

MULTI-ENCODER MODULE FOR BMD PCIE CARDS (update 2015/08/10)

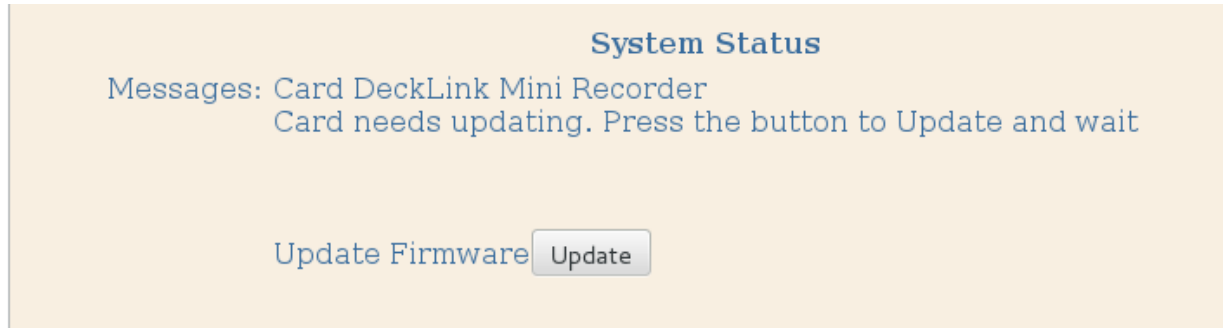
Introduction

Welcome to our new Multi-Encoder Module for Black Magic Designs (**BMD**) PCIe cards. Before reading this manual, you will need to have read the Linux Multimedia Universal Installer (**LIMU**) manual, have installed the Multi-Encoder module on it and have a **BMD** PCIe capturer card with at least one video input on it. Because version of 2015/08/10 includes the latest version driver 10.4.2, it should allow to work with all the new PCIe cards (all PCIe Decklink series and Intensity Pro 4K). If new cards are launched in the future, we will just publish an .upd file with the latest drivers to support it, and maybe new settings in the user panel. So you won't need to delete all of your previous settings to just add new capabilities to your computer.

This new software adds a new codec, the H.265 advanced video codec, that is able to use almost the half of the bandwidth of H.264 at the same quality. However, there is a complete lack of H.265 of players and hardware at the moment, so if you want to play H.265 streams, you will have to use our Player module. Because of its higher complexity in the algorithms of compression H.265 needs almost 3x or 4x the power of H.264 with the same quality. To handle an SDTV H.265 compression you will need an 8 hyperthreads CPU @ 3.6 GHz. However playback is very seamless, and will use only 2 hyperthreads @ 3.4 GHz even for Full HDTV H.265.

Handling the Encoder

First of all, we should have a glance at the Status tab, to see if everything is well detected.

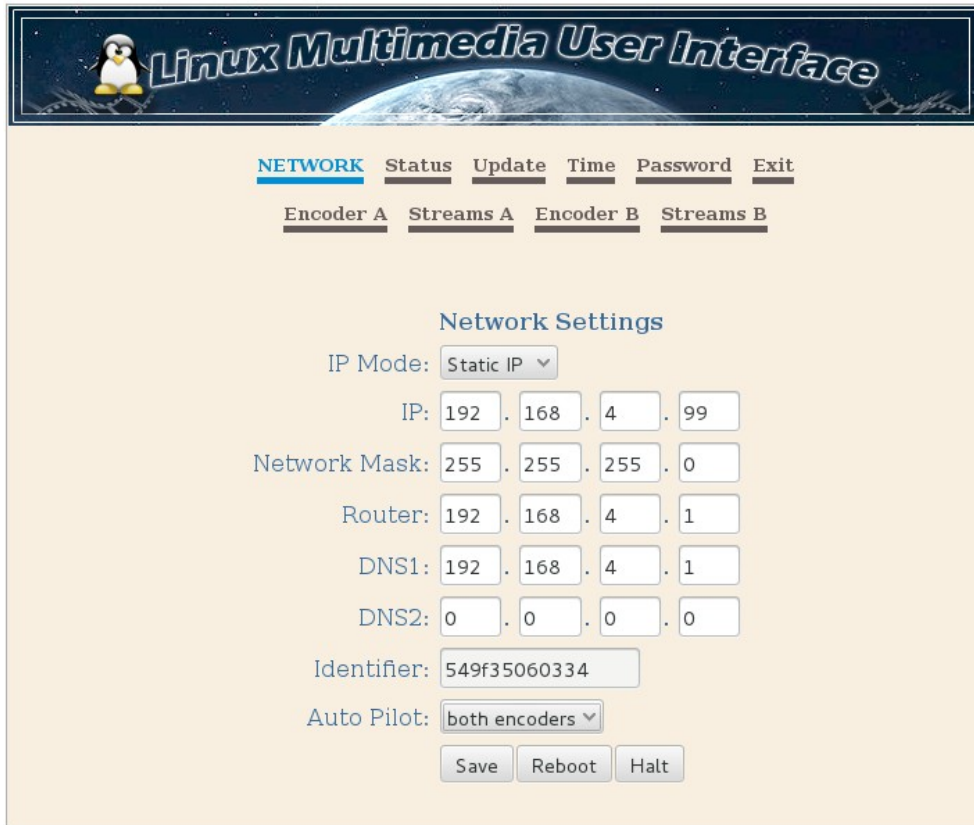


Here we have a very common situation, where the card has been detected or not, and needs a firmware update to work properly. When new BMD drivers are added or brand new cards are plugged in, this is a very common picture. So just connect your computer to the Internet and press Update. In seconds it will start to download the latest Firmware from BMD servers and will burn it on the ROM of the BMD PCIe card to work properly with the latest loaded drivers.



Do not power off your computer until this process ends, it could damage your card. Once you see the 100% completed message, you will be conducted to reboot your machine from the Network Tab. Once rebooted, the new driver will work properly with the new driver and you will see on Status a message with the name of the card detected by the driver, and of course “card successfully updated”. Now we can start to work with the encoder.

The user interface also allows to change the configuration of our network card



The screenshot shows the 'Linux Multimedia User Interface' window. At the top, there is a navigation bar with tabs: NETWORK (selected), Status, Update, Time, Password, and Exit. Below this, there are sub-tabs: Encoder A, Streams A, Encoder B, and Streams B. The main content area is titled 'Network Settings' and contains the following fields:

- IP Mode: Static IP (dropdown)
- IP: 192 . 168 . 4 . 99
- Network Mask: 255 . 255 . 255 . 0
- Router: 192 . 168 . 4 . 1
- DNS1: 192 . 168 . 4 . 1
- DNS2: 0 . 0 . 0 . 0
- Identifier: 549f35060334
- Auto Pilot: both encoders (dropdown)

At the bottom of the settings area, there are three buttons: Save, Reboot, and Halt.

The autopilot option will make the computer work as an encoder automatically when powered up with the settings entered in the Encoder tab, so if power is disrupted, when repowering up the device will start encoding a stream without our intervention. You will have to choose from using only the main encoder (Encoder A) or both encoders at the same time (Encoder A and B).

The Encoder A tab has many options, that we will describe in detail:

From Video Input to Output Size, there are all the options related to the capture and encoding of the video signal.

Video Input.- to select the video connection from where we capture the video

Video Mode.- to select the video format as is ingested. If you select a bad choice the encoder will insert a color bars signal. The software needs to know the video input format as is ingested.

Aspect Ratio.- here you can change the original aspect (anamorphic it uses 1:1 pixels)

Frame/Field.- here you can capture the original fields and frames as they are ingested, or you can deinterlace (very useful for WebTV) or progressive them so all the fields are kept on whole frames just doubling the frame rate.

Video Format.- here you can select your compression between H.264 and the new H.265 codec.

Linux Multimedia User Interface

Network Status Update Time Password Exit

ENCODER A Streams A Encoder B Streams B

Main Encoder Settings

Video Input: HDMI

Video Mode: PAL

Aspect Ratio: 16/9

Frame/Field: Original

Video Format: H.264 (AVC)

VBR Quality Level: 7 (increases quality and CPU usage)

VBR Compression Level: 1 (increases bitrate and decreases CPU usage)

Constant Bitrate Video: 0 kbps (0 = VBR)

Keyframe every: 2 second(s)

Audio Input: Embedded

Audio Input Level: 0 dB

Audio Format: HE-AACv2

Audio Bitrate: 48 kbps

Serial Data: none (9600 bps 8N1)

Output Size: 2/3 SDTV 480 x 384

Save Start

The next 3 option, work together, so we'll have 2 possibilities when encoding:

- Constant Bitrate(CBR): the bitrate will be constant in time, but the quality will change.
- Variable Bitrate(VBR): the bitrate will change to make quality to be constant in time.

If we decide to use CBR, we will have to set a non-zero value in kbps inside the Constant Bitrate field, and the VBR fields will be bypassed (ignored).

If we decide to use VBR, we will have to set a 0 value in that field, and set the quality level and the compression level of our encoding. The quality level goes in a range from 0 to 10, so the highest quality will use more CPU power. A good value is from 5 to 7. The compression level will decide which algorithm of compression to use. Here the range goes from 1 to 6. The higher value will use an easier algorithm so it will use less CPU power, but the compression efficiency will be lower, so the bitrate will be much higher. A good value is 1 or 2.

Keyframe every.- a good value is 1 or 2 seconds for better quality.

Audio input.- Embedded means inside a digital input signal, such as SDI or HDMI.

Audio Input Level.- It applies a gain to the audio input to increase (positive values) or decrease (negative values) the original audio level

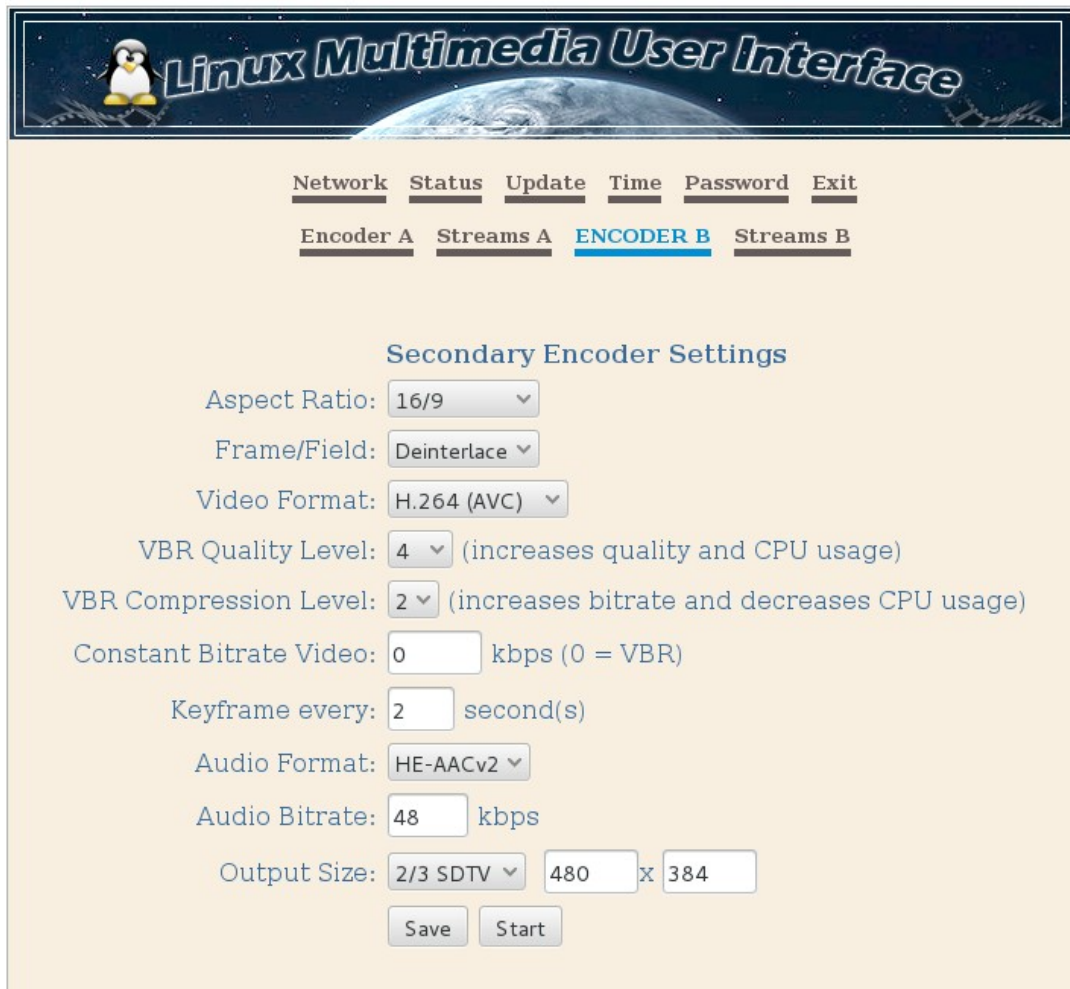
Audio Format.- here you can select your AAC profile (HE-AACv2 is the most advanced profile for audio compression with better compression efficiency at low bitrates level such as 32 o 48 kbps).

Audio Bitrate.- enter the bitrate to use in your audio compression profile.

Serial Data.- it will capture data from serial input (COM or USB port) to be attached to the video frames for remote controlling.

Output Size.- Custom will give us the choice to select the output size resolution. WebTV is for publishing video on web. Mobiles makes the stream to be compatible with mobile devices. The rest of profiles are obviously used for professional broadcasting.

The Encoder B only works once the Encoder A has been started. It takes the source from it and will build the second compression at a supposed lower bitrate. It has less options, and all of them has been already explained.



The screenshot displays the 'Linux Multimedia User Interface' with a navigation menu at the top. The 'ENCODER B' tab is selected. Below the menu, the 'Secondary Encoder Settings' are configured as follows:

- Aspect Ratio: 16/9
- Frame/Field: Deinterlace
- Video Format: H.264 (AVC)
- VBR Quality Level: 4 (increases quality and CPU usage)
- VBR Compression Level: 2 (increases bitrate and decreases CPU usage)
- Constant Bitrate Video: 0 kbps (0 = VBR)
- Keyframe every: 2 second(s)
- Audio Format: HE-AACv2
- Audio Bitrate: 48 kbps
- Output Size: 2/3 SDTV, 480 x 384

Buttons for 'Save' and 'Start' are located at the bottom of the settings panel.

Every encoder has its own Streams tab by its side. There you can enter up to 4 different copies of your encoder stream, to a final RTMP or UDP end. Just enter your settings and end URL and press Send to start the copy to stream.

Main Encoder Streams


	Stream 1:
options:	H.265 signalling: <input type="button" value="on"/> Rude URL: <input type="button" value="yes"/>
User/Pass:	<input type="text" value="user"/> / <input type="text" value="pass"/>
Publish URL:	<input type="text" value="udp://224.0.0.1:5000"/>
	<input type="button" value="Save"/> <input type="button" value="Send"/>

Network Status Update Time Password Exit
Encoder A Streams A Encoder B STREAMS B

Secondary Encoder Streams

	Stream 1:
options:	H.265 signalling: <input type="button" value="on"/> Rude URL: <input type="button" value="no"/>
User/Pass:	<input type="text" value="user"/> / <input type="text" value="pass"/>
Publish URL:	<input type="text" value="rtmp://domain.test.org/test/livestream"/>
	<input type="button" value="Save"/> <input type="button" value="Send"/>

Once we have video and all the Encoder tab options are set, we can press the Send button. A red label below will notify it. You may also follow it at the Status tab.

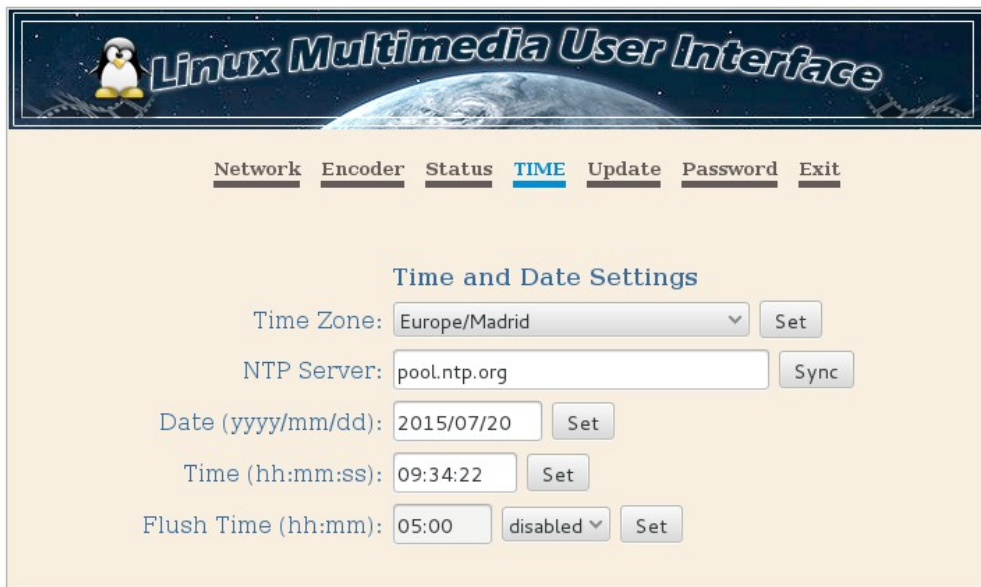


Network STATUS Update Time Password Exit
Encoder A Streams A Encoder B Streams B

System Status

Messages: Card DeckLink Mini Recorder
Card succesfully updated
Main Enc: frame=829 fps=26 time=31.12 bitrate=62.1kbits/s
Second Enc: frame=713 fps=31 time=26.52 bitrate=61.7kbits/s
CPU: 5.80%

In the Time tab you will find obvious settings to have your internal clock on time. You can also program a Flush Time when all the encoders will be restarted. This will make the stream to disrupt and reinitialize in 1 second. This is preferred for some people to avoid an increasing of the delay buffer in the long run (working during days) due to network issues.



The screenshot shows the 'Linux Multimedia User Interface' web panel. At the top, there is a navigation menu with tabs: Network, Encoder, Status, TIME (highlighted), Update, Password, and Exit. Below the menu, the 'Time and Date Settings' section contains several input fields and buttons: 'Time Zone' is set to 'Europe/Madrid' with a 'Set' button; 'NTP Server' is 'pool.ntp.org' with a 'Sync' button; 'Date (yyyy/mm/dd)' is '2015/07/20' with a 'Set' button; 'Time (hh:mm:ss)' is '09:34:22' with a 'Set' button; and 'Flush Time (hh:mm)' is '05:00' with a 'disabled' dropdown and a 'Set' button.

You can Update your encoder module loading our .upd files when published. If the encoder is working, a message will tell you to stop it before updating.



The screenshot shows the 'Linux Multimedia User Interface' web panel. At the top, there is a navigation menu with tabs: Network, Status, UPDATE (highlighted), Time, Password, and Exit. Below the menu, there are sub-tabs: Encoder A, Streams A, Encoder B, and Streams B. The main content area is titled 'Update Software' and displays 'Software version: 20150727-multiencbmdpc64'. At the bottom, there is a file upload area with a button labeled 'Examinar...', a message 'No se ha seleccionado ningún archivo.', and an 'update' button.

There is also a Password tab to change those settings for security reasons. By default the username/password is **admin/admin** but it is not related to the one in the admin's panel.

If you want to change the panel design with your own logo, text and/or CSS design, you can download the panel zipped from your own encoder. Use the URL of you user panel adding /panel.zip at the end (i.e <http://192.168.1.10/panel.zip>). Change whatever you want inside that panel and to upload it, you need permission from the admin's panel, unlocking the Brand option in the Network tab. Once unlocked, you have to be logged in the user panel to change and add to you user's panel URL /branding?id=identnumber to access the upload menu. (i.e: <http://192.168.1.10/branding?id=549f35060334>)

Remote control

Our encoders and player modules for BMD PCIe cards, are able to send and receive serial data from /to the COM port and USB (with a USB to serial converter). Therefore you will be able to control remote devices and playout systems from the encoder stream, to the receiving players. The control data travels attached to the video frame, so it will not be deleted by servers that only repeat the original video without recompressing it (i.e: Wowza, Evostream and so on). Even if you send an RTMP stream to say a Wowza Server, and you receive the original video from an HLS url, the data will be still there, in sync with the exact frame of video. We based our data format on that one used by Localia in Spain, so as them we decided to use 7 chars on every serial command at 9600 bps 8N1 in a COM port. So every frame could contain 7 chars of there serial commands inside. With this 7 chars in ASCII format you can build almost 2^{56} different commands, and only using capital alphanumeric chars the first one only alpha (i.e: H203500) we can build more than 56 billions of commands. I guess this is more than enough for anything you might want to do.

Localia used only these commands:

I010102 = switch to local playout

I010001 = switch to global playout

Hxxyyzz = time synchro xx:yy:zz (hour:minutes:seconds)

With these very few commands, you can build a complete TV network with receiving playouts (call them slave playouts) controlled by another one (we can call it the master playout in the main studios), so every action will be triggered in the right time, no matter the final latency in the players side, because the serial data commands will be always synced to the video frames to be rendered at every moment.

And that's all. Have a nice time with the Multi-Encoder module !!!

On August 2015