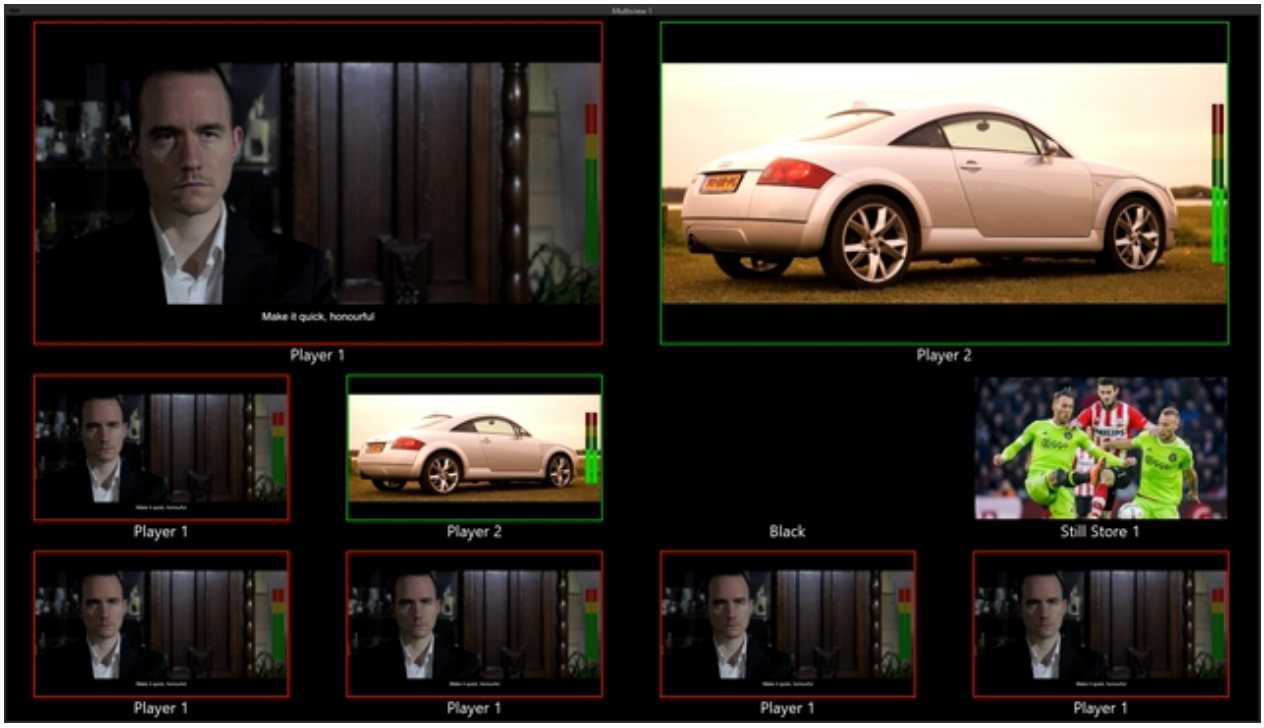
Video from any module can be monitored in the Monitor module. Multiple video sources on a single display, with optional alias and tally, can be monitored in the Multiview module.

The Monitor Module

Monitor modules can be used to monitor video from any source module (e.g. a camera module or the video switcher's program bus). Right click the module to open its popup menu.

The Multiview Module

Multiview modules can be used to monitor video from one or more input or output modules (e.g. a Camera module or an NDI Output module) including optional alias, tally and audio meters. In turn this module can be used as a source for other modules enabling you to send the Multiview video to remote monitors (e.g. over NDI). Right click the module to open its popup menu. The following additional menu entries are available.



Layout
Select the multiview layout.

View <number>
Select the video source for each view. Note this option is only available if *Auto* mode has not been set.

Properties > Audio Peak Meters
When set, each view includes an audio peak meter for the source.

Properties > Auto
When auto mode is set the sources for the views will match those in the program bus (the first two sources in *+2* layouts will be PVW 1 & PGM 1).

Properties > Label
When set, each view includes the source's alias.

Properties > Source Tally
When this option is set tally status is taken from the view's (NDI) source instead of local bus tally.

Properties > Tally
When set, each view includes the source's tally.

Recording

For "live to disk" recording the Recorder modules are available in VidBlasterX. Recordings can be saved in various formats. Other ways to record video is by routing it to one of the output modules using an external recorder, or streaming it and record at another (server) location.

The Recorder Module

The *Recorder* module records video from any module together with audio from any module or audio device directly to the hard drive in various formats. Right click the module to open its popup menu. The following additional menu entries are available.

Record/Pause
Starts/pauses recording. Recording starts when the tally turns red.

Stop
Stops recording.

Container
Select the desired video container. The audio and video codecs are automatically chosen. The default audio codec is AAC except for the MPEG-2 containers where MP2 is used and for MKV where Vorbis audio codec is used. The default video codec is H.264 except for the MPEG-2 containers which use the MPEG-2 codec. Note both the Matroska and MPEG-2 TS containers have the advantage that the file will always be readable, even if it is not properly closed (e.g. because of a power outage during recording).

Video Bitrate
Select or enter the target video bitrate. VBR is used if the (default) *Auto* setting is selected.

Audio Bitrate
Select the target audio bitrate.

Recording Path
Opens a dialog where you can select the folder and file name of the next recording(s). The default location is the *My Videos* folder and the default file name is *VidBlasterX*. A timestamp (_yymmddhhmmss) and recorder index (_n) will be automatically appended, e.g. VidBlasterX_210818143000_1 for a recording on August 18th 2021 at 2.30 PM with Recorder 1. For optimal performance, especially with HD video resolutions, it is highly recommended to select a dedicated SSD drive here.

Properties > Timer
When checked (default) the recording time is shown in the display (OSD).

Streaming

VidBlasterX includes a *Streamer* module that enables you to stream live video over a LAN or the internet. As a result, setting up a multicast stream or streaming to a (Adobe or Wowza) flash media server is easy. This means all popular streaming video services (aka Content Delivery Networks) are supported, including DaCast, Facebook and Youtube.

The Streamer Module

The *Streamer* module is used to send a video stream over a local, wide or global network. Right click the

module to open its popup menu. The following additional menu entries are available.

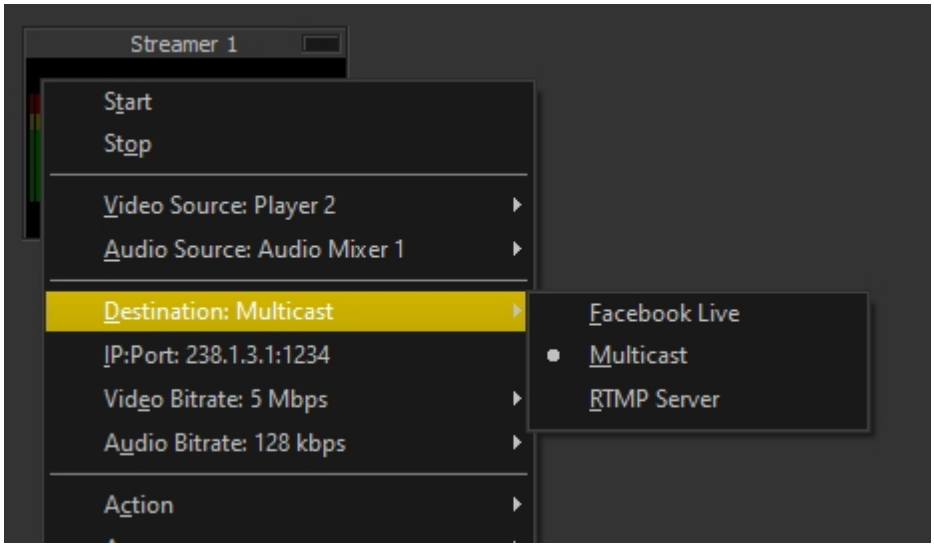
Start
Starts streaming. Note that due to stream setup and buffering it may take up to 30 seconds for the video stream to becomes visible on the server. Streaming starts when the tally turns red.

Stop
Stops streaming. Due to buffering, it is good practice to stream at least 30 seconds extra before stopping the stream. This ensures all viewers have seen the end of the streamed video when watching via a server.

Destination > Facebook Live
Presets the module to stream to Facebook, the only requirement is to enter the *Stream* address.

Destination > Multicast
Sets up a multicast stream. Requires the IP address and port to be entered (*IP:Port*). Multicast uses UDP protocol with MPEG-2 Transport Stream container and H264 video and AAC audio codec.

Destination > RTMP Server
Sets up an RTMP stream for a flash media server. Requires the *Server Username*, *Password* and *Stream* to be entered. Some services only require the stream address, in which case username and password should be left blank. RTMP streaming uses a flash video container (FLV) and H264 video and AAC audio codec.



Video Bitrate
The target video bitrate in Mbps.

Audio Bitrate
The target audio bitrate in kbps.

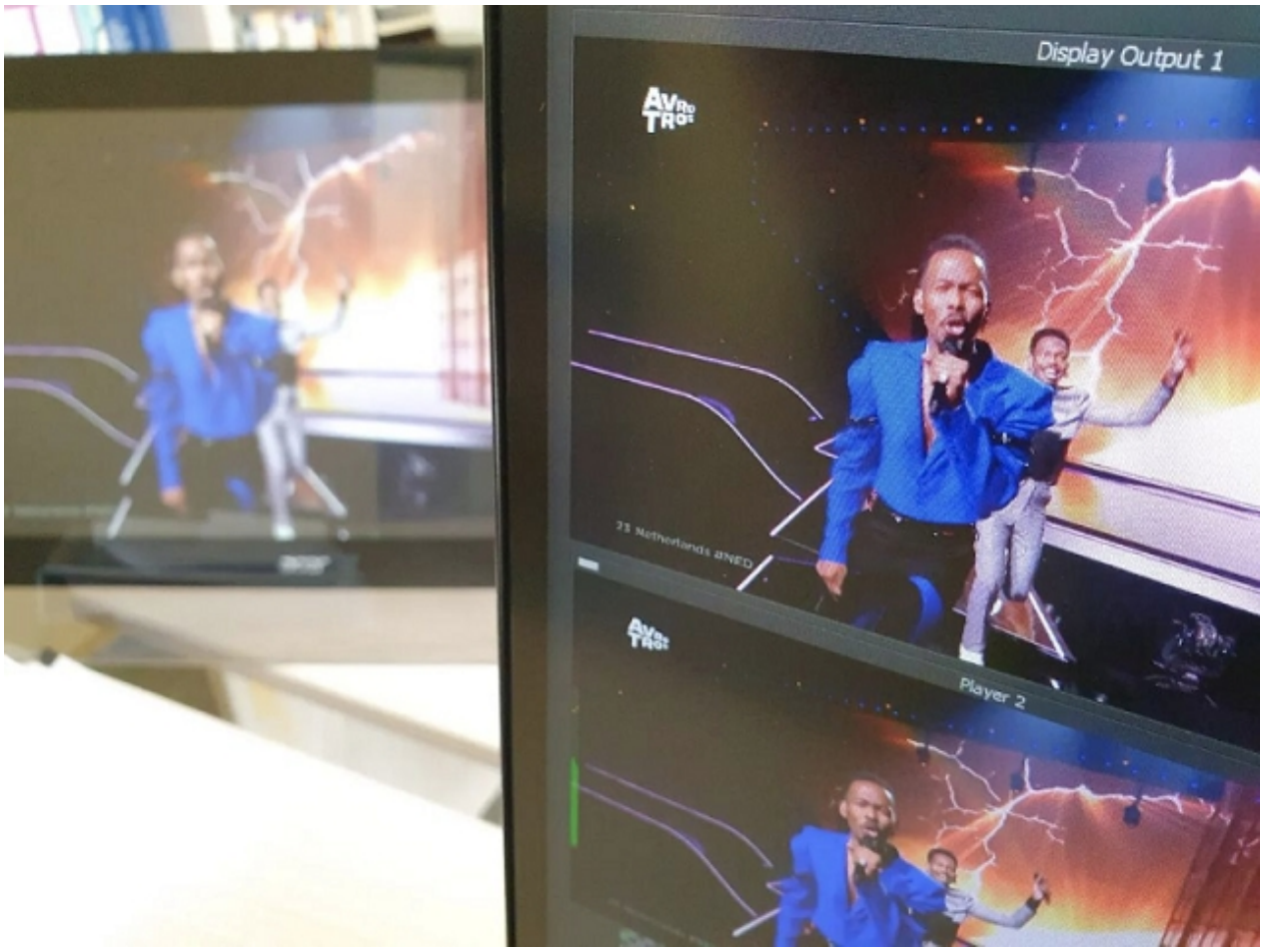
Properties > Timer
When checked (default) the streaming time is shown in the display (OSD).

Video Output

There are several ways video can be output: it can be recorded to disk, streamed over a network or internet, displayed on another monitor or output via a Blackmagic DeckLink card.

The Display Output Module

The *Display Output* module is used to send video or still images from any source to another monitor using one of the graphics cards already installed. Typically this graphics card drives an on-floor monitor or a video projector. Note the Windows desktop must be extended to include this card.



Right click the module to open its popup menu. The following additional menu entries are available.

On
Starts output of the video.

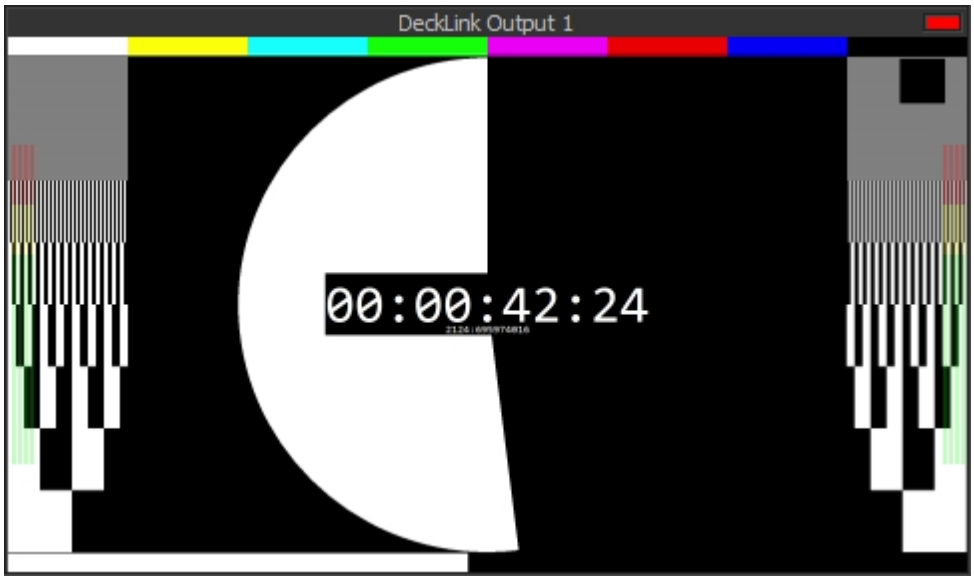
Off
Blanks the video output (black).

Video Output
The video destination, i.e. the display to which the video will be sent.

Properties > Anamorphic
To output anamorphic widescreen video, i.e. horizontally compress 16:9 video frames to 4:3, check this flag.

The DeckLink Output Module

The *DeckLink Output* module is used to send audio & video from any source to a Blackmagic Design DeckLink card like the Intensity Pro (HDMI) or DeckLink Duo 2 (SDI). The module will also perform any required scaling and/or frame rate conversion.



Right click the module to open its popup menu. The following additional menu entries are available.

Device
The video destination, i.e. the DeckLink card's output.

Video Mode
Video resolution, frame rate and optional interlacing to be output.

The IP Output Module

The *IP Output*[1] module is used to stream audio & video over a local network or the internet.

SRT
SECURE
RELIABLE
TRANSPORT

Right click the module to open its popup menu. The following additional menu entries are available.

On
Starts output.

Off
Stops output.

Protocol
The streaming protocol to be used. Currently supported protocols are *SRT Caller* and *SRT Listener* (server).

IP
If *SRT Caller* protocol is used, this is where the IP address or domain of the receiver is entered.

Port
Port used for SRT.

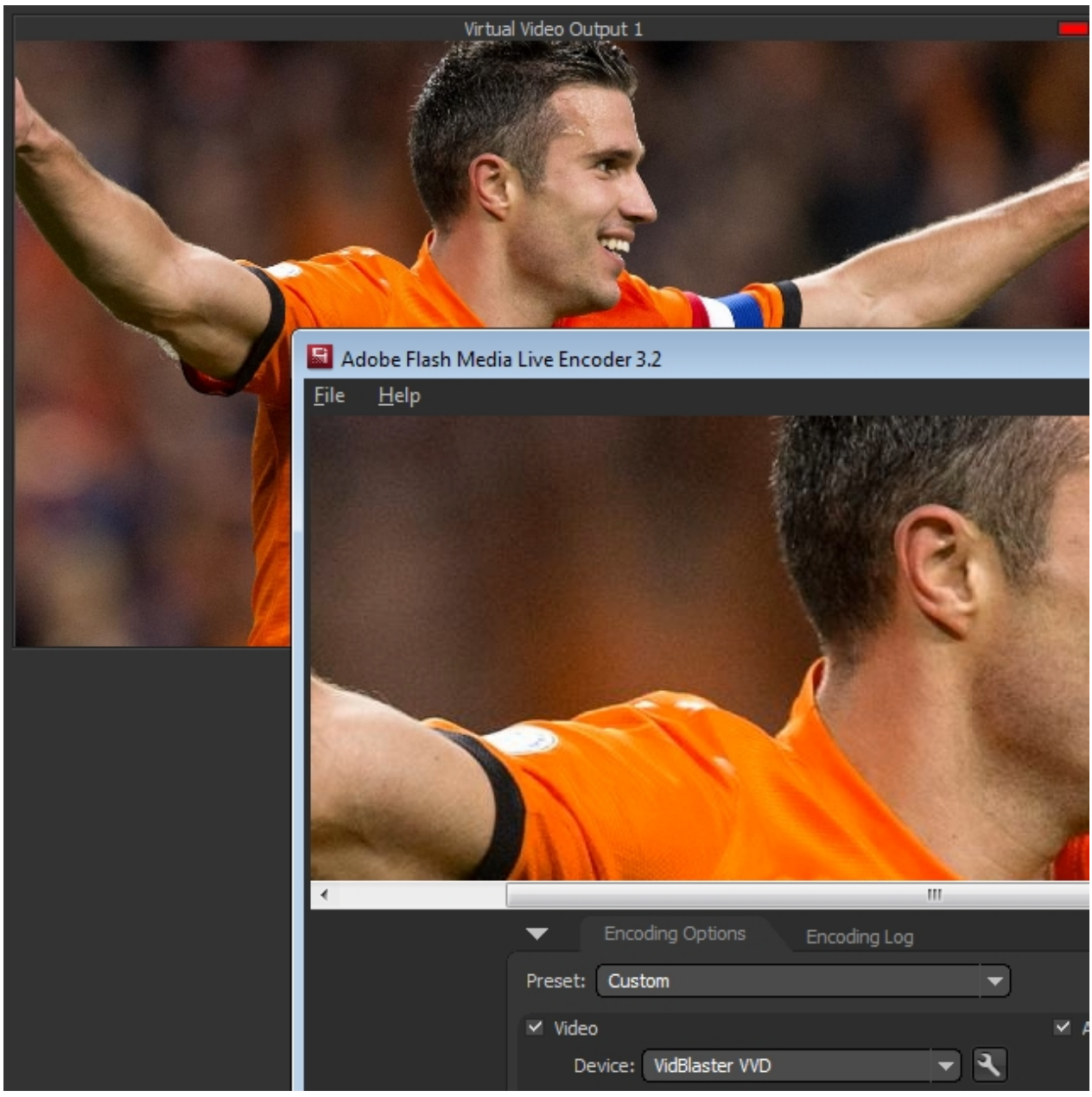
[1] Available upon request.

The NDI Output Module

The *NDI Output* module is used to make audio & video from any source available as an NDI source on the local network.

The Virtual Video Output Module

The *Virtual Video Output*[1] module is used to send video to a virtual video device (the *VidBlaster VVD*) that another application can use as video input. This enables you to send video to other applications.



Note that currently only one *Virtual Video Output* module can be loaded.

[1] Available upon request.

Replays, Slow Motion & Scoring

Video replays are often used in sport matches, where e..g. video from a camera can be played again at variable speed. An advanced replay/slomo system can be built in VidBlaster using one or more Slomo modules. Various controllers like the JLCooper Slomo Elite C, Skaarhoj XC8, Behringer CMD PL-1 and Studio 2A, and the Contour Design ShuttlePRO can be used to operate this system, but software control using macros or your own application and the API is also a available. The Slomo module can also be used standalone for one-click instant replay, as a preset delayer for audio and/or video (seven-second delay or profanity delay) or for time-shifting (tape delay or west-coast delay).

There are several ways a scoreboard can be implemented in VidBlasterX. You can use a third party scoreboard program and use screen capture, the API or chroma keying to get its output into VidBlasterX's video stream. By far the easiest way however is to use the *Scoreboard* module, which can either be keyed over any video stream using an effects bus, or used as a downstream keyer and keyed directly over the program bus.

The Replay Module

In its default form the `Replay` module can be used as a broadcast delayer or for basic replay. Controlled through the API or via an interfacing module like `JLCooper I/O` it will change into the most complex and powerful module available with its multi-input recorder, multiple players with slow motion functionality and built-in video and audio mixers.

Right click the module to open its popup menu. The following additional menu entries are available.

Fixed Delay
When using this module for e.g. basic replay or broadcast delay you can set the desired delay in frames here.

Record Drive
Select a drive for the record buffer, preferably a dedicated NVMe or USB 3.1 SSD.

Record Buffer Size
Select the percentage of the record buffer drive to be used for recording. The record buffer is circular, so

when it's full the oldest timecodes will be overwritten by the new ones. Also displays the corresponding recording time.

Clip Export Folder
Select a folder to store compressed video clips. Note saved clips are kept in the record buffer, the folder selected here is for optional (compressed) export of clips for e.g. publishing on social media. By default this points to Windows' Videos folder. The file name is automatically created including a timestamp (VidBlasterX_yymmddhhmmss.mp4, e.g. VidBlasterX_230805163944.mp4).

Actions > On Program > Replay
One-click instant replay. When selected a replay at half speed off the last 10 s will start when the module is selected for program output. The video below shows this feature combined with the play action and a touch screen.

Properties > Alpha
When *Premultiplied* is selected the recorded video will include the alpha channel. Note this increases disk requirements for bandwidth and capacity by 67%.

The JLCooper I/O Module

The `JLCooper I/O` module[1] acts as a gateway between the Elite C controller and the modules required to setup a professional slomo/replay system that will feel familiar to most professional slomo operators. Right click the module to open its popup menu. The following additional menu entries are available.

Off
Switches all connected Slomo modules off. Note SSD drives can handle a limited amount of write cycles so always switch off your replay system (or use this option) to reduce wear of your drives.

Bus
Bus dedicated for program/preview use. Typically a MIX bus will be used for this, but on dedicated systems the Program bus can be used as well. Add a MIX bus to your profile and select it here. The bus will automatically be programmed and can be ignored and (optionally) minimised.

Guardband
When possible, clips are saved with an additional amount of recording before and after the intended clip. The amount of seconds of this guardband is set here.

Playlist
Playlist module used to create playlists.

Port
Opens a dialog to set the TCP port (default 9980).

[1] Available in Broadcast edition only

JLCooper Slomo Elite C controller

This controller requires the JLCooper I/O module to be loaded. Connect the controller to your network through its ethernet port. To configure the controller, power it on while pressing down both the `F1` and `C4` buttons. Make sure to set it to `Client` mode. (Instructions how to setup a direct connection between your controller and PC is given below.) Finish by pressing the controller's `Enter` button. It should now connect to VidBlasterX.

Controller buttons

`1 2 3 4`
Select "camera angle", i.e. which Slomo module is used for program or preview video.

`1 2 3 4 5 6 7 8 9 10`
Digit keys, used to store or recall a clip or enter a bank number (first press *Shift*).

`C1`
Opens the playlists menu. In the playlists menu this button selects playlist 1. In the playlist menu this button exits to the main menu. In the options menu this stops (pauses) recording.

`C2`
In the playlists menu this button selects playlist 2. In the options menu this starts recording.

`C3`
In the playlists menu this selects playlist 3. In the playlist menu this selects the previous clip. In the options menu this resets both replay module and controller.

`C4`
Selects the previous bank of clips in both the main and playlist menus. In the playlists menu this button selects playlist 4.

`C5`
In the playlists menu this button selects playlist 5. In the playlist menu this loads the playlist. In the options menu this exports the currently loaded clip, often used for posting clips on social media.

`C6`
In the playlists menu this selects playlist 6.

`C7`
In the playlists menu this selects playlist 7. In the playlist menu it selects the next clip.

`C8`
Selects the next bank of clips in both the main and playlist menus. In the playlists menu it selects playlist 8.

`Clear`
Used to clear items like in & out points, clips and playlists. Activate this function by pressing once, the light will turn red, then press the item to clear.

`IN`
Marks in point (again). Use `Shift` > `IN` to go to the in point. Use `Clear` > `IN` to clear the in point.

`LIVE`
Returns all channels to live video. Any in & out points will be cleared, use `Shift` > `LIVE` to skip the clearing.

`OUT`
Marks out point (again). Use `Shift` > `OUT` to go to the out point. Use `Clear` > `OUT` to clear the out point.

`PLAY`
Plays from current position at 100% speed.

`PVW`
Select preview mode. `Clear` > `PVW` will sync the preview timecode to program timecode.

`Shift`
Used to access secondary commands like go to in & out points. Activate this function by pressing once, the light will turn green and automatically clear with the next button press.

`TAKE`
Performs a take. Type and length of the transition is set in the replay module.

`X2`
Select jog speed multiplier (1, 2, 4, 0.5).

Colour codes
Button colour codes for the large buttons are yellow (available), red (active) or off (inactive). Due to the controller's limitations the smaller buttons have a slightly different colour scheme: green (available), red (active) or off (not active).

Storing & trimming a clip
Jog to the start of the intended clip and press `IN`. Jog to the end of the intended clip and press `OUT`. Find a free clip number (optionally select a different clip bank) and press the corresponding digit key. The digit key will now turn green indicating it is occupied. To recall a clip find one and press the corresponding key, which will turn red indicating the clip is loaded. You can trim the clips by changing its in & outs. To change an in or out point jog to its new position and press `IN` or `OUT`.

Adding a clip to a playlist
In its simplest form adding a clip to a playlist requires just 3 button presses: `C1` to open playlist mode, the playlist number and the clip to add from the current bank. If you want to insert a clip first use the buttons `C3` and/or `C7` to mark the position where the new clip will be inserted. You can also remove clips from the playlist using `Clear`.

Clearing a playlist
From the main menu press `C1` to open playlist mode, `CLEAR` and the desired playlist.

Exporting a clip
Optionally select the desired clip bank and load a clip by pressing the corresponding digit key. Press the `Option` key and then from the main menu `C5` to export the clip. Press `Option` again to go back to the main menu.

Creating a direct cable connection between controller and PC
Typically the controller will be connected to a network for local, or to the internet for remote productions. For small productions or testing it may be preferable to connect the controller directly to the VidBlasterX PC with just an ethernet cable. If you have no experience in this, the list of instructions below may be used as a guide.

1. The default setting for an ethernet port is DHCP. If there is no DCHP server offering you an IP, a random self assigned IP in the 169.254.x.x range will be used. This is not very suitable in our case, as we want full control of the IP in order to be able to tell the controller where to connect to. Best thing to do here is to assign a static IP address for the port in the 192.168.x.x or 10.x.x.x range, e.g. 192.168.1.10.

2. Either disable Windows firewall for the ethernet port or create a firewall rule allowing communication with the controller.

3. Set the controller to *Function* as a *Client*.

4. Set the controller's *DESTIP Address* and *DEST Port Numb* to the IP address and port number of the ethernet port (this is displayed in the JLCooper I/O module).

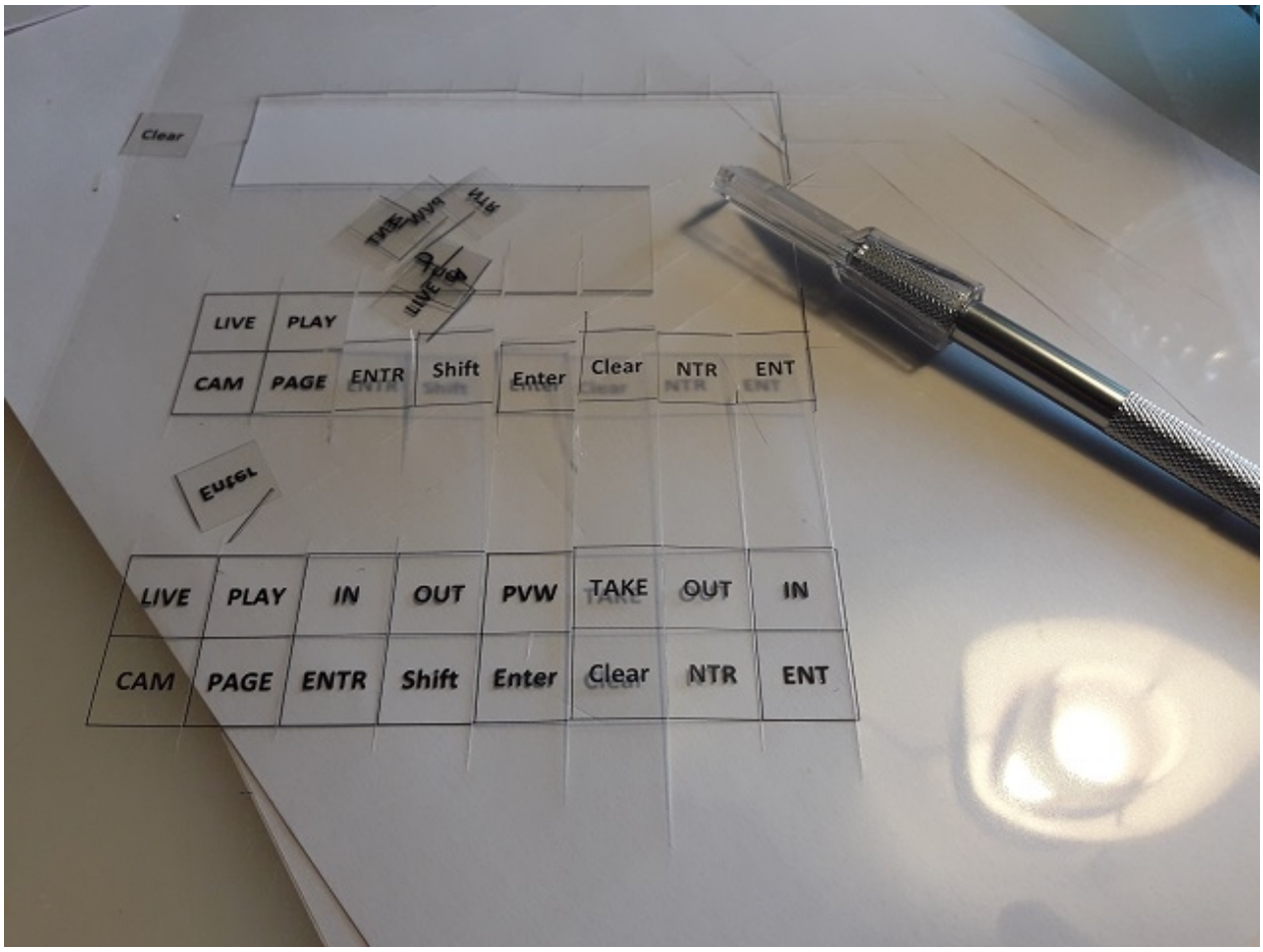
.

5. The IP of the computer and the controller need to be different, but in the same subnet. So for the controller's *IP Address* enter e.g. 192.168.1.11.

6. Change the subnet mask on the controller to 255.255.255.000.

Controller buttons
Upon request JLCooper Electronics will offer a VidBlaster edition of this controller but you can also quite easily recap some of the controller's buttons yourself using this (Word) template, transparent (overhead projector) sheets and a laser printer. As the horizontal row of player buttons has etched caps they cannot

be used. Instead use the spare set of button caps you received with the controller (if you did not receive them JLCooper will ship them to you free of charge). The controller's firmware is also checked (currently v2.28 is recommended) and you'll receive a notification if it needs to be replaced. You can download the firmware and the uploader utility from here.



The Skaarhoj I/O Module

The `Skaarhoj I/O`[1] module acts as a gateway between the XC8 controller and the modules required to

setup a professional slomo/replay system. Right click the module to open its popup menu. The following additional menu entries are available.

Off
Switches all connected *Slomo* modules off. Note SSD drives can handle a limited amount of write cycles so always switch off your replay system (or use this option) when not used to reduce wear of your drives.

Bus
Optionally select the switcher bus to allow multiplexing of and transition between camera angles. Typically a MIX bus will be used for this, but on dedicated systems the Program bus can be used as well. Add a MIX bus to your profile and select it here. The bus will be automatically be programmed and can be ignored and (optionally) minimised.

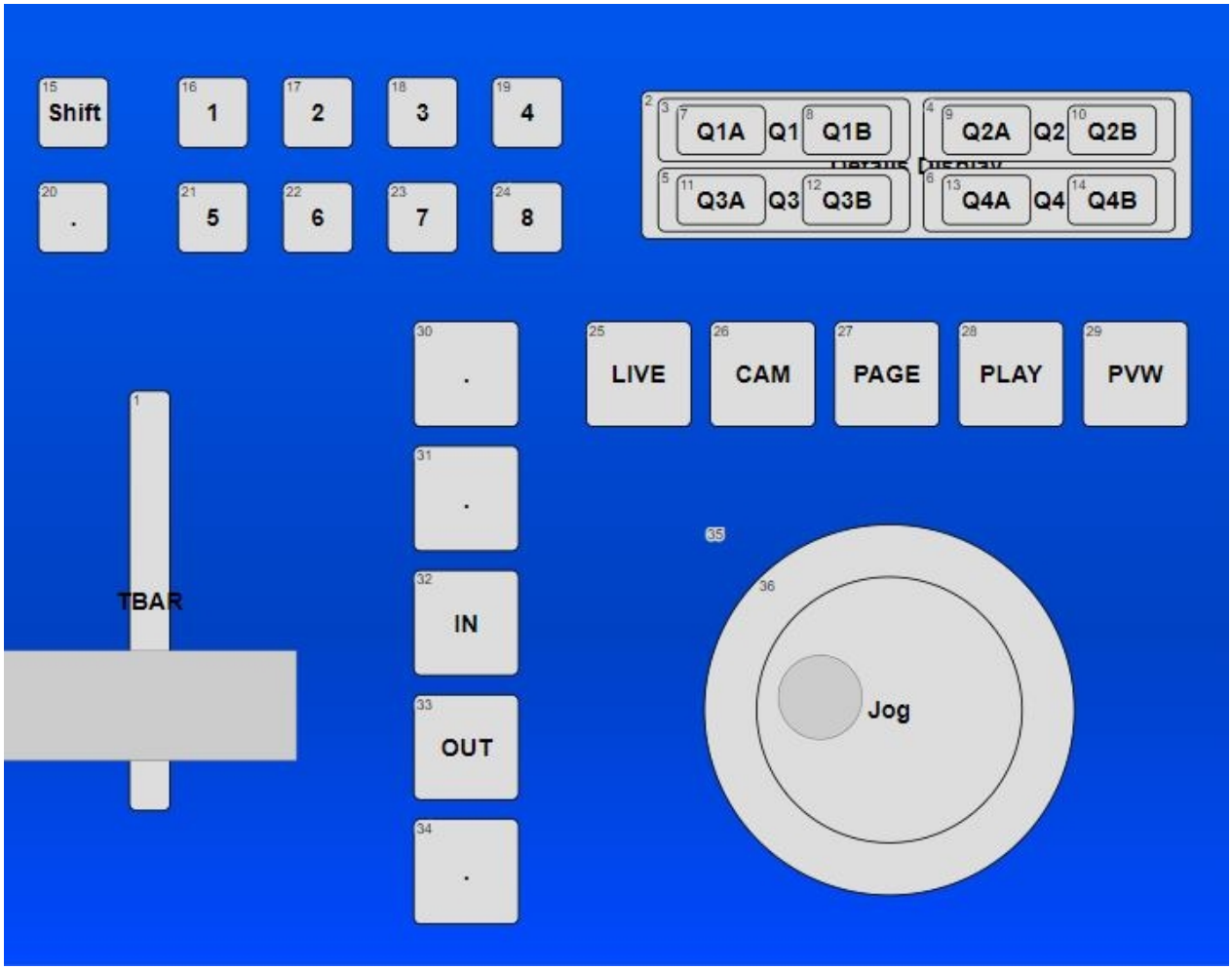
Guardband
When possible, clips are saved with an additional amount of recording before and after the intended clip. The amount of seconds of this guardband is set here.

Mode
Select the controller mode, i.e. the number of output channels per camera angle and the manner in which the XC8 controls them.

[1] Available upon request.

Skaarhoj XC8 controller

This controller requires the *Skaarhoj I/O* module to be loaded. Connect the controller to your network through its ethernet port and follow the "Raw Panel" instructions that come with the controller. Once properly configured VidBlasterX will automatically detect the controller and initialise it.



Until Skaaroj starts offering a VidBlasterX edition of this controller you may want to cap some of the controller's buttons. This can be done quite easily using this (Word) template, transparent (overhead projector) sheets and a laser printer.

Here's a brief overview of the buttons' functionality in alphabetical order.

1 2 3 4 5 6 7 8
Loads clip from in selected memory bank if available, stores current clip (if in & out points set) if memory empty. Also used for memory bank selection (press *Shift* first), page & bank selection (press *PAGE* first) and camera angle selection (press *CAM* first).

CAM
Press this button followed by *1..8* to select the "camera angle", i.e. which Slomo module is used for program or preview video.

Clear
Used to clear settings like in & out points.

IN
Marks in point. Use Shift > *IN* to go to in point. Use Clear > *IN* to clear the in point.

LIVE
Returns to live video. Any in & out points will be cleared, use Shift > *LIVE* to skip this step.

OUT
Marks out point. Use Shift > *OUT* to go to out point. Use Clear > *OUT* to clear the out point.

PAGE
Page followed by bank selection for clips.

PLAY
Plays from current position at 100% speed.

Shift
Used to access secondary commands like go to in & out points and clip bank selection.

Button colour codes are yellow (available) and red (active). The digit buttons use green for PVW and red for PGM during camera selection, and white/off in clips mode.

The Behringer I/O Module

The `Behringer I/O`[1] module acts as a gateway between the Behringer CMD PL-1 or Studio 2A controller and one or more Slomo modules to create a basic slomo/replay system on a budget. Note the controller needs to be connected before this module is loaded.

[1] Available upon request.

Behringer CMD PL-1 controller

VidBlaster has built-in native support for the Behringer CMD PL-1 and Studio 2A controllers. Simply connect it before you run VidBlaster suffices, no further installations are required.

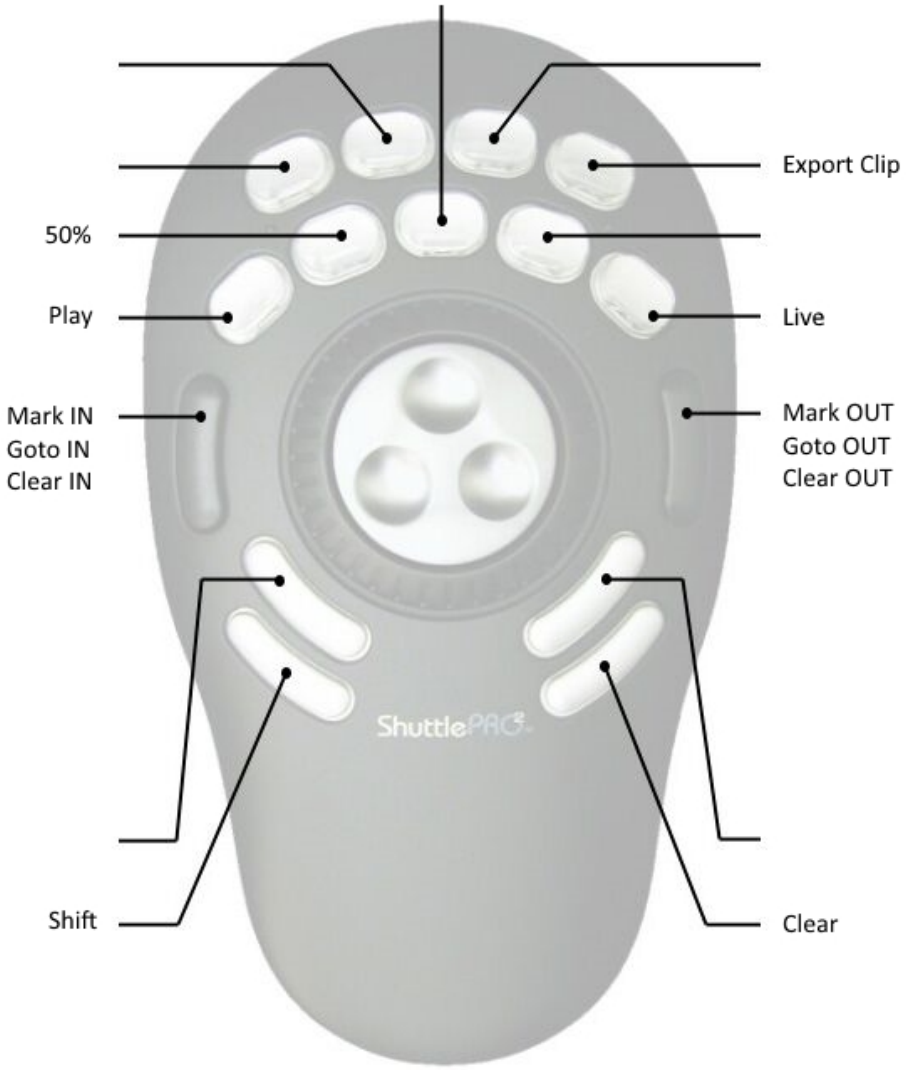
The ContourDesign I/O module

The `ContourDesign I/O`[1] module acts as a gateway between the ShuttlePRO v2 controller and one or more Slomo modules to create a budget slomo/replay system.

[1] Available upon request.

Contour Design ShuttlePRO v2 controller

VidBlaster has built-in native support for the Contour Design ShuttlePRO v2 controller. Simply install it per the manufacturer's instructions and it will be automatically detected. If you wish to create labels for some of the controller's buttons you can use this (Word) template, a transparent (overhead projector) sheet and a laser printer.



Second and third entry apply when Shift resp. Clear button pressed.

Jog wheel
Using the jog wheel you can move forward and backward through the recording in single frame steps. Holding down *Shift* increases the step size by a factor of 10.

Shuttle ring
Use the shuttle ring to move forward and backward through the recording in (exponentially) larger steps. Holding down *Shift* increases the step size by a factor of 10.

Play

Plays from current position at 100% speed.

50%
Plays from current position at 50% speed.

Live
Returns to live mode, clearing in & out points.

Mark IN, Goto IN, Clear IN
Set/goto/clear the IN point.

Mark OUT, Goto OUT, Clear OUT
Set/goto/clear the OUT point.

Export Clip
Often used for posting clips on social media, this command exports the recording marked with IN & OUT as a compressed video file to the *Clip Export Folder* set in the Slomo module.

Shift
Used to access alternate commands like go to in & out points and jogging with 1 s steps. Hold the button down to access these commands.

Clear
Used to access alternate commands like clearing in & out points. Hold the button down to access these commands.

NOTE: the current driver for the Shuttle controllers is old and imperfect. Contour Design is planning to develop a new driver but no date has been set.

The JLCooper Emulator Module

The `JLCooper Emulator` module is an emulator of the JLClooper Elite C controller that operates and connects through the `JLCooper I/O` module just like the real controller. At no additional hardware costs, this module can be used to create a professional slomo/replay system that will feel familiar to most professional slomo operators. It can also be used as a training tool or just to trial the controller before purchasing it.

The Scoreboard Module

The *Scoreboard*[1] module acts both as a still store for a scoreboard graphic, and as a character generator for the team names and scores. Edit boxes are available to (optionally) enter the name of the teams and their scores. Right click the module to open its popup menu. The following additional menu entries are available.

On
Switch overlay on.

Off
Switch overlay off.

Open
Shows the *Open* dialog to load a scoreboard graphic. Instead of using this command, you can also drag & drop a file on the module.

Font
Displays Font dialog to select font properties.

Position Scoreboard
Allows you to position the scoreboard graphic using x and y coordinates, offset is in pixels to top-left corner.

Position Team 1

Allows you to position the name of team 1 using x and y coordinates, offset is in pixels to top-left corner.

Position Team 2
Allows you to position the name of team 2 using x and y coordinates, offset is in pixels to top-left corner.

Position Team 1 Score
Allows you to position the score of team 1 using x and y coordinates, offset is in pixels to top-left corner.

Position Team 2 Score
Allows you to position the score of team 2 using x and y coordinates, offset is in pixels to top-left corner.

Option > DSK
By default the Scoreboard module acts as a source, which will typically be used in an FX bus. By checking this flag the module will act as a downstream keyer, directly keying over the program bus.

[1] Available upon request.

The Timer Module

The *Timer* module places an anti-aliased timer on a transparent background, that can then be keyed over other sources using the Effect bus. Right click the module to open its popup menu. The following additional menu entries are available.

Start
Resumes counting.

Reset
Stops the timer and resets it.

Stop
Pauses counting.

Font
Displays *Font* dialog to select font properties.

Video Switching, Mixing & Effects

Video switching, mixing, video effects like keying and tally support are handled by the switcher modules. The switcher modules can be used separately, or they can be docked to form one or more traditional video switchers. The switcher modules are very powerful yet easy to use. Despite their slightly different appearance and naming convention, they work almost the same as a professional video switcher in a television control room. The switcher modules can be controlled by mouse or, using macros, by keyboard or an external controller.

Switcher modules are available in four different flavours, each called a bus: the *Auxiliary* (*AUX*), *Effects* (*FX*), *Mix* (*MIX*) and *Program* (*PGM*) bus. The auxiliary bus is a single bus with radio buttons (i.e. only one source can be selected at any time) typically used to drive studio monitors. The effects bus can be used both as single or dual (A/B) bus. For every source an effect can be selected. Multiple sources can be active simultaneously, they are overlaid from left to right. The mix bus is similar to the auxiliary bus but with optional A/B function. Typically used as general purpose bus. Finally the program bus is also similar to the auxiliary bus but with optional A/B function, where the B bus will be named preview (PVW). This bus is typically used for the video stream to be broadcast, streamed or recorded. The program & preview buses also directly control the status of the source modules' tally lights. Each bus has a single video output. Buses share a lot of common functionality, and in fact you'll find that quite often more than one bus can be used for a certain task.

Right-click a button to open its popup menu. The following additional menu entries are available.

Crop, Position & Scale
The *Crop, Position & Scale* effect allows you to accurately crop, position and scale a source. Changing any

of the values can be done by entering the desired value (%), use the up/down buttons, or click and drag the *Crop*, *Position* or *Scale* buttons. An additional zoom function, which changes both the scale and position simultaneously, is available through the mouse scroll wheel. Hold down the Shift key to improve accuracy when using your mouse. To reset a value to its default double click it while holding down the Shift key.

Keying > Disabled/Embedded Alpha/Chroma
The FX bus supports keying using either the source's alpha or chroma channel. Note sources are overlaid from left to right, starting with the background.

Chroma Key > Auto Key
When chroma keying is selected, this option will scan the current video frame and compute it's key (average colour of the green or blue screen).

Video Source
You can change the source for the selected bus button here. Except for the source and button caption, all settings will remain unchanged.

Remove Source
Remove the selected source. A source can also be removed by dragging it away from the bus in vertical direction.

Bus Add Source
Add one or all sources to the bus. Note the order of sources can be changed by dragging the buttons to the desired position.

Bus Background
The video source selected here will act as background. Set background to *Black* (default) is you wish to select the background video source on the bus itself. Available for the effects bus only.

Bus Option > Alias
Opens a dialog allowing you to enter the FX bus's *Alias*, which can be a more descriptive version of the bus. Available for the effects bus only.

Bus Option > A/B Mode
Adds the B bus, to be used in combination with the transition panel. Allows for transitions. Not available for the auxiliary bus.

Bus Option > Tally
When set this bus directly updates source modules' tallies. By default this flag is only set for the program and preview buses.

Related videos

The Transition Panel

The transition panel is available for A/B buses and can be used to perform transitions, e.g. a dissolve, from A to B bus.

AUTO
Performs the selected transition.

TAKE
Performs a hard cut.

FTB
Shortcut to selecting the *Fade to Black* transition and clicking the *AUTO* button.

The Tally Lights Module

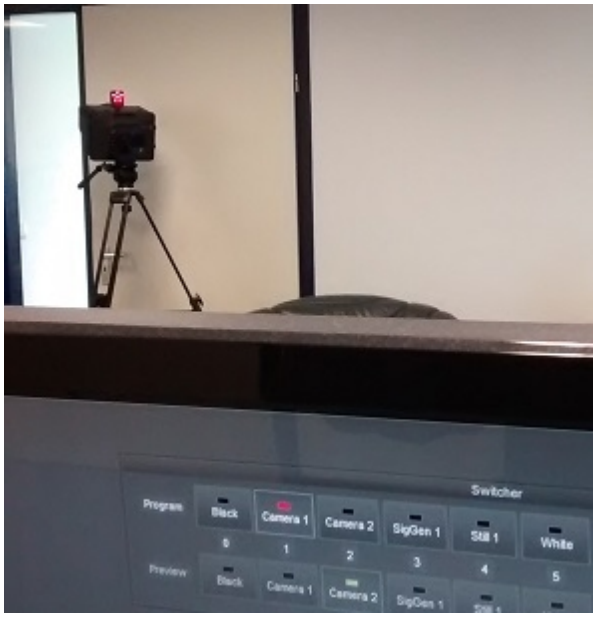
The *Tally Lights*[1] module can be added from the menu by selecting *Modules* | *Add* | *Tally Lights*. It currently supports the Tally-Lights LLC system, the Velleman K8090 as well as a generic driver for DIY projects. Installed tally lights software or hardware will be automatically detected when the module is loaded.

Note this module will take tally information from the *PGM 1* bus.

Right click the module to get access to its popup menu.

Remove Module
Removes the module.

[1] Available upon request.

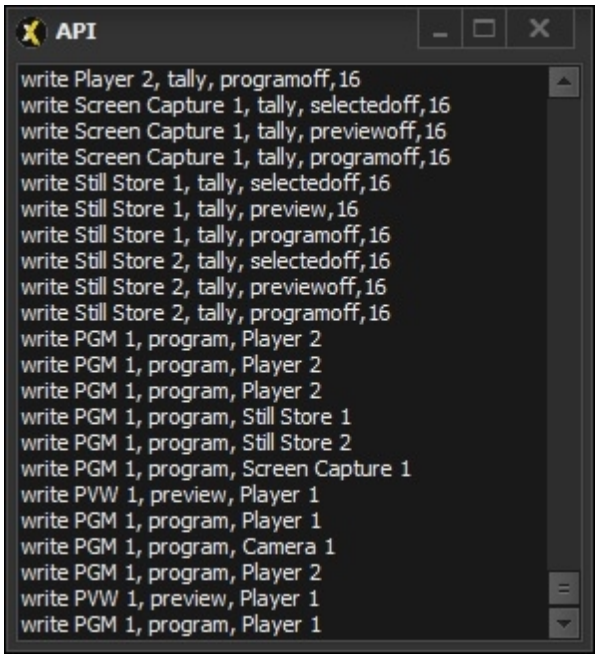


DIY Tally System

For DIY tally lights projects, or commercial systems that are not natively supported, the *Tally Lights* module has built-in support for a generic driver. The driver is automatically detected when installed in the following location off the VidBlasterX folder: *Plugins\Tally\Generic\Tally.exe*. Typically this will be *C:\Program Files\CombiTech\VidBlasterX\Plugins\Tally\Generic\Tally.exe*. The driver, *Tally.exe*, will be called for every switcher status change with an 8 digit argument, where each digit can be 0 (off), 1 (preview), 2 (program) or 3 (preview & program). E.g. an argument *20000021* means cameras 2 & 8 are selected for program and camera 1 for preview.

API

VidBlasterX can be (remote) controlled through its API (Application Programming Interface). The API is a powerful feature enabling you to create complex series of commands using macros, or to operate the software from any application anywhere in the world using the *TCP Server*.



From an API point of view, VidBlasterX is a collection of modules where each module can have one or more pins. A pin can be input and/or output and exchange data in various formats. The following API commands are available to get information about the API, obtain a list of available modules and access each module:

apiabout

apilist

apilist2

apiread

apiwrite

When a command is recognized, *200 Ok* will be returned, optionally followed by the requested value or an error message.

Almost all modules have pins that can be accessed through the API, a list of available pins for each module can be selected from the table of contents. Pins that are common to most or all modules are listed below and not repeated for each individual module.

Pin

`alias, <alias 1>[, <alias 2>...]`
: Sets or returns the alias(es) of a module, one for each output channel.

`audiochannels`
: Returns the number of audio channels.

`audioinput`
: Sets audio input.

`audiosource [, <source>]`
: Sets or returns the audio source.

`tally`
: Used to read and set the tally status. When setting the tally state is specified in the value parameter. Available states are *off*, *preview*, *previewoff*, *program, programoff*, *selected* & *selected*

off. An optional second value parameter (default is 0) can be added which is a zero-based index to the bus sending the command, enabling multiple buses to drive the same tally. A reference count will be kept and only when zero will a tally go in the off-state. Note that when dealing with single tallies, program state will always take precedence over preview state. When reading the possible return values are an empty string, "preview", "program" & "selected".

`videosource [, <source>]`
Sets or returns the video source.

Examples

The following example sets Camera 1's tally to program.

```
apiwrite Camera 1, tally, program
```

The following example starts all Players.

```
apiwrite Player 0, play, true
```

The following example sets Player 1's tally to preview. The command was sent from the second bus.

```
apiwrite Player 1, tally, preview, 1
```

The TCP Server Module

The *TCP Server* module[1] consists of two servers. The command server enables other applications to access the API from anywhere in the world, sending instructions to the program and retrieving status information. The event server sends event messages for several triggers. In addition to the generic events listed below, events that relate to a specific module are documented per module.

`onalias, <module>, <alias>`
Triggered when a module's alias changes.

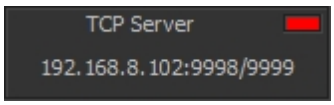
`onaudiooutput, <module>, <channels>`
Triggered when a module's audio output format changes, i.e. a change in the number of audio channels.

`onaudiosource, <module>, <source>`
Triggered when a module's audio source changes.

`onmodules, <modules>`
Triggered when a module is added or removed from the current profile. Includes a comma-separated list of all modules in the current profile.

`ontally, <module>, <tally>`
Triggered when a <module>'s tally changes status.

The module displays the local IP address and port numbers through which it can be accessed. A red tally will light up when a client connects to the server. Right click the module to open its popup menu. The following additional menu entry is available.



Port Number
Set the port number.

[1] Available in Broadcast edition only

API Command apilist

Returns a list of the titles of all modules that are currently loaded or, with optional module parameter, returns a list of the module's pins.

```
apilist [module]
```

Parameters

module
Optional. Name of module or (internal) function.

Example

The following example returns a list of all pins available from Player 1.

```
apilist player 1
```

API Command apiabout

Returns the application's edition & version number.

```
apiabout
```

API Command apilist2

Returns a list of the titles of all modules that are currently loaded and their type, comma separated. Type is a constant and therefore in all capitals, i.e. AUDIO, CAMERA, PLAYER.

```
apilist2
```

API Command apiread

Reads the value of a module's pin.

```
apiread module, pin
```

Parameters

module
Name of module or (internal) function.

pin
Name of pin. Pin names are case insensitive.

Example

The following example returns the 0-based index number of the Switcher's active program bus button.

```
apiread switcher, programbus
```

API Command apiwrite

Writes data to a module.

```
apiwrite module, pin, value
```

Parameters

module
Name of module or (internal) function. This is usually the module's name and index, e.g. *Camera 1*. To address all modules of the same type at once use an index of 0, e.g. *Camera 0*.

pin
Name of pin. Pin names are case insensitive.

value
Value(s) to be written. The type of value (e.g. text, integer, boolean) depends on the pin. Multiple values are separated by commas.

Remarks

The apiwrite function writes a value to a module's pin. If this is a text and you wish to use commas, leading and/or trailing spaces then use escaping. Escaping (\xhh, where hh is the hexadecimal value of the character) is supported to embed special characters in a text string (like commas and double quotes). Note some apiwrite commands address internal functions, here the module parameter is used to indicate the internal function (e.g. macro).

Example

The following example sets the name of team 1 in the module Scoreboard 1.

```
apiwrite scoreboard 1, team1, Chicago "Bulls"
```

Audio Mixer Pins

Pin

input, <source>
Adds input channel strip for <source> to the audio mixer.

volume, <source>|<caption>, <dBFS>|<percentage>
Writes the volume fader setting for channel strip with either <source> or <caption>, which can be the name of the source or caption of any fader including *master* and *monitor*. Value is in dBFS or in percentage (must end with percentage sign).
Returns the volume fader setting for channel strip with either <source> or <caption> in dBFS.

Example

The following example sets the fader of the Microphone input to -30 dBFS.

```
apiwrite audio mixer 1, volume, microphone, -30
```

The following example sets the fader with Mic caption halfway.

```
apiwrite audio mixer 1, volume, Mic, 50%
```

The following example returns the Player 1 fader position in dBFS.

```
apiread audio mixer 1, volume, Player 1
```

Camera Pins

Pin

videodevice, device[1]
Reads or writes the video device. When writing the value parameter is a text string.

Example

The following example selects the (first) Decklink Video Capture device for Camera 1.

```
apiwrite camera 1, videodevice, Decklink Video Capture
```

[1] Deprecated, supported for a limited time only for backwards compatibility

DeckLink Output Pins

Pin

off
Executes the off command. Value parameter is ignored.

on
Executes the on command. Value parameter is ignored.

Example

The following example selects Camera 1 as video source.

```

```

```
apiwrite decklink output 1, videosource, Camera 1
```

IP Input Pins

Pin

off
: Executes the off command, identical to pressing the *Off* button. Value parameter is ignored.

on
: Executes the on command, identical to pressing the *On* button. Value parameter is ignored.

Example

The following example starts decoding of the current stream.

```
apiwrite ip input 1, start, true
```

Macro Pins

Pin

execute
: Executes the macro. Value parameter is ignored.

Example

The following example executes macro #1.

```
apiwrite macro 1, execute, true
```

NDI Input Pins

Pin

`sourcetally`
: Returns source tally status. Return can be empty or the word *preview* or *program*.

Event

`onsourcetally, <module>, <tally>`
: Triggered when the <module>'s source tally changes status. Parameter <tally> can be empty or the word *preview* or *program*.

NDI Output Pins

Pin

off
: Executes the off command. Value parameter is ignored.

on
: Executes the on command. Value parameter is ignored.

Example

The following example selects Camera 1 as video source.

```
apiwrite output 1, videosource, Camera 1
```

Player Pins

Pin

duration
Returns the duration of the file in milliseconds.

elapsed
Returns the elapsed time in milliseconds.

file, <filename>
Writes or reads the name of the video file. When writing the value parameter is text string with name of the file.

pause
Pauses the player. Value parameter is ignored.

play
Starts the player. Value parameter is ignored.

playing
Reads the playing status, returns true or false.

position, <pos>
Sets position in either milliseconds or percent. Value parameters is numerical string (ms) with optional percentage sign (%).

stop
Stops the player.

Event

onfile, <module>, <filename>, <duration>, <audio-only>
Triggered when a file is loaded. Parameters include the module name, file name, file duration and audio-only flag.

Example

The following example starts Player 1.

```
apiwrite player 1, play, true
```

The following example sets Player 1's position to halfway.

```
apiwrite player 1, position, 50%
```

Playout Pins & Events

Pin

additem video|audio|AV, <clip>, <angle>, [<speed>, [<inoffset>, [<outoffset>, [<transition>, [<description>]]]]]
Adds video, audio or both as item to current playlist. Default speed is 100%, default offsets 0, default transition hard cut, default description empty. Playlists can be updated during playout, but current and upcoming clip should not be changed.

clearlist
Clears playlist.

loadlist <filename>
Loads playlist <filename>.

play
Start playout.

playing
Returns name of the clip currently playing, or empty string if not playing.

savelist <filename>
Saves current playlist to <filename>.

speed <percent>
Sets overall playout speed to <percent>. The speed set here is multiplied by the speed setting of each item in the playlist. Default is 100%.

stop
Pauses playout. If playout was already paused this will reset and pause playout at the first item.

timecode, <timecode>
Write to this pin to change the <timecode>. Timecode is considered absolute unless it is preceded by a plus or minus sign in which case it is relative to the current timecode.
Read from this pin to get the current timecode.

Event

onerror, <module>, <message>
Triggered when an error occurs loading a playlist.

Example

The following example adds the video part of clip #123 to the playlist with all default settings.

```
apiwrite playout, additem, video, 123
```

Powerpoint Pins

Pin

file
Reads or writes the file name. When writing the value parameter is a text string with the name of the file which is immediately loaded.

next

Selects next slide, identical to clicking the *Next* button. Value parameter is ignored.

prev
Selects previous slide, identical to clicking the *Prev* button. Value parameter is ignored.

Example

The following example selects the next slide in module Powerpoint 1.

```
apiwrite powerpoint 1, next, true
```

Recorder Pins

Pin

record
Starts recording, identical to clicking the *Record* button. Value parameter is ignored.

stop
Stops recording, identical to clicking the *Stop* button. Value parameter is ignored.

time
Returns the elapsed recording time in seconds.

Example

The following example starts Recorder 1.

```
apiwrite recorder 1, record, true
```

[1] Deprecated, supported for a limited time only for backwards compatibility

Scoreboard Pins

Pin

file
Loads graphic file, value parameter is text string with name of the file.

off
Executes the off command, identical to pressing the *Off* button. Value parameter is ignored.

on
Executes the on command, identical to pressing the *On* button. Value parameter is ignored.

scoreteam1
Sets the score for team 1, value parameter is text string with score.

scoreteam1x
Sets the x coordinate of the score for team 1, value parameter is text string with coordinate.

scoreteam1y
Sets the y coordinate of the score for team 1, value parameter is text string with coordinate.

scoreteam2
Sets the score of team 2, value parameter is text string with score.

scoreteam2x
Sets the x coordinate of the score for team 2, value parameter is text string with coordinate.

scoreteam2y
Sets the y coordinate of the score for team 2, value parameter is text string with coordinate.

team1
Sets the name of team 1, value parameter is text string with name.

team1x
Sets the x coordinate of the name for team 1, value parameter is text string with coordinate.

team1y
Sets the y coordinate of the name for team 1, value parameter is text string with coordinate.

team2
Sets the name of team 2, value parameter is text string with name.

team2x
Sets the x coordinate of the name for team 2, value parameter is text string with coordinate.

team2y
Sets the y coordinate of the name for team 2, value parameter is text string with coordinate.

Example

The following example sets the score of team 1 to 0.

```
apiwrite scoreboard 1, scoreteam1, 0
```

Replay Pins & Events

Pin

`addtoplaylist, <playlist>, <pos>, <clip>`
Adds <clip> to <playlist> at <pos>.

`channel, <channel>, <input>`
Selects <input> for <channel>. E.g. **`apiwrite replay 1, channel, 1, 1`** will assign input 1 to channel 1 (typically PGM).
Returns input for <channel>.

`channels, <count>`
Set the <count> number of output channels (currently limited to 2), value parameter is text string with number of channels. The default setting is 1. Change this pin before writing to other pins.
Returns the number of output channels when read.

`clip <channel>`
Returns the name of the clip currently loaded, or empty string if none.

`clips [first, last]`
Returns a comma-separated list of available clips for this module (aka camera angle). Optionally the clip range can be limited from <first> to <last> (1-based).

`deleteclip, <name>`
Deletes video clip. Value parameter holds text string with clip <name>.

deleteplaylist, <playlist>
Clears contents of <playlist>.

deleteplaylistclip, <playlist>, <clip>
Removes <clip> from <playlist>.

exportclip <channel>
Saves recording/clip in <channel> from in to out point as compressed video file. All videos will be stored in the export folder. The file will be automatically named VidBlasterXSlomoClip_yymmddhhmmss_n, e.g. VidBlasterXSlomoClip_200818143000_1 for Slomo module 1 on August 18th 2020 at 2.30 PM.

fixeddelay <s>
Set fixed delay in seconds (0 is disabled).

guardband <time>
A guardband is the additional <time> of media saved, both before and after the intended clip. Value parameter holds text string with guardband in seconds.

in <channel>
Marks current timecode for <channel> as IN point if the value parameter is true, or erases the IN point if value is false.
Reads IN timecode for <channel>.

inputs
Returns number of inputs/cameras.

live <channel>
Enables/disables live mode for <channel>. Value parameter is "true" or "false".
Returns live mode status ("true" or "false") for <channel>.

loadclip <channel>, <index>
Loads video clip <index> into <channel> play buffer.

loadplaylist, <channel>, <index>
Loads playlist <index> into <channel> play buffer.

mode <text>
Read/write mode. Value parameter is <text> string. For internal use only.

off
Disables the module and stops internal recorder. Value parameter is ignored.

on
Enables the module and starts internal recorder. Value parameter is ignored.

osd <text>
Output OSD <text>. Value parameter is text string.

out <channel>
Marks current timecode for <channel> as out point if the value parameter is true, or erases the out point if value is false.
Read out timecode for <channel>.

playlistclips, <playlist>
Returns a comma-separated list of clips in <playlist>.

recording

Returns true if recording.

`recordpath`
Returns the full path to the record file.

`reset`
Resets all settings except the number of channels.

`saveclip <channel>, <index>`
Saves recording in <*channel*> from in to out point as video clip with <index> (0..9999).

`speed <channel>, <speed>`
Sets <speed> of <channel> at which frames are output. Fixed frame output is set with speed 0, (delayed) real time playback with speed 100. Value parameter is text string with percentage value.

`take`
Executes set transition.

`tally <channel>, <status>`
Sets tally <status> of <channel>. Value parameter is text string "Program", "Preview" or empty.

`timecode <channel>, <timecode>`
Write to this pin to change the <timecode> of <channel>. Timecode is considered absolute unless it is preceded by a plus or minus sign in which case it is relative to the channel's current timecode. Read from this pin to get the current timecode of <channel>. Returns 0 if no timecode available.

`timecodeframe <channel>`
Read from this pin to get the timecode of <frame> number. The frame number is considered an absolute value unless it is preceded by a plus or minus sign in which case it is considered an offset to the channel's current frame number. In this mode, which is typically used for jogging, the offset it automatically limited to avoid wrapping (which would result in the timecode jumping from newest to oldest or vice versa). Returns 0 if no timecode available.

`timecodelive`
Returns current live timecode in seconds.

`timecodelivehms`
Returns current live timecode formatted as hh:mm:ss.

Note by default 2 channels are available: 1 (PGM) and 2 (PVW). Where applicable to address both channels at the same time use 0.
Note timecode is a text representation of a double, where the integral part is unix time and the optional fractional part represents the fraction of the second. Decimal separator is a period.

Event

`onchannel, <module>, <channel>, <input>`
Triggered when the channel assignment has changed.

`onclipadded, <module>, <clip name>`
Triggered when a clip is created, includes the name of the clip.

`oncliploaded, <module>, <channel>, <clip name>`
Triggered when a clip is loaded. Use *onlive* event to detect clip unload.

`onclipremoved, <module>, <clip name>`
Triggered when a clip is deleted, includes the name of the clip.

`onexportclip, <module>, [true|false]`

Triggered when clip export started or stopped.

`oninputs, <module>, <inputs>`
Triggered when the number of inputs/cameras changes.

`onlive, <module>, <channel>, <live>`
Triggered when a channel's live status changes.

`onplay, <module>, <channel>, <speed>`
Triggered when a channel's play status changes.

Example

The following example sets program playback to half speed.

```
apiwrite slomo 1, speed 1, 50
```

The following example sets the timecode for channel 1 of all Slomo modules to January 1st, 2020 0:00 UTC.

```
apiwrite slomo 0, timecode 1, 1577836800
```

Still Store Pins

Pin

file
Writes or reads the name of the image file. When writing the value parameter is a text string with the name of the file.

grab
Grabs video frame from source, identical to pressing the *Grab* button. Value parameter is ignored.

Example

The following example selects Camera 1 as video source.

```
apiwrite still store 1, videosource, Camera 1
```

Streamer Pins

Pin

`start`
Starts streaming. Value parameter is ignored.

`stop`
Stops streaming. Value parameter is ignored.

`time`
Returns the elapsed stream time in seconds.

Example

The following example starts Streamer 1.

```
apiwrite streamer 1, start, true
```

Switcher Pins

Pin

`abmode`
: Enables (value = true) or disables (value = false) A/B mode.

`auto [<type>[, <ms>]]`
: Executes current transition. Optionally takes transition type and length parameters (see *transitionlength* & *transitiontype* pins). Note this command is only supported by A buses that are set in A/B mode.

`crop <source>, <x1>, <y1>, <x2>, <y2>`
: Returns/sets top-left (x1, y1) & bottom-right (x2, y2) crop of <source> in FX bus. These settings can also be found in the *Source Settings* dialog.

`deselect`
: Resets active sources in bus. Value parameter is a comma-separated list with source names or button captions (case-insensitive)

`croppposscale <source>, [true|false]`
: Returns/sets the enabled status of the crop/position/scale effect of <source> in the FX bus. This setting can also be found in the *Source Settings* dialog (*Enable*).

`optiontally [true|false]`
: Enables or disables (default) the tally option.

`position <source>, <x>, <y>`
: Returns/sets (x, y) position of <source> in FX bus. This setting can also be found in the *Source Settings* dialog (*Position*).

`ready`
: Returns *true* if switcher ready, i.e. fully constructed and all source buttons present.

`scale <source>, <x>, <y>`
: Returns/sets x & y scale of <source> in FX bus. This setting can also be found in the *Source Settings* dialog (*Scale*).

`select <source1>[, <source2>, ...]`
: Sets active sources in bus. Value parameter is a comma-separated list with source names or button captions (case-insensitive).

`selected`
: Returns comma-separated list with names of active sources, their alias and their 0-based bus index, e.g. *camera 1, CAM01, 0*.

`sources`
: When reading this returns a comma-separated list with names of all sources. When writing the bus is (re)built with sources supplied as a comma-separated list in the value parameter.

`take`
: Executes take. Note this command is only supported by A buses that are set in A/B mode.

`time`
: **Deprecated**: Reads or sets transition time in ms. Note this command is only supported by A buses that are set in A/B mode.

`toggle`
: Toggles source's active status. Value parameter is a source name or button caption (case-insensitive).

`transitionlength <ms>`
: Sets transition length in ms. Note this command is only supported by A buses that are set in A/B mode.

`transitionperc <perc>`
: Sets transition to <perc> [0..100].

`transitiontype <name>`
: Selects transition <type>.

Event

`onselected <module>, <source>, <alias>, <index>`
: Triggered when a bus source is (de)selected. Includes a comma-separated list of selected sources, their alias and bus index.

Example

The following example sets the transition time of the program bus to 1 s.

```
apiwrite PGM 1, time, 1000
```

The following example selects Camera 1 and Player 1 in FX 1.

```
apiwrite FX 1, select, "Camera 1,Player 1"
```

Timer Pins

Pin

`reset`
: Stops timer and sets it to its start value. Value parameter is ignored.

`start`
: Starts timer. Value parameter is ignored.

`stop`
: Stops timer. Value parameter is ignored.

Example

The following example starts Timer 1.

```
apiwrite timer 1, start, true
```

Diagnostics

If you ever find yourself in a situation where the result of your production is not as expected then there are several tools at your disposal to find out why.

If performance is an issue then the *Signal Generator* and *Diagnostics* modules are useful tools to test the PC and/or connected hardware, or simply monitor the program's internal status. They can also assist in finding out where delays or even frame drops are introduced.

To closely examine video signals both the Video diagnostic and Scopes module are at your disposal.

If the programs is not properly responding to user commands then the *Log* may provide additional information. If the program does not start at all, a profile reset (hold down left *Shift* key as the program starts) will clear the current profile and load a default profile instead.

To debug API commands, either from an internal or external source, the *API Command Stack* window may provide valuable information. Same goed for commands from external midi or x-keys input devices which can be monitored in the MIDI and X-keys event logs.

If you have a support contract further debug information can be send to the developer, most of this advanced functionality is hidden.

The Diagnostics Module

The Diagnostics module can be used to inspect system performance in both numerical and graphical presentations.

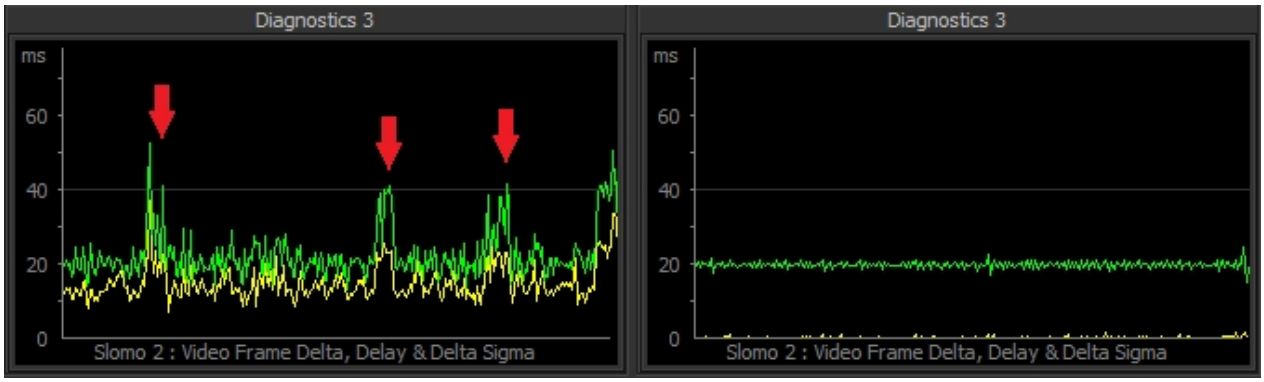
The following diagnostics are available:

CPU
Shows the processor ID, the number of (virtual) cores and usage of all CPU cores as well as the average CPU usage. If core usage exceeds 95% the graph will be red. A dark green graph indicates how the audio engine is performing, where 0 means it has 100% headroom and 100% means it is unable to keep up. Typically this line will be under 1%.

Memory
Shows memory usage of VidBlasterX.

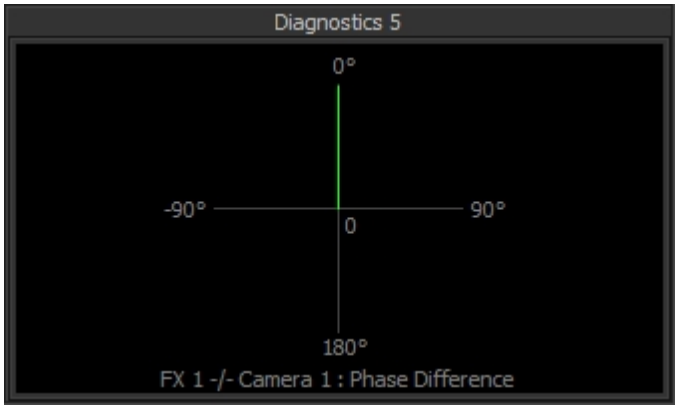
Audio
Shows the audio's latency, under- and overflows for the selected source (like AV Recorder). The latency is shown in ms and is corrected for any intentionally added latency for buffering and/or delay correction. Typically this value should be near 0 ms. Under- en overflows relate to corrections in the audio buffer and are displayed a a red line going down (underflow) or up (overflow) from the vertical middle of the graph. Ideally under- and overflows should never happen.

Video
Shows the video's time stamp delta and -delay and, for modules like NDI Input that use a PLL, the delta sigma for the selected module. The time stamp delta, plotted in green, is the increase in the video frame's time stamp, or in other words the duration of a video frame, measured at the output of the selected module. When the time stamp is not available (missing frame), a vertical red line is shown. The time stamp delay (yellow) is the difference in the video frame's time stamp between the input and output of the selected module, i.e. the time it takes the module to process the video frame. Where available the time stamp delta of the source (Camera) or buffer size (NDI) is displayed in dark green. Delta sigma (dark grey) is the amount of compensation the PLL requires to lock onto the input clock. If SyncLok is enabled, a light grey line at the bottom will indicate SyncLok is achieved. The vertical scale is in milliseconds (ms), horizontally each pixels represents one video frame.

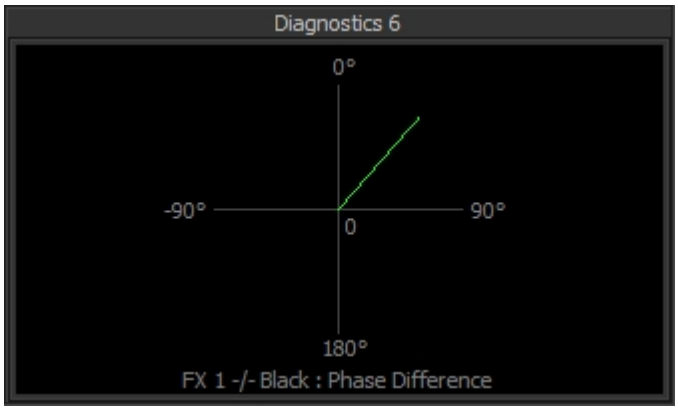


Even the most basic two-line video diagnostic is extremely informative, it shows the system's stability, video jitter and clock accuracy and indirectly the performance of each module and the underlying hardware. The example above shows the video diagnostic of a Slomo module running at 50 fps, so each frame lasts 20 ms. On the left you can see processing of a single frame (yellow line) takes about 12 ms on average, with spikes over 20 ms. The frame times (green line) are mostly 20 ms, but there's heavy jitter and during aforementioned spikes a frame is lost and the frame time doubles to 40 ms. Clearly this module is not up to its job and dropping frames. On the right the exact same setup but on a more powerful system. Note how the processing time of a frame is around 1 ms, resulting in a clean almost jitter free video signal. This module will not drop frames nor is it likely to ever do this considering its headroom. Using diagnostics you can guarantee your system is operating to broadcast specifications and spot any issues long before they are visible to the naked eye.

Phase
Shows the phase difference between the selected source and the selected *Reference*. The digit in the center of the meter indicates the number of "turns" (i.e. frames) the meter has made from zero.



Above image shows a diagnostic with the phase vector steady at 0 degrees, indicating the *FX 1* bus is synchronised to *Camera 1*.



Here the phase vector is slowly rotating clockwise, indicating the *FX 1* bus is not synchronised to the *Black* video signal (internal master clock).

Logic
A diagnostic similar to a logic analyser that can capture and display logical data of up to three sources based on a trigger event. Currently the logical data taken from a module is its frame clock, where an odd frame number is represented by a 1 and an even frame number by a 0. If no trigger pin is selected a transition from 0 to 1 in the first source (*Module 1*) will be used as a trigger. Currently supported triggers are the Video Switcher's *ProgramBus* and *PreviewBus* pins.

Data
Currently only supported by the Streamer module, this graph shows the stream's bit rate for the selected module. The vertical scale is in megabits per second (Mbps), horizontally each pixels represents one seconds

Right click the module to get access to its popup menu with the following entries:

Module
Module 1
Module 2
Module 3
Select the module that will be used as source for the diagnostic.

Reference
Select the module that will be used as reference for the *Phase* diagnostic.

Trigger
Select the module that will be used as trigger for the *Logic* diagnostic.

Copy
Makes a screen shot of the module and copies it to the clipboard.

Save As
Makes a screen shot of the module and saves it to disk.

Note the availability of menu entries varies with the diagnostic selected. Also note that, to save resources, diagnostics that are not selected will not be updated. If you require two or more diagnostics to be updated and/or visible at the same time, use one Diagnostics module per required diagnostic.

The Signal Generator Module

The *Signal Generator*[1] module can be used to generate various test signals.

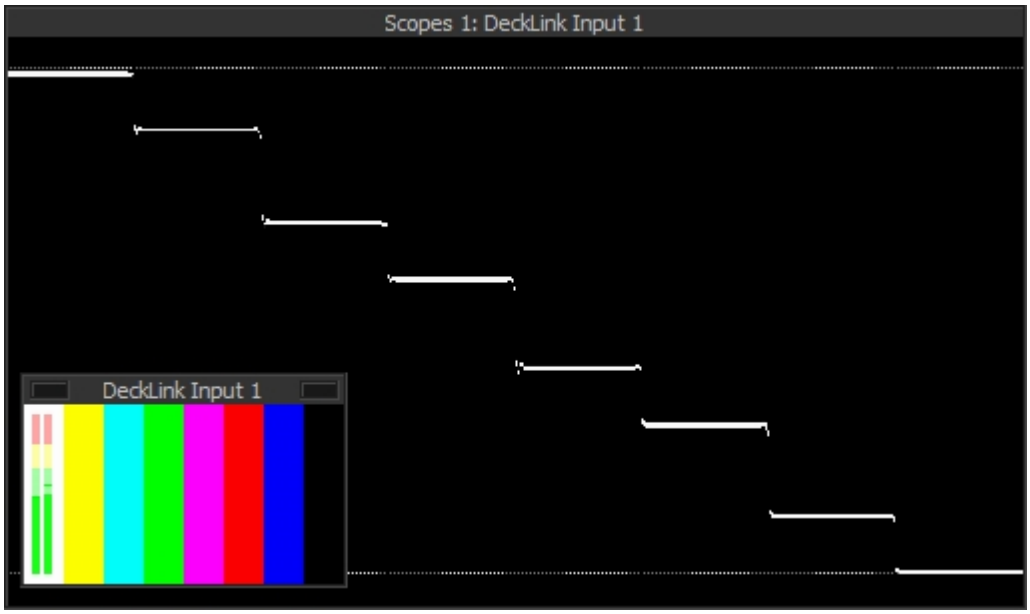
AV sync reference
White time/frame counter on black background. At every full second the frame inverts and is accompanied by 1000 sine wave at -24 dBFS, at every 10 seconds at -18 dBFS. The bar at the bottom consists of 32 evenly spaced squares that can be black or white. The first 25/30 represent the frame number, resetting to 1 each second. At rates > 30 fps each square represents two frames and the single last square (31) signals the odd frame numbers. The last square (32) signals the odd frame numbers since the start of each second.

Colour Bars
Colour bar pattern (white, yellow, cyan, green, magenta, red, blue, and black) with software, CPU & video information. Audio is a 1000 Hz sine wave at -18 dBFS.

[1] Available upon request.

The Scopes module

The *Scopes*[1] module includes a luminance waveform scope. It visualises the brightness of the video signal and is a great way to verify exposure, compare camera calibrations and test for proper green screen lighting.



[1] Available upon request.

Credits & Disclaimer

Thanks

Special thanks go out to Barry Furnival for making the early years of beta testing so much fun, to Luria Petrucci, Adam Curry, Roderick Vonhögen and Brian Brushwood for introducing VidBlaster to the rest of the world, to Jan Akalla for his relentless testing, his copywriting and immeasurable patience, to Martin Kay for his technical insights and strive for perfection, to Johan Lundberg, Peter Löfås, Rob Ashard and the Wilsson brothers for their expert feedback and lifting the program to professional levels. I could not have done it without you!

Credits

This product spawns FFmpeg (GPL v2). This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. This product includes cryptographic software written by Eric Young. This product includes the Simd Library by Ihar Yermalayeu. This software is based in part on the work of the Independent JPEG Group. This software includes or uses the TurboJPEG API, see the The Modified (3-clause) BSD License. Copyright (C)2009-2023 D. R. Commander. All Rights Reserved. Copyright (C)2015 Viktor Szathmáry. All Rights Reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Neither the name of the libjpeg-turbo Project nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. This software is provided by the copyright holders and contributors "As is", and any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall the copyright holders or contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage..

Disclaimer

Although this product has been thoroughly tested, CombiTech claims no responsibility for any damages caused by the use or misuse of this product. This product is distributed 'as is' with no warranty expressed or implied. CombiTech and its agents, distributors, resellers and other representatives will not be responsible for any losses incurred, either directly or indirectly, by the use of this product. Use this product entirely at your own risk.