VidBlasterX

Table of contents

Welcome!

Welcome!
Welcome to VidBlasterX, the easy to use video production tool to create anything from a single camera recording to a live multi camera television broadcast. VidBlasterX is a versatile and powerful ultra-low latency cascading vision mixer and video router with built-in scalers, time base correctors, frame synchronisers, IP video encoders and decoders, video players, recorders, keyers, effects and much more. 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.

Editions

VidBlasterX is available in 3 editions: Home, Studio & Broadcast. The difference between these editions is in the number of modules allowed in a single profile: the Home edition supports up to 7 modules, the Studio edition up to 25 and the Broadcast edition up to 50. The Broadcast edition also comes with several advanced features: UDP streaming and multiple recorder & streamer modules in a single profile. Another difference is the level of support: the Home & Studio editions come with community support, the Broadcast edition comes with both community and priority support.

The VidBlasterX Trial edition is equal to the Studio edition, with a watermark added to all output channels, and can be downloaded at download.vidblasterx.com. Beta releases and their download links are announced in the VidBlasterX Community.

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 version of the program, each will have their unique folder and shortcuts, enabling you to trial new versions while continuing to use your current production version. At first run the program will check for a recent older version and attempt to import its profile.

Upgrade v4 to v5
When upgrading a profile from v4 to v5, any overlay modules will be automatically replaced by *Still Store* modules. The overlay effect itself will be created using an *Effects* bus (*Modules > Add > Switcher > Effects Bus*) with two or more sources selected. Typically the first source will be *PGM 1* and to its right the still store module(s) with the overlay(s).

Resetting the profile
Changes in Windows, in device drivers or hardware may prevent VidBlasterX from starting properly. If you hold down the left *Shift* key while starting the program you will be queried if you want to load the default profile. Choose *Yes* and VidBlasterX will ignore any stored profiles and load the default profile instead, thus avoiding any errors caused by third party drivers.

Related videos

Licensing

Licensing VidBlasterX will remove the trial logo and optionally upgrade the edition to Broadcast. Go to the VidBlasterX site to order a licence.

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. VidBlasterX 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 full HD at 1080p. The result of this forward-thinking approach is that the VidBlasterX 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 (equivalent to the

Studio edition with all its features fully functional) 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 Product Key.

The licence
VidBlasterX comes 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 VidBlasterX on multiple (virtual) desktops, you will need to buy a licence for each one.

Licensing your installation of VidBlasterX is a two-step process which involves two elements: a product key and an activation key. It's important to understand the significance of these two types of key. They may look similar but they are not interchangeable.

The product key
A product key is a licence for a particular edition of VidBlasterX. This is what you are given (or sent by email) when you purchase VidBlasterX. Each time you extend your licence with another year you will be given a new product key. A product key is used to initiate any new installation or subscription prolongation of VidBlasterX. In these circumstances the dialogue box will ask for a product key. A product key will allow temporary licensing for up to 30 days, but for continued use an activation key is required.

The activation key
An activation key links a product key to the hardware of a particular PC. It acts to limit potentially illegal multiple installations as might occur as the result of software piracy. Each activation key has to be obtained by first installing VidBlasterX on the designated PC, entering the product key and then submitting an activation request code that encapsulates information about the PC's hardware and the product key used.

Obtaining and entering a product key
You can order a VidBlasterX licence either through the official VidBlasterX store on the VidBlasterX site or through your local reseller. The licence is delivered by e-mail as a product key. It is primarily your responsibility to take care of it and not to lose it. Making backup copies of the software and product key is not difficult, so make sure you do - and know where to find them! Copy and paste the product key into the registration window. You can bring up the registration window through the menu *Help* > *Registration*. If the product key entered is correct, you will be prompted with a success message and VidBlasterX will restart with the watermark removed. Your registered installation is now valid for 30 days. To continue using the registered version you also need to activate your installation.

Activating your installation
To continue using the registered installation you will need to activate it. Activation is done by copying the activation request code and sending it by e-mail to activate@vidblasterx.com. Make sure to place the request in the body of the email, not its subject, and not to attach any files. The activation request code is found in the registration window and is marked in red. If the request is correct you will shortly receive an activation key by e-mail. Copy and paste this key into the registration window and your VidBlasterX installation will be fully activated. Note that the request code and activation keys are generated with the help of your computer's hardware ID. Hence the activation key for one computer is not valid on any other machine. Major changes in the hardware setup (changing CPU, etc) can result in your system getting a new hardware ID. This will render your installation of VidBlasterX invalid. Just repeat the activation process to receive a new activation key which is valid for the new hardware ID.

Note: As the number of activations per licence is limited, make sure to finish all required hardware changes before applying for a new activation key!

Note: To make sure the email with your activation key arrives, please add CAR@mikeversteeg.com to your contacts. Whitelisting this email address will prevent the email from being blocked by your spam filters. Try to avoid free services like hotmail and outlook, they generally have poor filtering.

Support

Support for VidBlasterX depends on the licence you bought. The Home & Studio licences come with

community support. Communities have been established for your convenience on both facebook and google+. For direct email support by the developer support incidents are available for purchase. The advanced Broadcast licence comes both with community support and priority support. If you bought your licence directly from the developer CombiTech, priority support includes direct email support by the developer as well as priority with bug fixes and feature requests. Broadcast licensees with a professional background are also invited to join the special VidBlasterX Broadcast Professionals community.

Note: Emails to the support email address will only be processed if you purchased a Broadcast licence from CombiTech and your email address has been white-listed.

About CombiTech

CombiTech is a Netherlands based company owned by Mike Versteeg, developing software for over 20 years. Programs like Mscan Meteo, Mscan SSTV, StudioRack, CastBlaster and WinPodder all have high quality and ease of use in common.

SyncLok

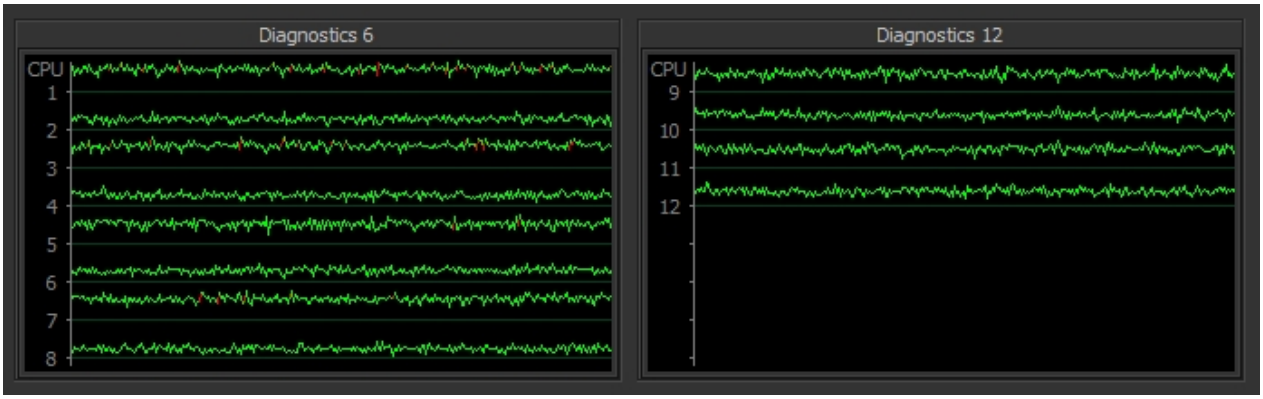
SyncLok is a proprietary technology developed by CombiTech. Professional (broadcast) vision mixers use genlock to ensure proper synchronisation between sources. This requires expensive cameras, video players, IP and other video sources. Not all VidBlasterX users will have a budget for this, so a different 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. For broadcast quality video output SyncLok needs to be enabled (see *Video > Options*) and obtained for the main studio cameras. The status of SyncLok can be (optionally) seen in a module's display (see *Appearance > OSD > SyncLok*).

BusWarp

BusWarp is a proprietary technology developed by CombiTech to reduce resource usage and latency in real-time video processing software. BusWarp dynamically analysis video routing and reroutes video streams in real time.

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 great 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 6 core (12 virtual cores) i7-5820K CPU ingesting 8 HD 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.

Enable SyncLok
If (most of) your cameras and other video sources have stable and proper timing characteristics then enabling SyncLok will ensure a minimum of lost or doubled video frames, removing any video jitter you may experience. SyncLok must be enabled and achieved to obtain television broadcast worthy output.

The Main Window

One of the most powerful features of VidBlasterX is its modularity. By choosing the building stones you need, 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.

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 > Recent Profiles
Profiles from previous versions are available here. Note importing old profiles may not always be possible and some modules or settings may be lost.

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 > 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 > API Command Stack
Used for debugging purposes. Opens the *API* window.

View > Advanced > Log
Used for debugging purposes. Opens the *Log* window.

Modules > Add
Add selected module.

Modules > Remove
Remove selected module.

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

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, or when the program's output is connected to another application or server.

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, or when the program's output is connected to another application or server.

Video > Option > SyncLok Enabled
SyncLok is disabled by default as it may have an adverse effect when used with video sources that have varying timing, or when using a PC that is not able to keep up with the processing of all video streams. Check this flag to enable it.

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

Help > About
Displays the About window.

Help > Registration
Displays the Registration window.

Help > Check for Updates
Displays program's version number, and the version numbers of the latest release and beta versions.

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 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*). Modules are magnetic, in that they snap to each other when dragged within the vicinity of another module. 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. To conserve space most modules can be collapsed by double clicking the title bar. Double click the title bar again to return the module to its original state. Note a collapsed module will still consume resources as it continues to function. Modules can have one or two tally lights: a program/preview/fx tally (left) and a status tally (right, often used to indicate on/off status).

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.

On
Activates the module's main function.

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.

Audio Output
The module sends its audio to this output. By default this is set to either the default Windows playback device or, when Audio Mixer 1 is present, *No Sound*. DirectSound & WASAPI devices are used in 2-channel stereo mode, ASIO devices in multi-channel mode.

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 selecting on when off, or off when on.

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

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

Appearance > 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. A multiview configuration, i.e. a collection of displays, can be achieved by disabling all controls. All modules remain fully operational through their popup menu.

Appearance > OSD
Allows you to select which On Screen Display elements, e.g. audio meter and status icons, are visible. The **audio meter** has a range from -52 to 0 dBFS, divided in 3 segments: green from -52 to -18 dBFS, yellow from -18 to -10 dBFS and red from -10 to 0 dBFS. Both left and right audio channels indicate the audio peak value in dBFS, with a peak-hold line indicating the highest peak value in the last 750 ms. The **SyncLok** status icon indicates whether the video device or source streams video frames with accurate timing (SyncLok active or "locked"), or not. If SyncLok is not possible, timing will be taken from VidBlasterX's internal master clock.

Appearance > Size
Sets the display size as percentage of the native video resolution.

Option > Alias
Opens a dialog allowing you to enter a module's *Alias*, which can be a more descriptive or compact version of the module's name. E.g. for a *Camera* module this could be the name of the camera operator, for an *NDI Output* module this could be the name of the actual source.

Option > 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, especially for HD streams). Select *Ignore* if VidBlasterX should ignore the alpha channel entirely.

Option > 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.

Option > 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.

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

Remove Module
Clears and removes the module.

Macros

Click *View* | *Macros* 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. Macros can even 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 be either entered, or recorded using the popup menu's *Record* command.

#
The macro number, used when referencing macros.

Description
Optional description of the macro.

Commands
One or more commands to be executed. Note commands use the 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*) or a midi command from an external device. Midi commands are entered as M<deviceid>:<channel><key>, e.g. *M0:0,1* for device #0, channel 1 and key code 1. 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.

Related videos

Audio Overview

VidBlasterX has two distinct audio systems built into its design: the legacy audio engine that is designed to route all audio to an external mixer and vice versa, and the new VBXA2 audio engine which is built around the Audio Mixer module. VBXA2 is a new audio system that brings the same flexibility to audio routing that you are already used to for video. VBXA2 offers high quality, low latency audio mixing and routing. Audio can be ingested from internal modules or external devices, mixed, and output to modules like the Recorder and Streamer while monitored on an external audio device.

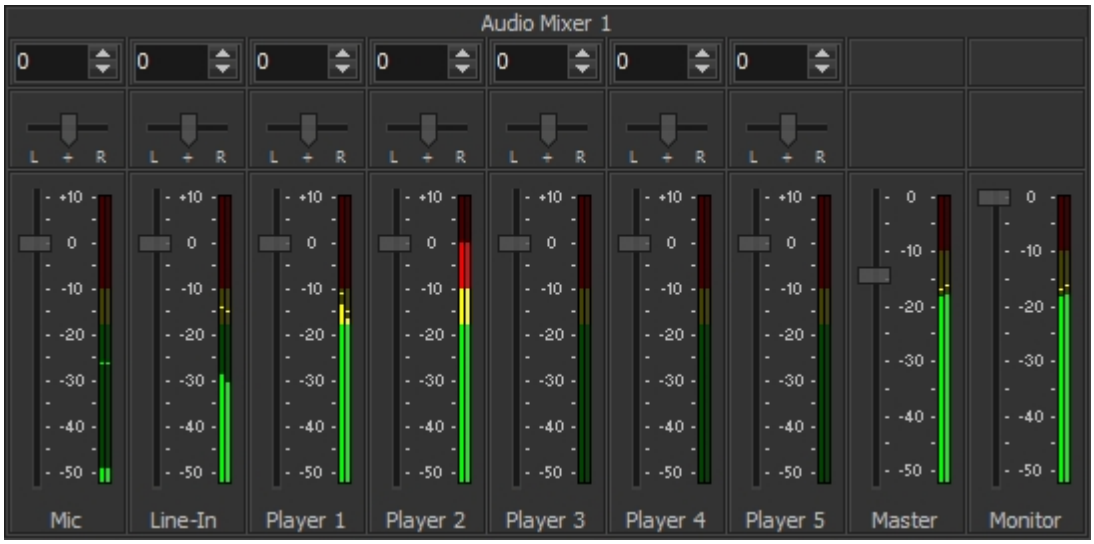
Although you can use both the legacy audio system and VBXA2 at the same time, typically you'll be using just one.

Using the legacy audio system
All modules that generate audio, like the Player module, have an *Audio Output* option in their popup menu where you can select to which external audio device you wish to route audio. This usually is part of an external audio mixer, or connected to it. Modules that ingest audio, like Recorder, have an *Audio Source* menu entry where you can select from which external device audio needs to be taken. Typically this is the output of an external audio mixer, or a sound card to which it is connected.

Using VBXA2
As audio from modules will be directly ingested by the Audio Mixer module, all modules that generate audio should not send their audio to external devices. This can be achieved by selecting *Audio Output* > *No sound*. The module is then selected in the Audio Mixer module. Modules that ingest audio will select the mixer as source, e.g. *Audio Source* > *Audio Mixer 1*.

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 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> Channel
By default an input is always processed as a stereo signal. This option allows you to use only the left or right channel, which is then mapped to both channels.

<input> Default
Set all controls on the channel strip to its 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 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.

Add Channel Strip
Adds a new channel strip for the selected input. Audio can come from internal *Modules*, external *Audio Devices* or *Audio Loopback Devices* (audio outputs).

Appearance > 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.

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 VidBlasterX by one or more *Camera*, *IP Input*, *NDI Input* or Screen Capture 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. Note many webcams offer video in compressed form and require (much) more resources than video capture cards due to the required decompression. 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 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.

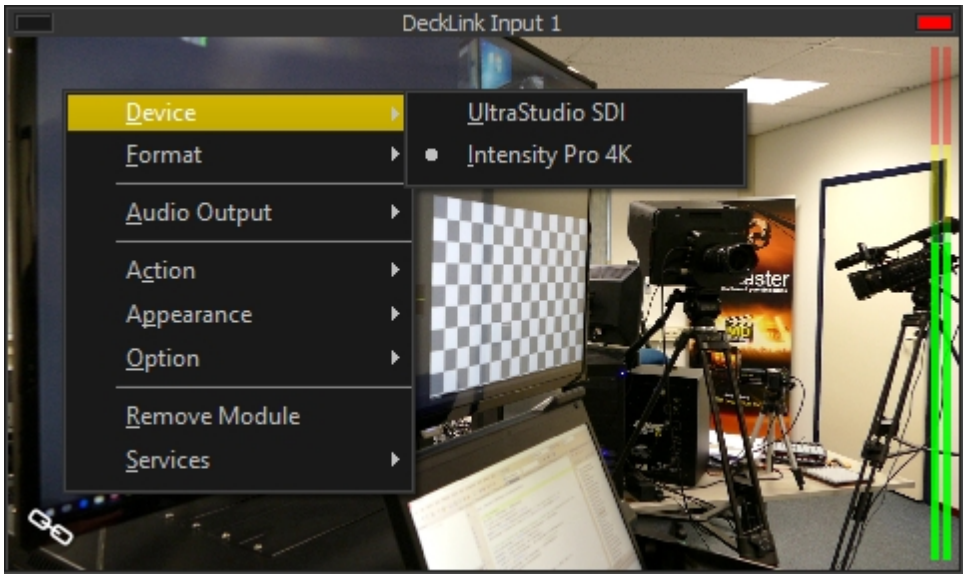
Video Frame Rate Multiplier
Converts the device's frame rate. This can be useful in case of driver errors, or when the capture card works at double the desired frame rate. Usually the default *Auto* settings works best.

Option > 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 a video stream from a Blackmagic DeckLink card directly, bypassing* its DirectShow driver. Due to their software processing* (and their many USB devices) DeckLink cards are more prone to frame dropping, so an additional Phase Locked Loop with buffering is added to keep the stream and buffer size (and so latency) constant.

*Blackmagic DeckLink cards are designed quite differently from other brands in that most processing is done in software. The advantage is lower production cost. The disadvantages are higher CPU and memory usage, higher latency, higher bandwidth requirements for the motherboard and a higher chance of instability when the system cannot cope. In addition the DirectShow drivers are poorly supported and at times unstable. Despite these disadvantages the cards remain very popular.



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

Device
A list of all DeckLink devices available.

Format
DeckLink devices are extremely picky and will only work if you select the exact same signal format as applied to the device. Colour bars will be displayed if you select a format that is supported but that does not correspond to the applied video signal.

The IP Input Module

The *IP Input* module is used to process and decode audio and video from incoming HTTP, RTSP, RTMP & (multicast) UDP streams. Typical use for this module is to ingest video from IP cameras. Popular codecs like H264 and MJPEG 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.

Low Latency
By default the decoder will use a dynamic cache size that is automatically adjusted to the quality of the stream. When this flag is checked the cache and audio are disabled so that minimal latency is achieved.

The NDI Input Module

The NDI[1] Input module is used to ingest a video stream from an NDI compatible source connected to a local network. Note NDI sources always use compressed video and may require more resources than video taken from video capture cards through the Camera module. Right click the module to open its popup menu. The following additional menu entries are available.

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

Option > Frame Buffer Size
Because NDI does not offer a steady stream of video frames (networks don't operate that way) a PLL is

used to retrieve a stable clock. The size of its buffer is selected here (default 2). 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.

Option > Hold Last Frame
By default NDI streams are assumed to be true 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 sends intermittent video frames.

[1] NDI™ (Network Device Interface) is a standard created by New Tek 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.

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.

Start
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.

Stop
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.

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.

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.

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.

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

Related videos

Monitoring Video

Video streams from any module can be monitored in the Monitor module. Up to 16 streams, 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 source modules (e.g. a camera module or the video switcher's program bus) including optional alias and tally. 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 source for each view. Note this option is only available if *Auto* mode has not been set.

Appearance > OSD > Alias & Tally
When set, each view includes the source's alias and a tally bar.

Option > Auto
When auto mode is set (default) 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).

Adding Video, Stills & Text

VidBlasterX supports many video formats, including Windows and Apple native formats as well as more advanced broadcast formats and MP3 audio files. The files are loaded into a *Player* module which can be used to play the video. 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 must can have both straight or premultiplied alpha, but for performance reasons premultiplied is preferred. Player can also play MP3 file 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 Click > Play/Pause
When set clicking the display has the same function has clicking the *Play* button.

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.

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

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 in combination with the Character Generator module and an Effects bus to create video overlays like lower thirds. 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 still (bmp, jpg, png) 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 option is on the still will be saved without further interaction, if it is off a *Save As* dialog will open.

Option > 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.

Option > DSK
By default the Still Store 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 (mimicking the behaviour of the v4 Overlay module).

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 Effect 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.

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

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.

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.

Recording

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

The Recorder Module

The *Recorder* module[1] records video from any source together with audio from any 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.

Settings
Opens the *Settings* dialog where the file format can be selected. Currently 5 container types are supported: Matroska, MPEG-2 Program Stream, MPEG-2 Transport Stream, MPEG-4 and QuickTime. Note both Matroska and MPEG-2 TS have the advantage that the file will always be readable, even if it is not properly closed (e.g. if a power outage occurs during recording). Matroska is also the only container that supports uncompressed audio (PCM). The video codec is automatically selected based on the container type: the

MPEG-2 containers use the MPEG-2 code, all other containers will use H.264. The video bit rate is variable to keep a constant (high) quality. The video compression setting also affects the bit rate, where the low setting results in the highest bit rate and the lowest CPU usage. This setting is only used for H.264 compression and has no effect on MPEG-2 compression. Best video quality is achieved by selecting RGB444 as colour space, although not every video player supports this. Same goes for MPEG-2 which is not supported by the default Windows Media Player. The audio codec (AAC, AC3, MP3 and PCM) is manually selected as is the audio bit rate. Use the *Audio Delay* setting to obtain perfect audio/video sync in recordings. Typically audio will be early as video takes longer to process, so a positive audio delay should be set.

Save Recording To
All recordings will be stored in the folder selected here. The file will be automatically named VidBlasterX_yymmddhhmmss, e.g. VidBlasterX_140818143000 on August 18th 2014 at 2.30 PM. When more than one recorder module is used an additional index will be added to the file name (e.g. VidBlasterX_140818143000_1). The default folder is the "My Videos" folder. For optimal performance, especially with HD video resolutions, it is highly recommended to select a dedicated SSD drive here.

[1] Broadcast edition supports multiple Recorder modules

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 Facebook and Youtube. Streaming live video to selected CDN's like DaCast is even easier using the available presets. VidBlasterX Broadcast supports multiple streamer modules, allowing you to quickly and easily setup multiple streams.

The Streamer Module

The *Streamer* module[1] 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.

Settings
Opens the *Settings* dialog where the server type and codecs can be selected. Note UDP Multicast uses the MPEG-2TS container, all other server types use the flash video container. The video codec is always H264. By default (*Auto*), the stream's video resolution is the same as the main video resolution. If you prefer to stream in a lower or higher resolution then select it from the list. Note that due to resampling of the video frames, the resulting video stream's quality may be less than optimal if you select a different resolution than the main video resolution. The video and audio quality of the live video stream is influenced by the encoder compression setting and the individual bit rates. For the best video quality choose the highest bit rate that is possible, but keep the combined audio and video bit rates below the sustained (not average or peak!) upstream bandwidth of your internet connection. Note some CDNs don't support more than 500 kbps (total) for their free accounts. Default video bit rate is 200 kbps. A higher compression setting will give a better video quality for a given bit rate at the expense of higher CPU usage. If CPU usage is not an issue, e.g. for dedicated streaming systems, use the highest setting. Choose lower compression settings to lower CPU usage. Typically the *Medium* setting (default) is a good compromise between CPU usage and video quality. Audio can be encoded both in *AAC* and *MP3* format. If the server breaks the connection, or fails to respond for 30 seconds, the streamer will try to automatically resume streaming if the *Auto Restart Stream* flag is set. Note if you are seeing a lot of restarts (and timeout related messages in the log) you either are trying to stream at too high a bit rate, or there is a serious problem with your network or ISP (network dropouts or collisions).

The following server types are supported..

RTMP/Flash Media
Typically servers running either Adobe Flash Media Server software or Wowza, which includes all popular live streaming services like Ustream and Youtube Live. Enter the server's *URL* and *Stream* and, if authentication is required, a *Username* and *Password* (leave these blank for Youtube Live).

UDP Multicast[2]
Requires the IP address and port to be entered (*IP:Port*). UDP Multicast uses the MPEG-2TS container, H264 video codec and MP3 audio codec.

CDN: DaCast
CDN: Livestream Original
CDN: SaleEngine
CDN: Solidtango
CDN: Sunday Streams
Enter your account's *Username* and *Password* and enter or select a *Channel*. Note you may need to refresh the list by selecting <*update*> first (not supported by Sunday Streams).

CDN: Facebook
Requires the *Stream* (facebook calls this a stream key) to be entered. *URL* and *Audio Codec* and bit rate are preset according to facebook guidelines. To start a live stream and obtain the stream key, first login to your facebook page and select *Publishing Tools* in the top navigation bar. On the left menu, click on the *Video Library* option under the *Videos* section. Click on the + *Live* button to begin configuring your live post. The *Stream URL* and *Stream Key* will now be displayed, you only need to copy the stream key. You can now start the VidBlasterX streamer. On facebook Preview and check your stream. Click the *Go Live* button when you are ready to publish your stream on facebook.

[1] Broadcast edition supports multiple Streamer modules
[2] Broadcast edition supports UDP Multicast

Video Output

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

The Display Output Module

The *Display Output* module is used to send video from any source to another monitor using one of the graphics cards already installed. 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.

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

On
Starts output of the video.

Off
Blanks the video output (black).

Action > On Click > On/Off
When set clicking the display starts/stops the video output.

Option > 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 DeckLink card like the Intensity Pro (HDMI) or DeckLink Duo (SDI).

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 DeckLink card (port) to which the video will be sent.

Video Output Mode
Select the DeckLink card's output video resolution and frame rate. Note due to a bug introduced after Desktop Video version 10.8.3, interlaced modes only work when the *Low Latency* flag is set.

Option > Low Latency
Set this flag when low latency video output is required. Note this disables audio. Low latency mode will typically be used for studio floor monitors, live-event monitor feeds and IMAG. On modern (4K) DeckLink cards this will bring latency down to one frame or less.

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.

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

On
Starts output to the NDI source.

Off
Stops output to the NDI source, last video frame sent will always be black.

Note due to a bug in NDI3 an access violation may occur if you remove this module (this includes reloading a profile). Newtek is working on this. Meanwhile restarting VidBlasterX suffices as workaround.

The Virtual Video Output Module

The *Virtual Video Output* 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 like Skype[1], Google Hangouts or Flash Media Live Encoder.

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

On
Starts output to the virtual video device.

Off
Stops output to the virtual video device, last video frame sent will always be black.

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

[1] Unfortunately Skype has blocked the use of virtual video devices after v7.16.0.102 and also prevents the use of older versions. Any developments on this w ill be posted in the VidBlasterX Community. Meanw hile a w orkaround is to output video to a separate

display and use Skype's screen sharing feature to get the video into Skype.

Replays & Scores

Video replays are often used in sport matches, where a brief period of video from e.g. a camera can be played again. The Replay module adds a variable delay of up to 30 seconds to a selected video source, and in turn makes the delayed video available as new video source

There are in 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

One or more *Replay* modules can be added by selecting *Modules* | *Add* | *Replay*. The module stores all frames in memory and is therefore very fast, light on CPU usage, but does require additional memory (up to 5.6 GB for full HD @ 30 fps). The formula to get the memory requirement per replay module in MB is <vertical resolution> * <horizontal resolution> * <frames per second> * 3 * 30 / 1000000. If there's not enough memory available, the module will be disabled.

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

Delay
Set the delay in seconds.

The Scoreboard Module

The scoreboard 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.

Video Switching, Mixing & Effects

Video switching, mixing, keying and other video effects 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, playout controller 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 (*Key* being the default). 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 similar to the auxiliary bus but with optional A/B function. Typically used for the video stream to be broadcasted 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.

Source Settings
Select the desired video effect here including all relevant settings. Note sources are overlaid from left to right, start with the background. Available for the effects bus only.

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 Effects Bus

The switcher's *Effects (FX)* bus is mostly used to create a composition of multiple video streams and change their position and size and/or key one stream over the other.

Related videos

Cropping, Positioning & Scaling

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.

Note cropping is not supported by the chroma keyer.

Related videos

Keying

VidBlasterX has a built-in keying facilities, both for video with *Embedded alpha* and video with a fixed coloured background (green screen or *Chroma* keying). Keying is used to overlay one video stream over another. With alpha keying the alpha channel (transparency information) is embedded in the source, e.g. a player or still store module can offer frames with embedded alpha. With chroma keying the transparency information is part of the image itself, where transparent pixels usually have a green colour. Alpha keying is the default key setting, for chroma keying you need to select the chroma keyer. The chroma keyer is an easy to use, yet advanced 3D keyer with chroma blurring and spill suppression features optimised for green screen keying. Thanks to efficient machine-level coding the keyer runs entirely on the CPU so that very low latencies of just a few ms can be achieved, making it ideal for live presentations. To enable the chroma keyer for a source on the FX bus select *Settings* from its popup menu, this will open the *Settings* dialog. Now select the *Key* tab, select *Chroma* type and *Enable* it. The following settings are available.

Auto Key
This will calculate the chroma key and its related distance settings fully automatic. If the video frame does not show sufficient green screen, or if you prefer to select the key manually, then right click the dialog and choose Advanced Settings to open the chroma keyer's advanced settings window.

Chroma Blur
Ideally a video camera will output its colour information (chroma) in the same resolution as brightness (luma), but usually this is much less. Typically a camera will output in 4:2:2 (chroma sampled at half resolution) or even 4:2:0 (chroma sampled at one quarter resolution). As the name implies, a chroma keyer uses the chroma part of the video signal to key, and as a result keying borders will look jagged. Enabling this setting will smooth out those borders. This option uses additional resources and increases latency so only use this option when required. Besides using a better camera, chroma blurring can also be avoided by downscaling the video to be chroma keyed, either by using the Scale effect or by setting the camera in a higher resolution.

Spill Suppression 1
Spill Suppression 2
Colour from the background (green screen) can spill into the foreground image in various ways: by chroma blur and reflection onto the edges of the foreground image, by reflection of walls onto the entire foreground

image, and by bleeding through semi-transparent parts of the foreground (like hair). Spill suppression tries to reduce this spill as much as possible without actually changing the foreground image. For this two different techniques are available: spill suppression 1 which works in YCbCr colour space and spill suppression 2 which works in RGB colour space. Each technique has its specific advantages and disadvantages. Use either one or both at your discretion, keeping in mind both require additional resources and increase latency. Good lighting can prevent spill and is always preferable.

Note: to maintain high quality judder free video, keyed videos will be synchronised to the background. Videos that have a frame rate not matching the background (the main video frame rate) will be played faster or slower in order to match up with the background frame rate. If this is undesired then produce your videos with the desired frame rate.

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.

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
Returns the alias of a module.

audioinput
: Sets audio input.

audiosource, audiodevice[1]
: Sets audio device.

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.

videosource, source[1]
: Reads or sets the video source.

Examples

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

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

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

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

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

The TCP Server Module

The *TCP Server* module enables other applications to access the API from anywhere in the world. When loaded it will display the local IP address and current port number 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.

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.

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 Player Pins

Pin

file
Reads or writes the audio file name. When writing the value parameter is text string with the name of the file.

play
Executes the play command, identical to clicking the *Play* button. No action if button has *Pause* caption. Value parameter is ignored.

playpause
Executes the play or pause command, identical to clicking the *Play* or *Pause* button. Value parameter is ignored.

stop
Executes the stop command, identical to clicking the *Stop* button. Value parameter is ignored.

Example

The following example starts Audio Player 1.

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

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

Character Generator Pins

Pin

text
: Set the text.

Example

The following example changes the text of Character Generator (CG) 1 to "Hello World!".

```
apiwrite cg 1, text, Hello World!
```

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
```

Macro Pins

Pin

execute
: Executes the macro. Value parameter is ignored.

Example

The following example executes macro #1.

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

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 seconds.

elapsed
: Returns the elapsed time in seconds.

file
: 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
: Sets position in either milliseconds or percent. Value parameters is numerical string (ms) with optional percentage sign (%).

stop
: Stops the player. Value parameter is ignored.

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%
```

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
```

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
```

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

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
```

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

Switcher Pins

Pin

auto
: Executes current transition. Note this command is only supported by A buses that are set in A/B mode.

deselect
: Resets active sources in bus. Value parameter is a comma-separated list with source names.

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

select
: Sets active sources in bus. Value parameter is a comma-separated list with source names.

selected
: Returns comma-separated list with names of active sources.

sources
: Returns comma-separated list with names of all sources.

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

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

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"
```

Diagnostics

If you ever find yourself in a situation where the application does not work 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. 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. 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:

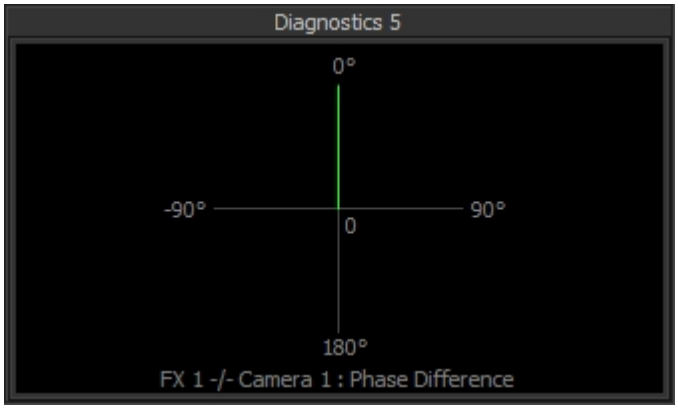
CPU Core Usage
Shows the usage of the first 8 (virtual) CPU cores. If more than 8 cores exist a second tab named *CPU Core Usage 2* will be available, showing cores 9 through 16. If core usage exceeds 75% the graph will be red.

Memory
Shows memory usage (green) and memory available (red) in GB. Additionally the memory consumption of a module's child processes can be display (yellow).

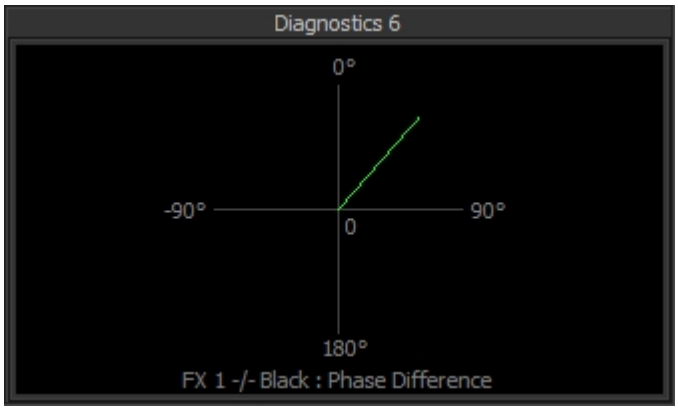
Audio
Shows the audio's latency, under- and overflows for the selected source (like AV Recorder). The latency is shown in ms and is corrected for any intentionally added latency for buffering and/or delay correction. Typically this value should be near 0 ms. Under- en overflows relate to corrections 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. 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. The graph can be a very useful indicator for the stability, jitter and accuracy of the video clock and indirectly of the performance of each module and the underlying hardware.

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 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.

Credits & Disclaimer

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.

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 for his expert feedback and lifting the program to professional levels, and to all the VidBlaster(X) MVPs out there. I could not have done it without you!

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.