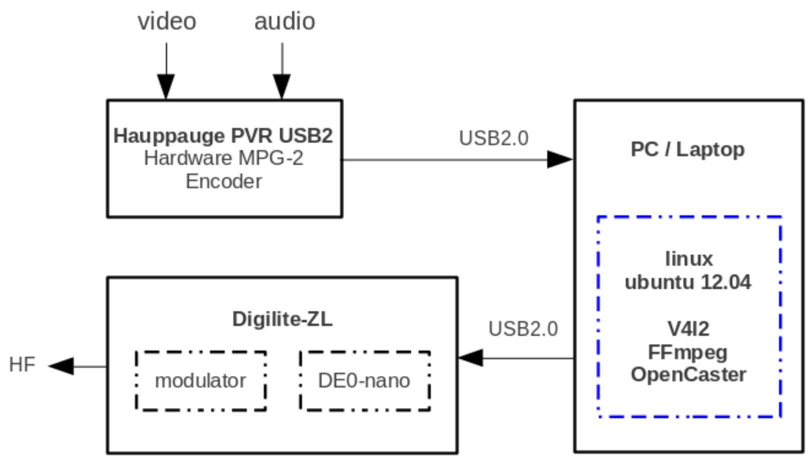
Overview

DigiLiteZL is a DVB-S transmitter device designed to be used for digital amateur television (DATV) transmissions, nominally in the 1.2GHz amateur band. It receives an MPEG-2 transport stream from a PC via a USB connection, performs various coding operations according to the DVB-S specification, and generates a QPSK-modulated output waveform.

This document describes one of the solutions to generate the live MPEG-2 transport stream to feed Digilite-ZL equipped with his DE0-nano FPGA board . The proposed solution is based on a pc or laptop running linux Ubuntu 12.04 LTS with additional multimedia utilities and a video capture card from Hauppauge.

The installation process described below has been tested starting with an empty disk on which ubuntu has been installed.



Hardware

The main hardware, in addition of course to the Digilite-ZL and a PC, is a video capture card from Hauppauge that uses a MPEG-2 Connexant compression chip.

There are a number of different models that you can use:

1. Hauppauge PVR150, PVR250 and PVR350 PCI video capture cards (pc)
2. Hauppauge WinTV PVR USB2 and HVR-1900 external devices (laptops)

Please remember that you need two USB port if you implement the solution on a laptop. One for the Digilite-ZL and one for the Hauppauge box.

The setup described in this document uses the WinTV PVR USB2 external device and a a no-name laptop Intel Core Duo CPU 2.66 GHz CPU and 2 Gb RAM.

Software installation

The proposed solution to generate the live MPEG-2 transport stream is implemented on linux operating system with multimedia toolsbox available from the linux community.

This comprises: linux Ubuntu 12-04, v4L2-utils utilities, FFmpeg tools and Opencaster 3.1.4-

Ubuntu 12.04 installation

The software can be downloaded from http://www.ubuntu.com/download

With the CD generated, you can try Ubuntu without affecting your current system. If Ubuntu load and run without problem from the CD, you can install it automatically from the CD. If you have windows installed on you pc, the installer will detect this automatically and make a second partition for Ubuntu as well as the dual boot configuration.

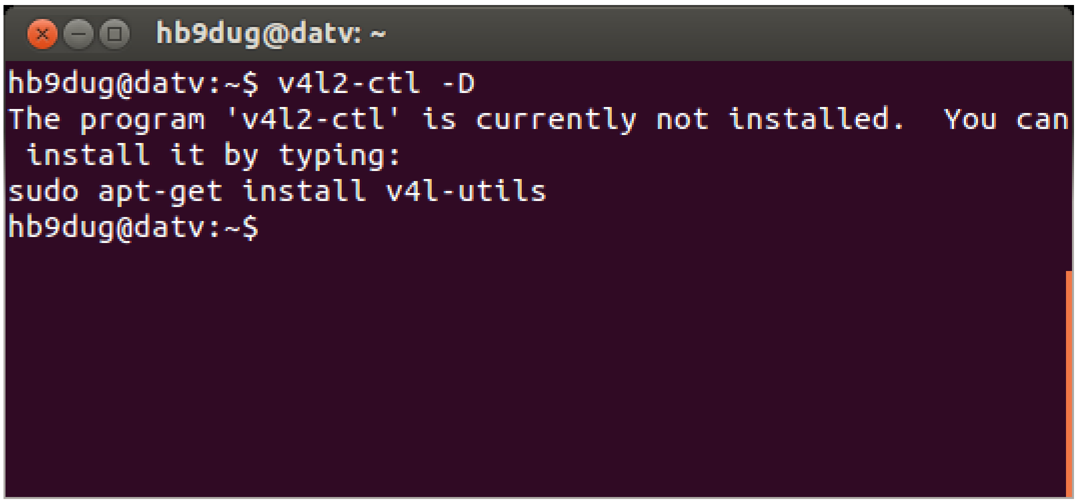
V4l2 installation

v4l-utils is a collection of various video4linux (V4L) and DVB utilities. These utilities are used to configure the Hauppauge video capture card.

The commands (CMD) are to be typed on a terminal window that you can open with **CTRL-ALT-T**

1. Checking if the v4l2 utilities are installed on ubuntu

   Type the CMD: **v4l2-ctl -D**



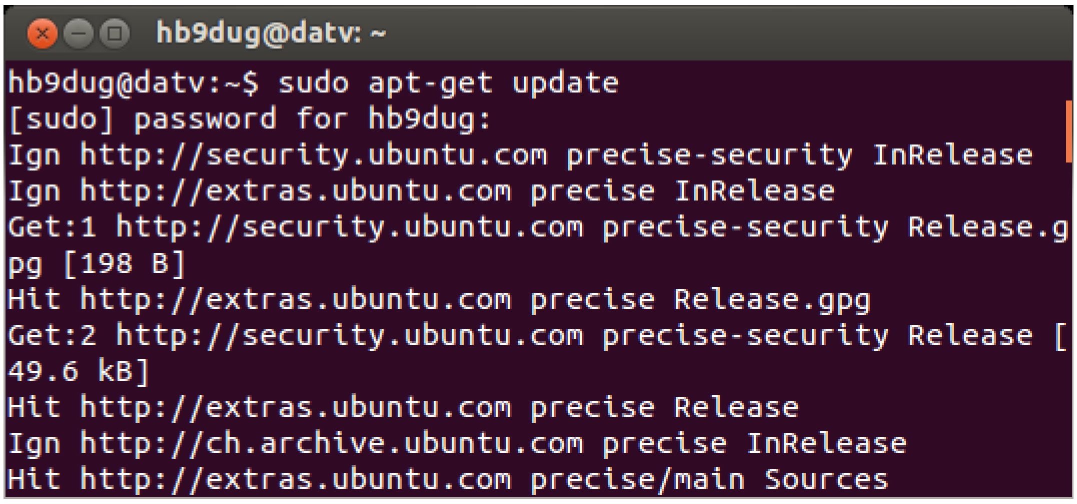
```
hb9dug@datv:~$ v4l2-ctl -D
The program 'v4l2-ctl' is currently not installed.  You can
 install it by typing:
sudo apt-get install v4l-utils
hb9dug@datv:~$
```

   If you get the the above message go to the next step otherwise jump to step 3.

2. Installing the v4l2 utilities

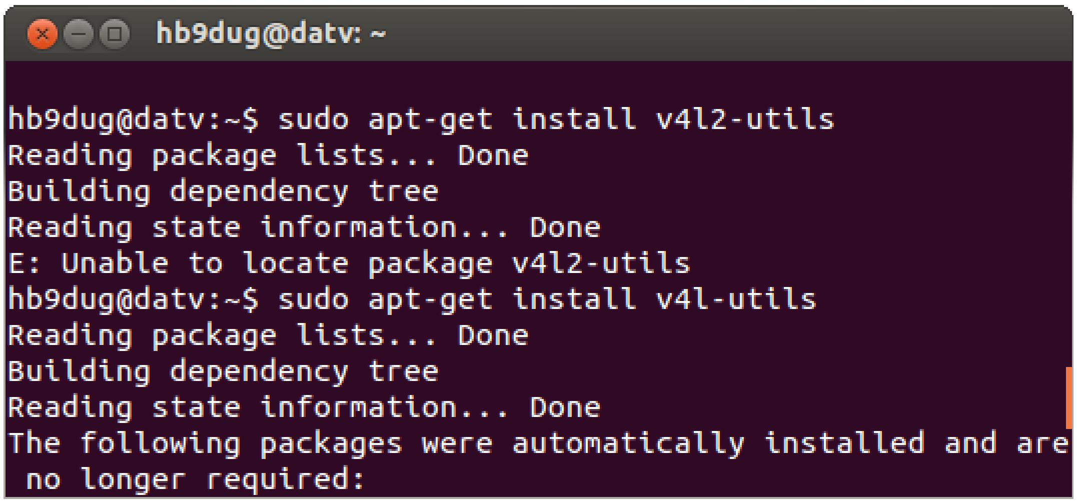
   First, we need to update the local package index with the latest changes made in

repositories available on the Internet with the CMD:
sudo apt-get update.



```
hb9dug@datv:~$ sudo apt-get update
[sudo] password for hb9dug:
Ign http://security.ubuntu.com precise-security InRelease
Ign http://extras.ubuntu.com precise InRelease
Get:1 http://security.ubuntu.com precise-security Release.g
pg [198 B]
Hit http://extras.ubuntu.com precise Release.gpg
Get:2 http://security.ubuntu.com precise-security Release [
49.6 kB]
Hit http://extras.ubuntu.com precise Release
Ign http://ch.archive.ubuntu.com precise InRelease
Hit http://extras.ubuntu.com precise/main Sources
```

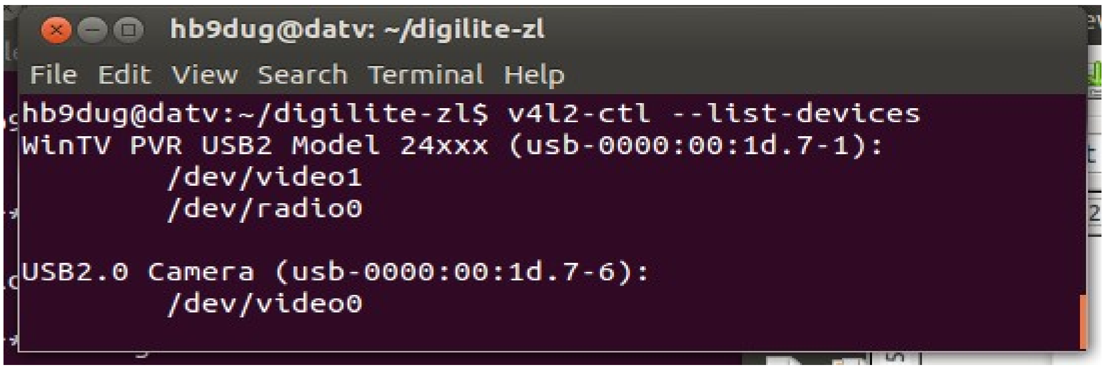
Then we download and install the v4l2 utilities with the CMD:
sudo apt-get install v4l-utils



```
hb9dug@datv:~$ sudo apt-get install v4l2-utils
Reading package lists... Done
Building dependency tree
Reading state information... Done
E: Unable to locate package v4l2-utils
hb9dug@datv:~$ sudo apt-get install v4l-utils
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are
 no longer required:
```

3. Checking that v4l2 utilities are installed

Run the CMD: **v4l2-ctl –list-devices**.


hb9dug@datv: ~/digilite-zl
File Edit View Search Terminal Help

```
hb9dug@datv:~/digilite-zl$ v4l2-ctl --list-devices
WinTV PVR USB2 Model 24xxx (usb-0000:00:1d.7-1):
        /dev/video1
        /dev/radio0

USB2.0 Camera (usb-0000:00:1d.7-6):
        /dev/video0
```

If you see your Hauppauge device, you have successfully installed the tools.

Please note the device name (in the screen dump above **video1**). You will need this information later on.

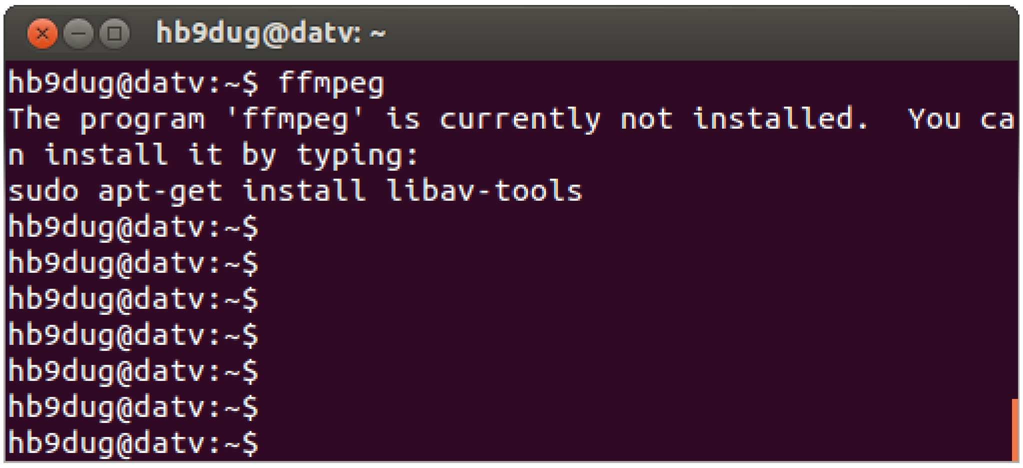
FFmpeg tools installation

FFmpeg is a complete, cross-platform solution to record, convert and stream audio and video.

The commands (CMD) are to be typed on a terminal window that you can open with **CTRL-ALT-T**

1. Checking that the ffmpeg tools are installed on ubuntu

   Type the CMD: **ffmpeg**

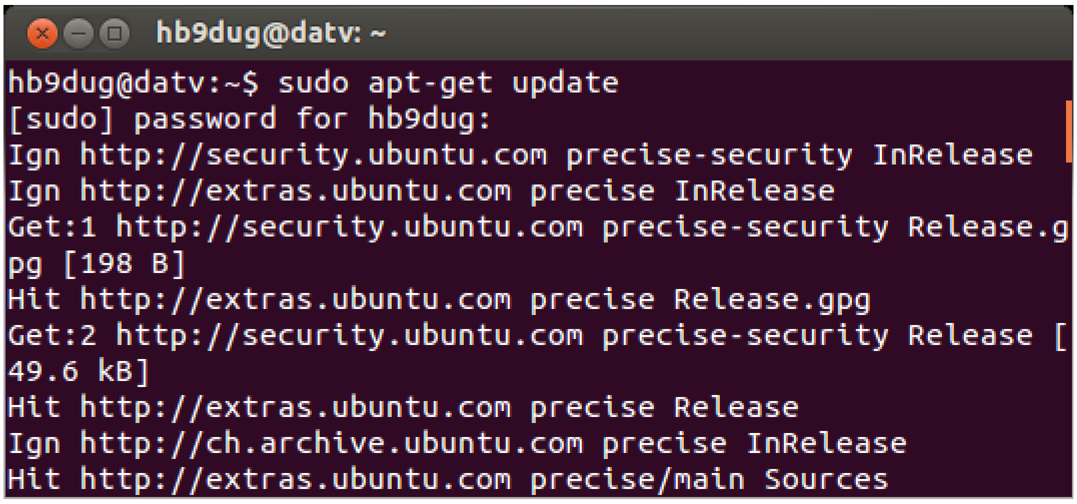


```
hb9dug@datv: ~
hb9dug@datv:~$ ffmpeg
The program 'ffmpeg' is currently not installed.  You ca
n install it by typing:
sudo apt-get install libav-tools
hb9dug@datv:~$
hb9dug@datv:~$
hb9dug@datv:~$
hb9dug@datv:~$
hb9dug@datv:~$
hb9dug@datv:~$
hb9dug@datv:~$
```

   If you get the the above message go to the next step otherwise jump to step 3.

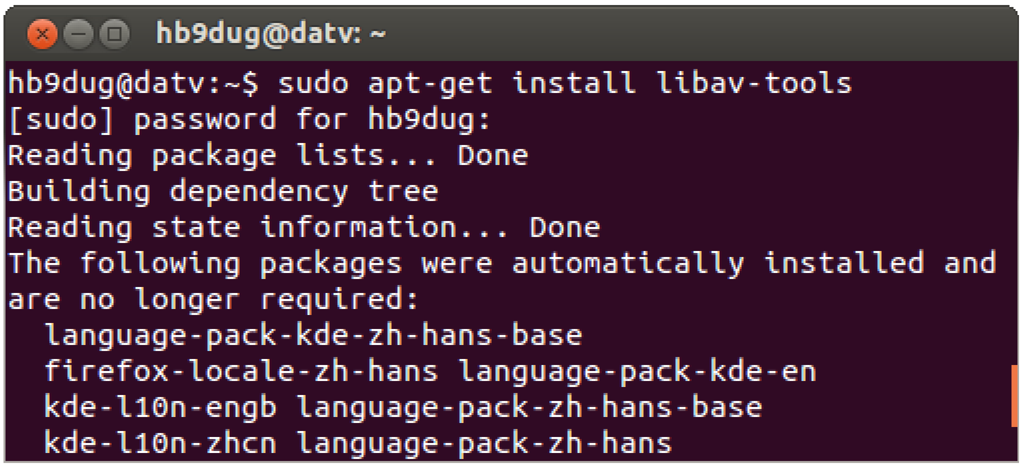
2. Installing the the ffmpeg tools

   First, we need to update the local package index with the latest changes made in repositories available on the Internet with the CMD: **sudo apt-get update.**



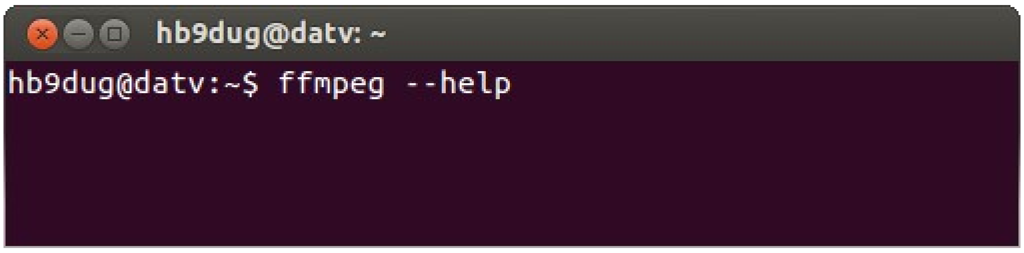
```
hb9dug@datv: ~
hb9dug@datv:~$ sudo apt-get update
[sudo] password for hb9dug:
Ign http://security.ubuntu.com precise-security InRelease
Ign http://extras.ubuntu.com precise InRelease
Get:1 http://security.ubuntu.com precise-security Release.g
pg [198 B]
Hit http://extras.ubuntu.com precise Release.gpg
Get:2 http://security.ubuntu.com precise-security Release [
49.6 kB]
Hit http://extras.ubuntu.com precise Release
Ign http://ch.archive.ubuntu.com precise InRelease
Hit http://extras.ubuntu.com precise/main Sources
```

Then we download and install the ffmpeg tools with the CMD:
sudo apt-get install libav-tools



```
hb9dug@datv:~$ sudo apt-get install libav-tools
[sudo] password for hb9dug:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and
are no longer required:
  language-pack-kde-zh-hans-base
  firefox-locale-zh-hans language-pack-kde-en
  kde-l10n-engb language-pack-zh-hans-base
  kde-l10n-zhcn language-pack-zh-hans
```

3. Checking that the ffmpeg tools are installed
   Just run the CMD: **ffmpeg --help**



```
hb9dug@datv:~$ ffmpeg --help
```

If you see a long list of the options available for ffmpeg then you have successfully installed the tools.

Installation OpenCaster 3.1.4 software

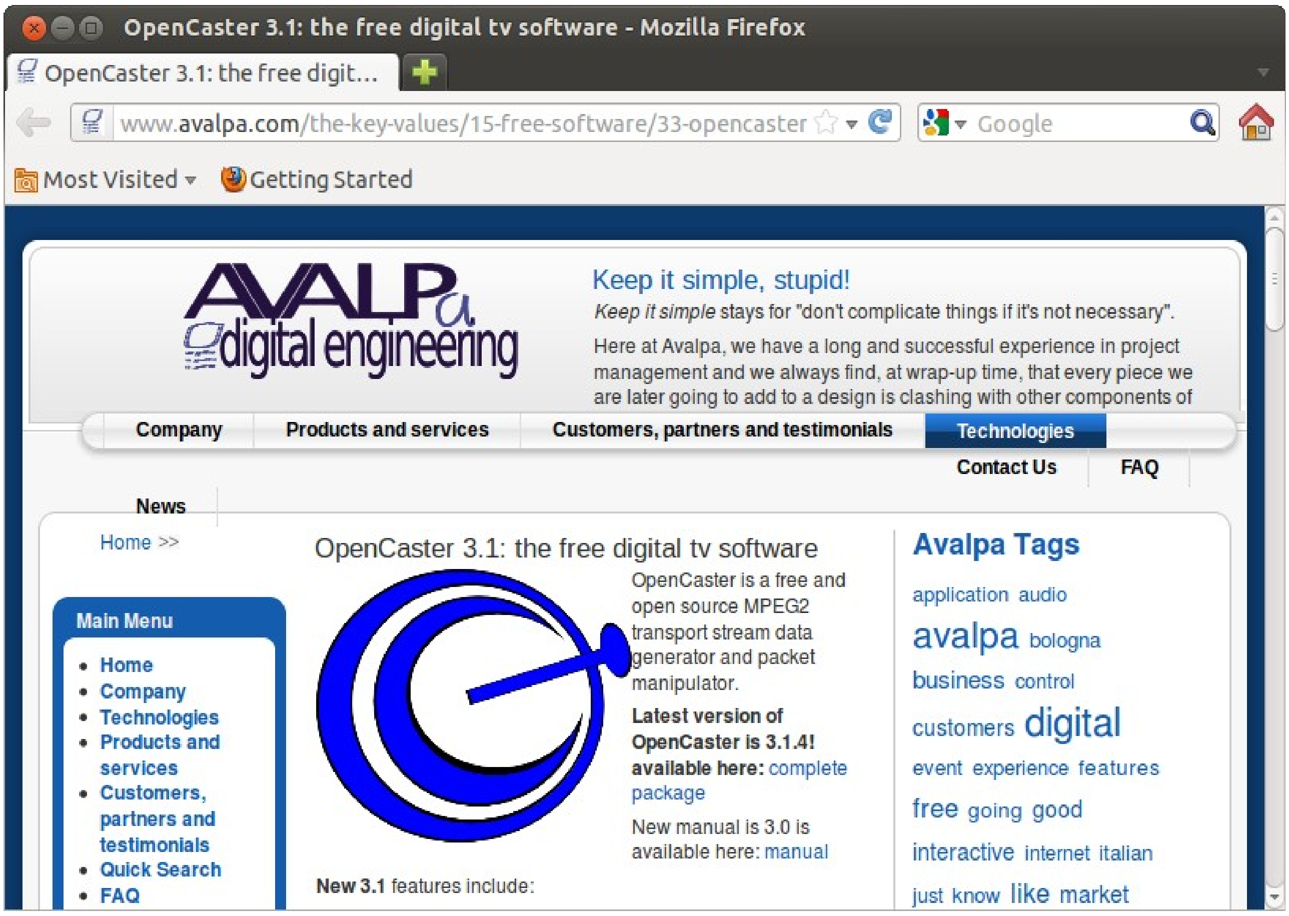
Opencaster is a free and open source MPEG2 transport stream data generator and packet manipulator.

The commands (CMD) are to be typed on a terminal window that you can open with **CTRL-ALT-T**

1. Downloading the software

   The software should be downloaded from the Avalpa web site at:

   http://www.avalpa.com/the-key-values/15-free-software/33-opencaster


OpenCaster 3.1: the free digital tv software - Mozilla Firefox
www.avalpa.com/the-key-values/15-free-software/33-opencaster
Company
Products and services
Customers, partners and testimonials
Technologies
Contact Us
FAQ
News
Main Menu
Avalpa Tags

OpenCaster 3.1: the free digital tv software

OpenCaster is a free and open source MPEG2 transport stream data generator and packet manipulator.

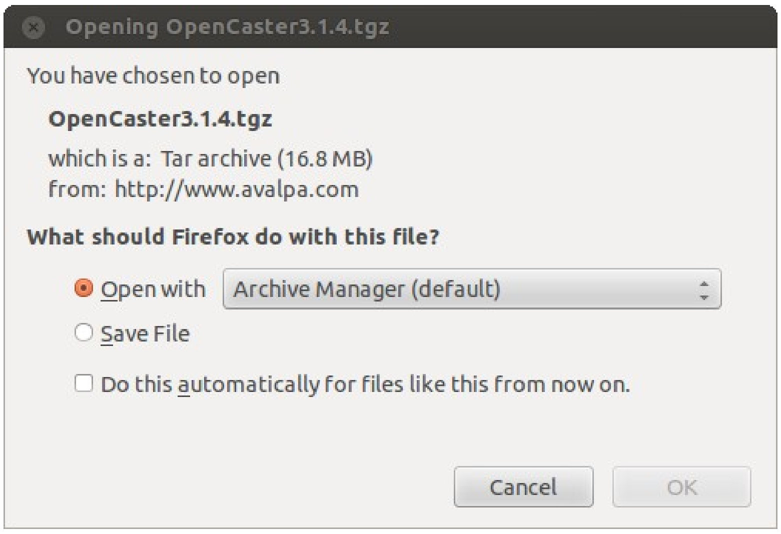
Latest version of OpenCaster is 3.1.4! available here: complete package

New manual is 3.0 is available here: manual

New 3.1 features include:

Once on the web site, click on "**complete package**" to download the software.

On the pop-up window from Firefox, select the option: **Open with Archive Manager (default)** and then click on **OK**


Opening OpenCaster3.1.4.tgz

You have chosen to open

OpenCaster3.1.4.tgz

which is a: Tar archive (16.8 MB)
from: http://www.avalpa.com

What should Firefox do with this file?

[✓ Open with] Archive Manager (default)
Save File
Do this automatically for files like this from now on.

On the pop-up window from the Archive Manager, just click on **Extract**. A directory will be created in the Home directory with the source files needed to compile OpenCaster.

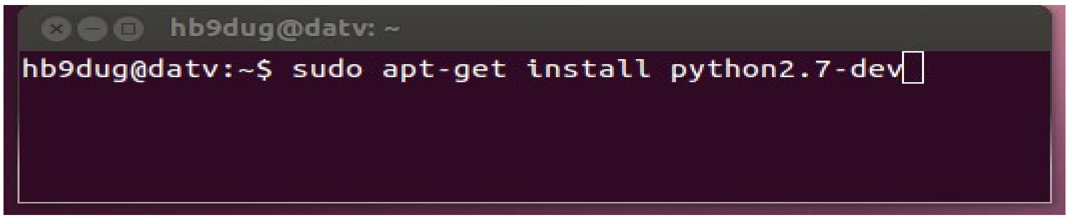
2. Compiling and installing

OpenCaster is downloaded in source code so it needs to be compiled into executable files before it can be used. It also needs other source code packages to be able to compile successfully.

With a fresh installation of Ubuntu 12.04 LTS, only the "python2.7-dev" package needs to be installed.

For more details consult the **INSTALL** file in the OpenCaster3.1.4 directory.

To install the "python2.7-dev" package, run in a terminal window the following CMD:
sudo apt-get update
sudo apt-get install python2.7-dev

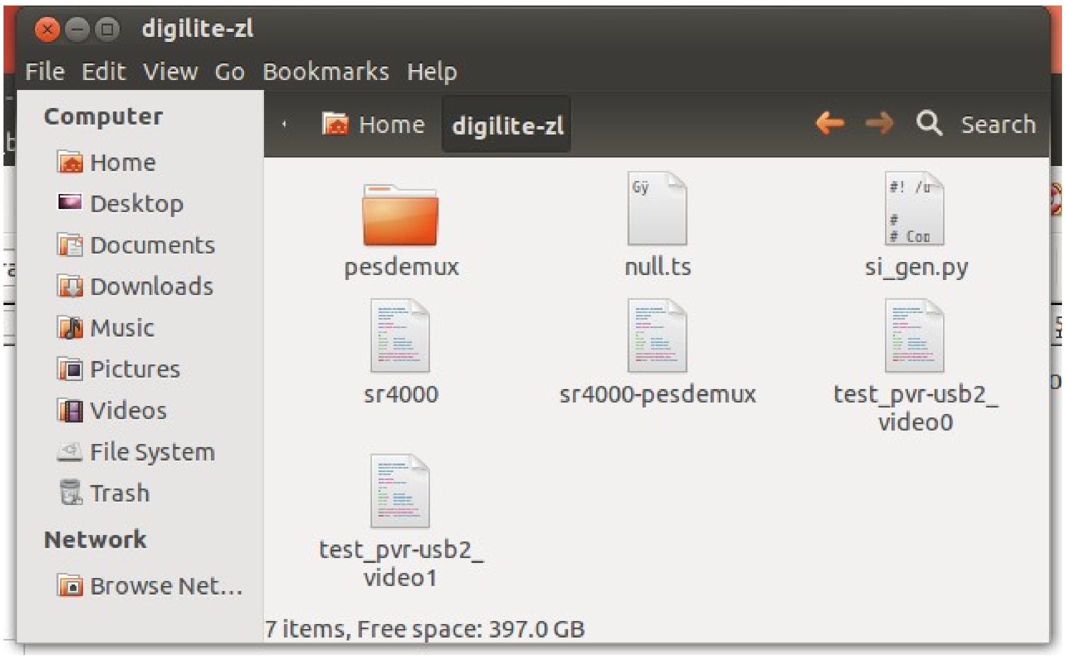


```
hb9dug@datv:~$ sudo apt-get install python2.7-dev
```

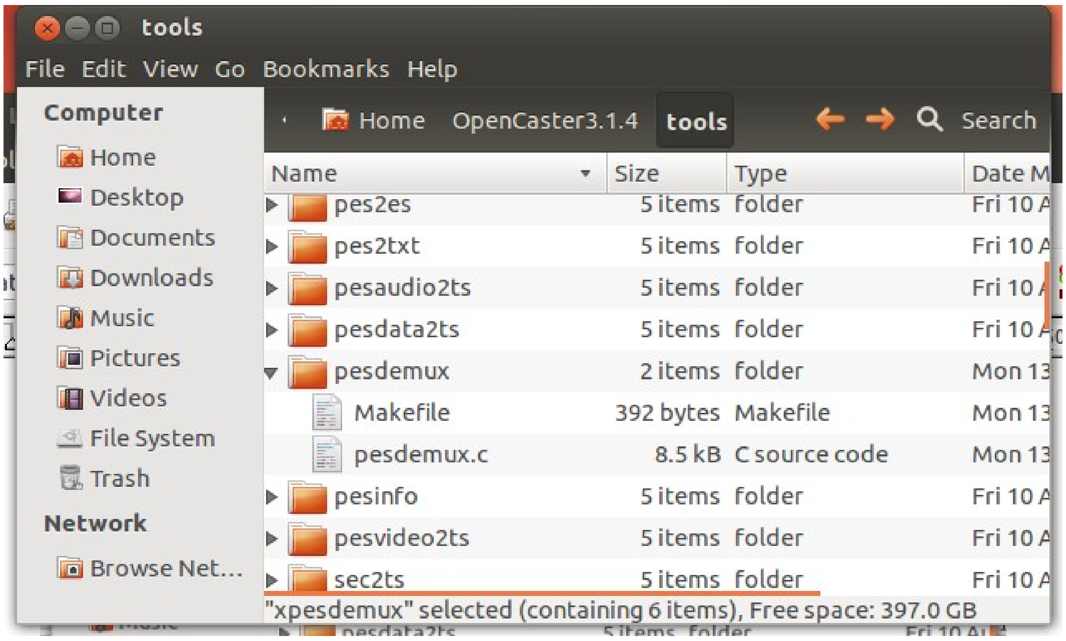
Lorenzo Pallara from Avalpa has kindly provided us with an additional OpenCaster tool called '**pesdemux**' to demultiplex the Program Stream coming from the Hauppauge card. The tool can replace ffmpeg command in the application script file (see the software section).

The source code for pesdemux is not included in the package you have downloaded. Before you can compile and install OpenCaster3.1.4, you need to copy the source code for it in the right place in the OpenCaster3.1.4 directory with the following procedure:

Step 1:
From the digilite-zl directory, copy the directory 'pesdemux' in the OpenCaster3.1.4/tools directory with the following CMD:
cp -r pesdemux $HOME/OpenCaster3.1.4/tools



If successful, you should see a directory called 'pesdemux' with 2 files in it.

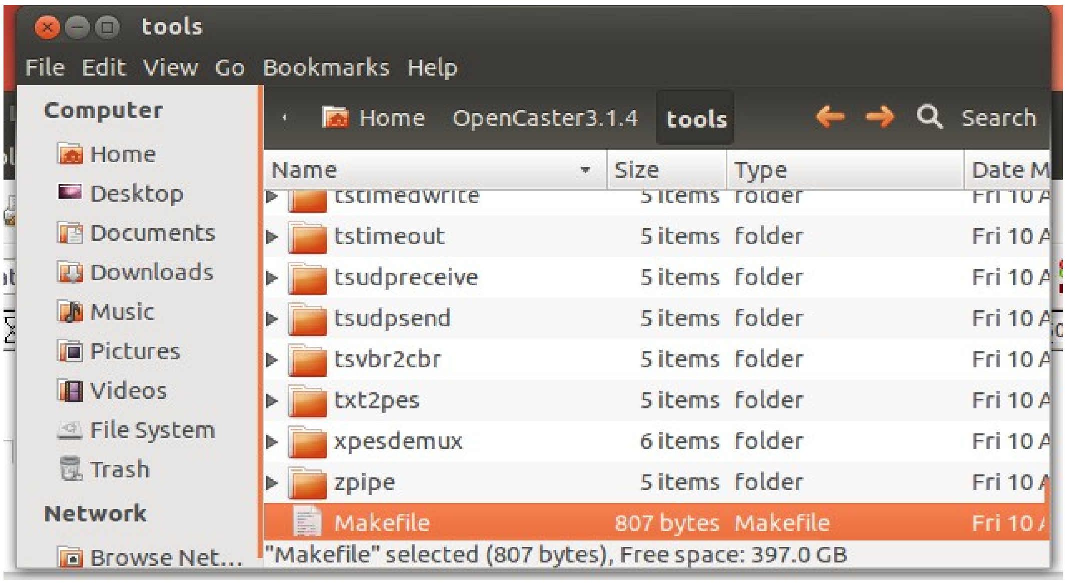


Name	Size	Type	Date M
pes2es	5 items	folder	Fri 10 A
pes2txt	5 items	folder	Fri 10 A
pesaudio2ts	5 items	folder	Fri 10 A
pesdata2ts	5 items	folder	Fri 10 A
pesdemux	2 items	folder	Mon 13
Makefile	392 bytes	Makefile	Mon 13
pesdemux.c	8.5 kB	C source code	Mon 13
pesinfo	5 items	folder	Fri 10 A
pesvideo2ts	5 items	folder	Fri 10 A
sec2ts	5 items	folder	Fri 10 A

"xpesdemux" selected (containing 6 items), Free space: 397.0 GB

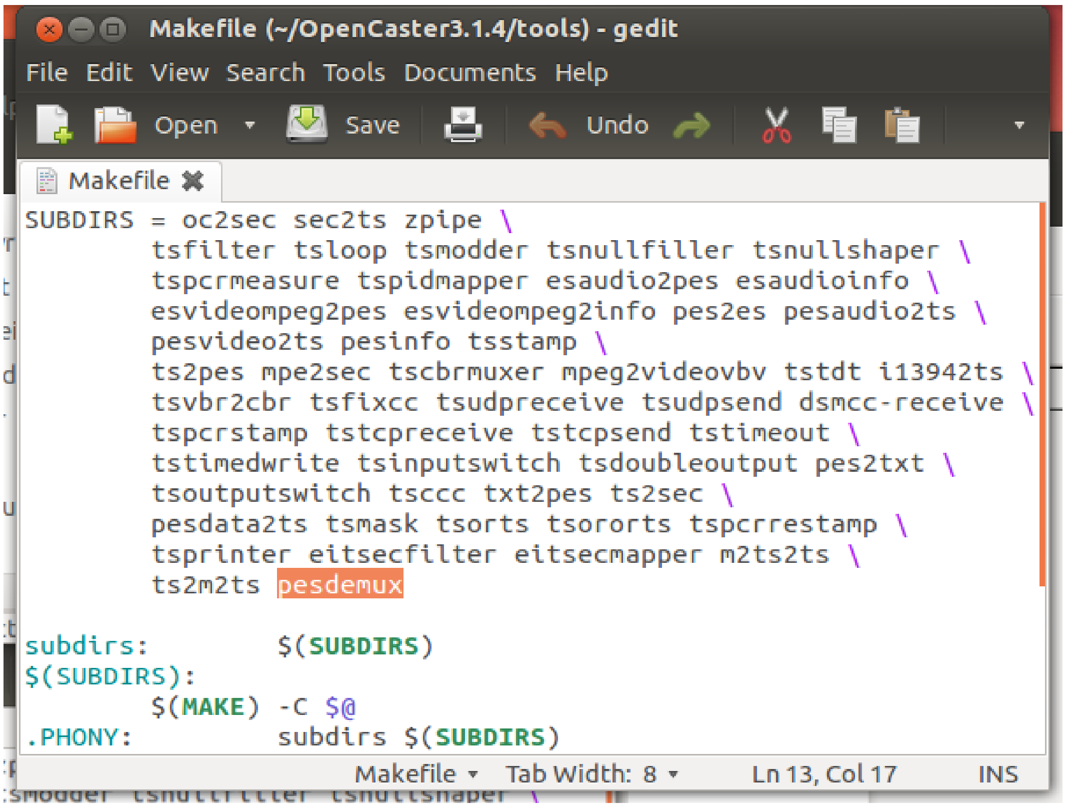
Step 2:
From the OpenCaster3.1.4/tools edits the 'MakeFile' file to add the subdirectory 'pesdemux' in the SUBDIRS variable with the following CMD:

- double-click on the file 'MakeFile'
- on the pop-up window click '**Display**'
- add the word 'pesdemux' at this end of the SUBDIRS variable
- click '**Save**'



Name	Size	Type	Date M
tstimedwrite	5 items	folder	Fri 10 A
tstimeout	5 items	folder	Fri 10 A
tsudpreceive	5 items	folder	Fri 10 A
tsudpsend	5 items	folder	Fri 10 A
tsvbr2cbr	5 items	folder	Fri 10 A
txt2pes	5 items	folder	Fri 10 A
xpesdemux	6 items	folder	Fri 10 A
zpipe	5 items	folder	Fri 10 A
Makefile	807 bytes	Makefile	Fri 10 A

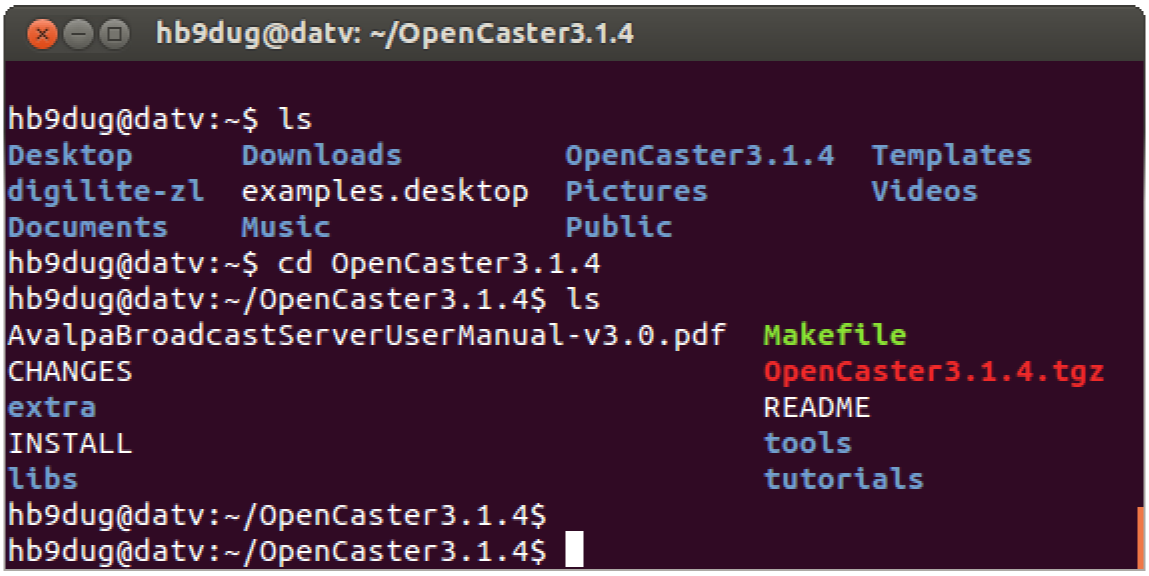
"Makefile" selected (807 bytes), Free space: 397.0 GB


Makefile (~/OpenCaster3.1.4/tools) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
Makefile
Makefile
Tab Width: 8
Ln 13, Col 17
INS

```
SUBDIRS = oc2sec sec2ts zpipe \
        tsfilter tsloop tsmodder tsnullfiller tsnullshaper \
        tspcrmeasure tspidmapper esaudio2pes esaudioinfo \
        esvideompeg2pes esvideompeg2info pes2es pesaudio2ts \
        pesvideo2ts pesinfo tsstamp \
        ts2pes mpe2sec tscbrmuxer mpeg2videovbv tstdt i13942ts \
        tsvbr2cbr tsfixcc tsudpreceive tsudpsend dsmcc-receive \
        tspcrstamp tstcpreceive tstcpsend tstimeout \
        tstimedwrite tsinputswitch tsdoubleoutput pes2txt \
        tsoutputswitch tsccc txt2pes ts2sec \
        pesdata2ts tsmask tsorts tsororts tspcrrestamp \
        tsprinter eitsecfilter eitsecmapper m2ts2ts \
        ts2m2ts pesdemux

subdirs:        $(SUBDIRS)
$(SUBDIRS):
        $(MAKE) -C $@
.PHONY:         subdirs $(SUBDIRS)
```

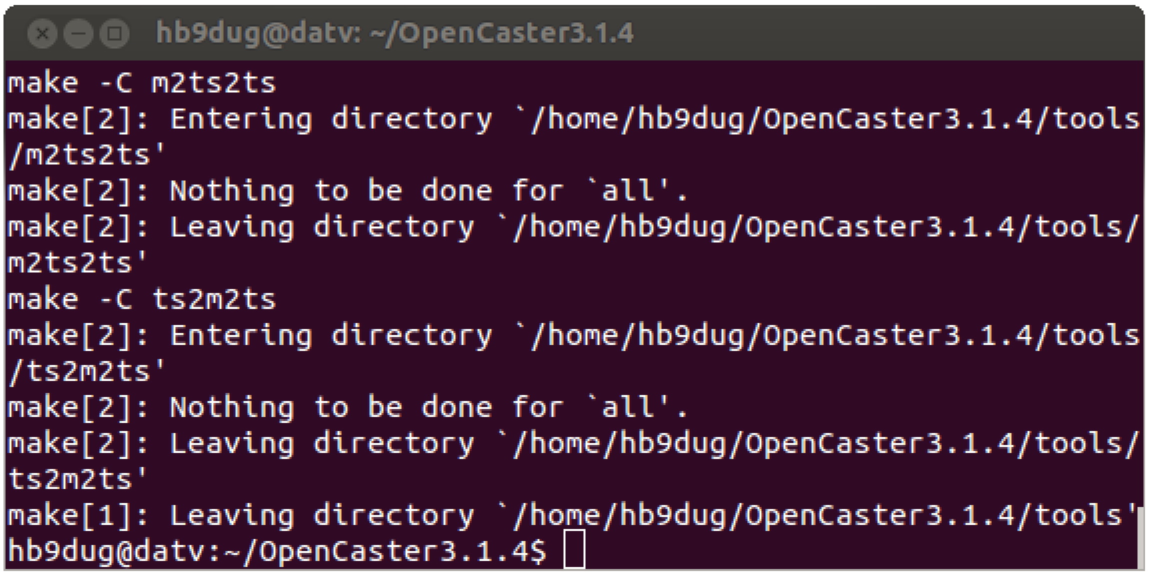
Now we are ready to compile OpenCaster3.1.4. For that, we need to move to its directory with the following CMD:

ls	(show directories and files)
cd OpenCaster3.1.4	(change the current working directory)
ls	



```
hb9dug@datv:~$ ls
Desktop      Downloads         OpenCaster3.1.4  Templates
digilite-zl  examples.desktop  Pictures         Videos
Documents    Music             Public
hb9dug@datv:~$ cd OpenCaster3.1.4
hb9dug@datv:~/OpenCaster3.1.4$ ls
AvalpaBroadcastServerUserManual-v3.0.pdf  Makefile
CHANGES                                   OpenCaster3.1.4.tgz
extra                                     README
INSTALL                                   tools
libs                                      tutorials
hb9dug@datv:~/OpenCaster3.1.4$
hb9dug@datv:~/OpenCaster3.1.4$
```

To compile OpenCaster, execute the following CMD: **make**



hb9dug@datv: ~/OpenCaster3.1.4

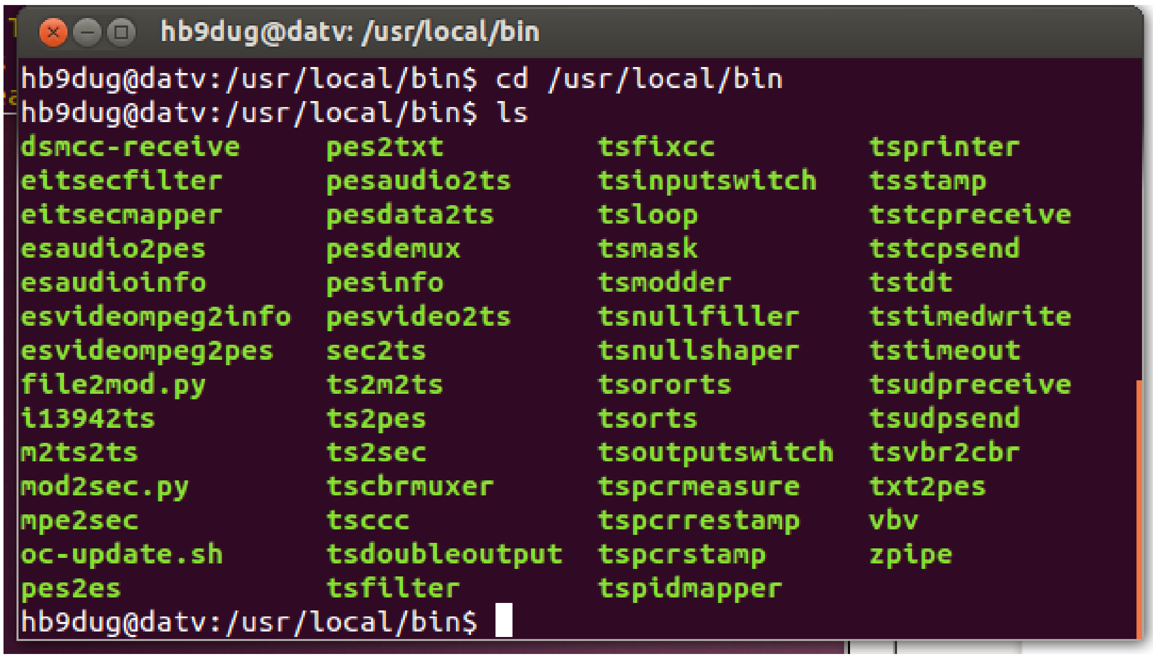
```
make -C m2ts2ts
make[2]: Entering directory `/home/hb9dug/OpenCaster3.1.4/tools
/m2ts2ts'
make[2]: Nothing to be done for `all'.
make[2]: Leaving directory `/home/hb9dug/OpenCaster3.1.4/tools/
m2ts2ts'
make -C ts2m2ts
make[2]: Entering directory `/home/hb9dug/OpenCaster3.1.4/tools
/ts2m2ts'
make[2]: Nothing to be done for `all'.
make[2]: Leaving directory `/home/hb9dug/OpenCaster3.1.4/tools/
ts2m2ts'
make[1]: Leaving directory `/home/hb9dug/OpenCaster3.1.4/tools'
hb9dug@datv:~/OpenCaster3.1.4$ 
```

A successful **make** CMD should report no error messages as above.

To install OpenCaster, execute the following CMD:
sudo make install

A successful **sudo make install** CMD should also report no error messages.

You can check that the programs have been installed in the following directory:
/usr/local/bin



hb9dug@datv: /usr/local/bin

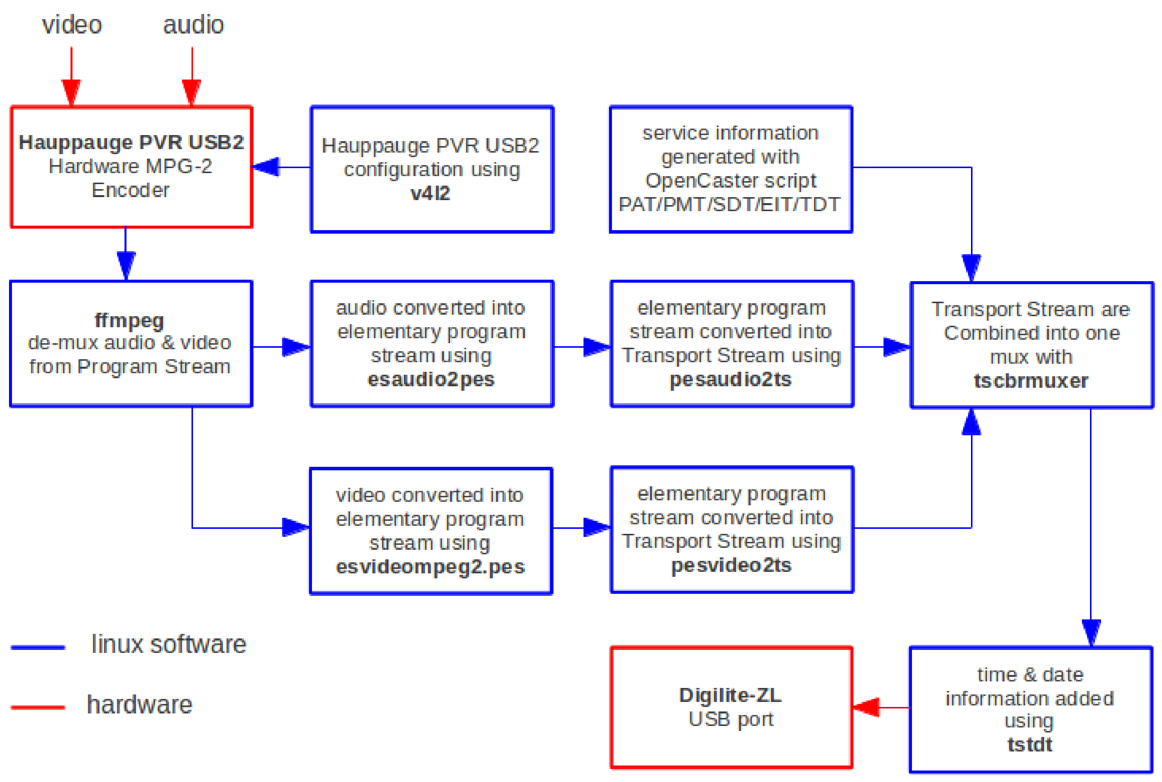
```
hb9dug@datv:/usr/local/bin$ cd /usr/local/bin
hb9dug@datv:/usr/local/bin$ ls
dsmcc-receive      pes2txt        tsfixcc         tsprinter
eitsecfilter       pesaudio2ts    tsinputswitch   tsstamp
eitsecmapper       pesdata2ts     tsloop          tstcpreceive
esaudio2pes        pesdemux       tsmask          tstcpsend
esaudioinfo        pesinfo        tsmodder        tstdt
esvideompeg2info   pesvideo2ts    tsnullfiller    tstimedwrite
esvideompeg2pes    sec2ts         tsnullshaper    tstimeout
file2mod.py        ts2m2ts        tsororts        tsudpreceive
i13942ts           ts2pes         tsorts          tsudpsend
m2ts2ts            ts2sec         tsoutputswitch  tsvbr2cbr
mod2sec.py         tscbrmuxer     tspcrmeasure    txt2pes
mpe2sec            tsccc          tspcrrestamp    vbv
oc-update.sh       tsdoubleoutput tspcrstamp      zpipe
pes2es             tsfilter       tspidmapper
hb9dug@datv:/usr/local/bin$ 
```

Check you see the programs listed below. They are needed by our application: esaudio2pes, esvideompeg2pes, pesaudio2ts, pesvideo2ts, tscbrmuxer, tstdt and pesdemux.

This end the installation of the softwares needed to produce a live TS stream to feed Digilite-ZL.

Software configuration

The configuration of the different modules will be done through a script file, which at time of execution, will send a live TS to the Digilite-ZL module.


video
audio
Hauppauge PVR USB2 Hardware MPG-2 Encoder
Hauppauge PVR USB2 configuration using v4l2
service information generated with OpenCaster script PAT/PMT/SDT/EIT/TDT
ffmpeg de-mux audio & video from Program Stream
audio converted into elementary program stream using esaudio2pes
elementary program stream converted into Transport Stream using pesaudio2ts
Transport Stream are Combined into one mux with tscbrmuxer
video converted into elementary program stream using esvideompeg2.pes
elementary program stream converted into Transport Stream using pesvideo2ts
linux software
hardware
Digilite-ZL USB port
time & date information added using tstdt

The block diagram below shows the interaction between the different modules.

Presently the solution has been tested with a clean installation of Ubuntu version 12.04 LTS, v4l-utils version 0.8.6-1ubuntu2, FFmpeg version 0.8.3-4:0.8.3-0ubuntu0.12.04.1 and OpenCaster3.1.4. If you use others softwares version and issues arise, you should revert to the versions above.

The script file is made of different modules as illustrated in the above block diagram. We will describe briefly each of the modules.

Hauppauge PVR USB2 configuration block

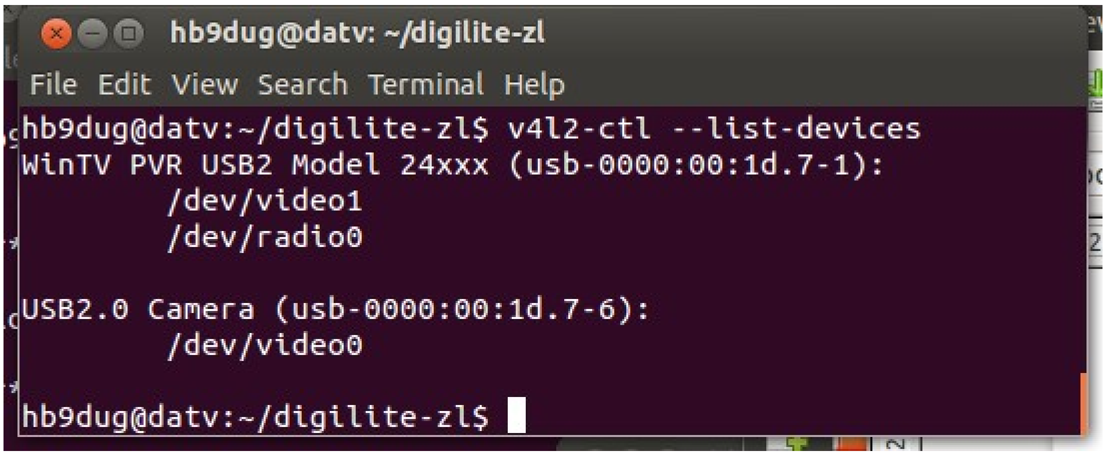
To configure the card, we use v4l2 utilities to set:

- audio bit-rate (x kbps), audio mode (stereo)
- stream type (MPEG-2 Program Stream)
- video bit-rate (x kbps)
- video bit-rate mode (constant bit rate)
- video aspect ratio (4:3)
- resolution (720x576)
- card input (composite, s-video)

This translate in the following v4l2 commands:

```
v4l2-ctl -d /dev/videoX -c audio_layer_ii_bitrate=9          # 192Kbps
v4l2-ctl -d /dev/videoX -c audio_stereo_mode=0               # stereo
v4l2-ctl -d /dev/videoX -c stream_type=0                     # MPEG-2 Program Stream=0
v4l2-ctl -d /dev/videoX -c video_bitrate=2300000             #
v4l2-ctl -d /dev/videoX -c video_bitrate_mode=1              # Constant Bit-rate = 1
v4l2-ctl -d /dev/videoX -c video_aspect=1                    # 4:3
v4l2-ctl -d /dev/video1  - -set-fmt-video=width=720,height=576
v4l2-ctl -d /dev/video1 - -set-input=1                       # composite = 1
```

Please note that you should adjust the "**X**" of the video device according to the video device you see for your USB Hauppauge card when you run the CMD: **v4l2-ctl –list-devices**.


hb9dug@datv: ~/digilite-zl
File Edit View Search Terminal Help

```
hb9dug@datv:~/digilite-zl$ v4l2-ctl --list-devices
WinTV PVR USB2 Model 24xxx (usb-0000:00:1d.7-1):
        /dev/video1
        /dev/radio0

USB2.0 Camera (usb-0000:00:1d.7-6):
        /dev/video0

hb9dug@datv:~/digilite-zl$
```

ffmpeg block or pesdemux

Ffmpeg split the MPEG-2 Program Stream coming from the Hauppauge card into separate elementary streams, re-encode audio to lower bitrate (128k), stream the results in an audio buffer (audio.es) and a video buffer (video.es).

This is done with the CMD:

```
ffmpeg -v -10 -i /dev/video1 -acodec mp2 -ab 128k -f mp2 -y /tmp/audio.es -vcodec copy -f mpeg2video -y /tmp/video.es
```

If you went more details on ffmpeg, run the CMD: **man ffmpeg**

The CMD provided by Avalpa does the same but leaving the audio bitrate at 192 kbs:

```
pesdemux /dev/video1 /tmp/audio.es  /tmp/video.es
```

Esaudio2pes and esvideompeg2pes blocks

esaudio2pes and esvideompeg2pes convert the elementary streams into program streams. The results are streamed in an audio buffer (audio.pes) and video buffer (video.pes).

This is done with the CMD:

esaudio2pes /tmp/audio.es 1152 48000 384 0 > /tmp/audio.pes

esvideompeg2pes /tmp/video.es > /tmp/video.pes

If you went more details, please consult the Avalpa Broadcast Server Manual.

pesaudio2ts and pesvideo2ts blocks

pesaudio2ts and pesvideo2t convert the program streams into transport program streams. PIDs are added, well as bit-rates for the audio and video. The results are streamed in an audio buffer (audio.ts) and video buffer (video.ts).

This is done with the CMD:

pesaudio2ts 257 1152 48000 384 0 /tmp/audio.pes > /tmp/audio.ts

pesvideo2ts 256 25 64 3200000 0 /tmp/video.pes > /tmp/video.ts

If you went more details, please consult the Avalpa Broadcast Server Manual.

Service information generation block

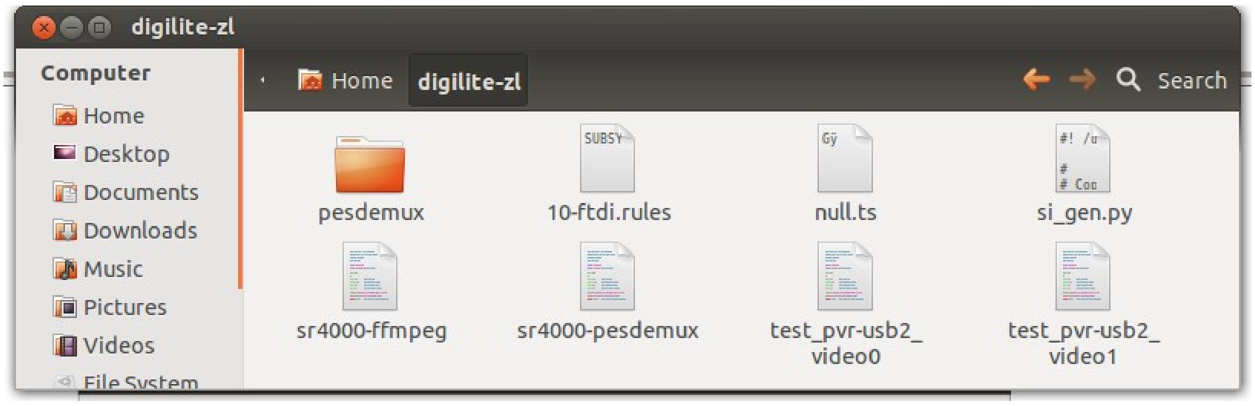
With the previous steps, we have generated blank transport streams to which we need to add the service information tables to make it into a DVB transport stream.

This process is done with OpenCaster suite using a script. The script file will generate separate .ts files with each of the parts of the DVB stream data. The files are PAT.ts, PMT.ts, SDT.ts, NIT.ts, EIT.ts, TDT.ts

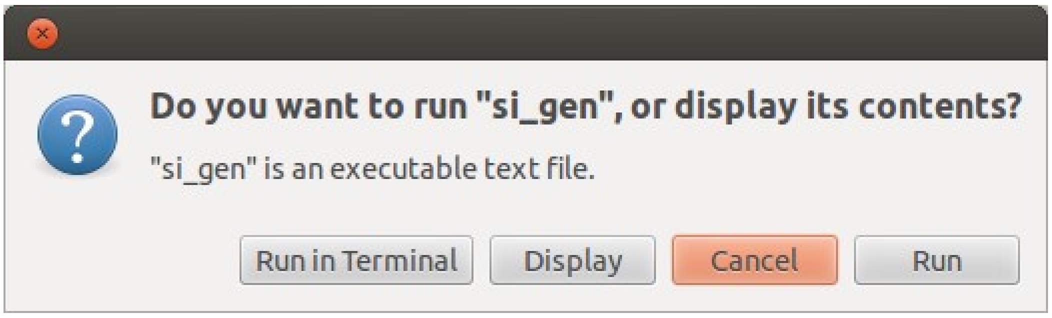
You can change this script to add your own details like Channel name, Provider name and set the current program information for a channel.

To change the parameters, follow the procedure below:

Download le zip file: **config_files.zip** from the Digilite-ZL web site and unzip it in /home/username/digilite-zl. You should see the following files:



Open the script file **si_gen** by double-clicking on it.

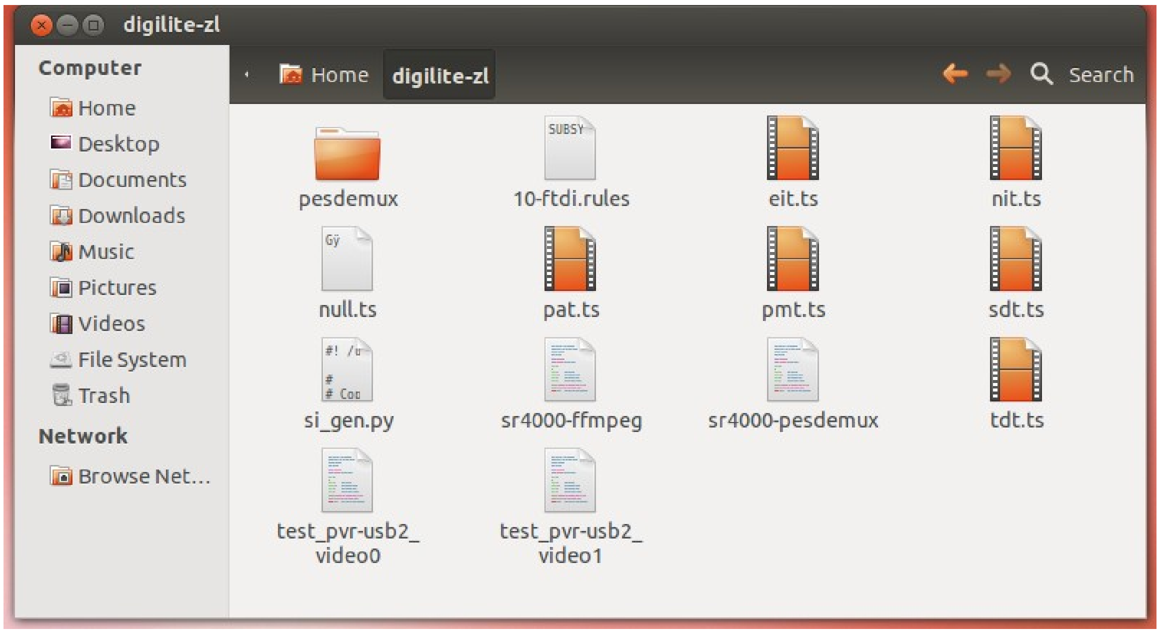


Do you want to run "si_gen", or display its contents?

"si_gen" is an executable text file.

Confirm editing by selecting “**Display**” on the above window.
Now you can modify the network_name, service_provider_name and service_name.
Click “**Save**” to save and close the file.

Once you have done your changes, you need to run the script by double-clicking on the file and confirm running the script by selecting “**Run in Terminal**”
On the /home/username/digilite-zl, now you should see the following additional TS files.



If you went more details, please consult the Avalpa Broadcast Server Manual.

Maybe for the first time you try this process, just run the script and leave the editing for the next time...

tscbrmuxer and tstdt blocks

The final step is to generate the DVB Transport Stream by muxing together the audio, video and service information transport streams with the CMD **tscbrmuxer**. and finally insert the current date and time with the CMD **tstdt**.

The important parameter to truly control is the total bit-rate of the TS. Today the FPGA code support a Symbol Rate of 4.000 MS/s with a FEC ½ this give a total available binary rate 3'686 kbit/s.

So the sum of all the different TS bit-rate should below the available bit-rate for the SR at 4.000/FEC ½.

This will depend mainly of the bit-rate attributed to the video and audio streams. The difference to total bit-rate should be filled with the null.ts bit-rate. See the table below.

TS file	kbit/s
video.ts	3'200'000
audio.ts	188'000
pat.ts	4'000
prnt.ts	4'000
sdt.ts	4'000
nit.ts	4'000
eit.ts	4'000
tdt.ts	4'000
null.ts	274'275
total	**3'686'275**

Finally, we insert the current date and time in the TS with the CMD **tstdt** and send the TS throughout the /dev/ttyUSB0 to the Digilite-ZL USB port.

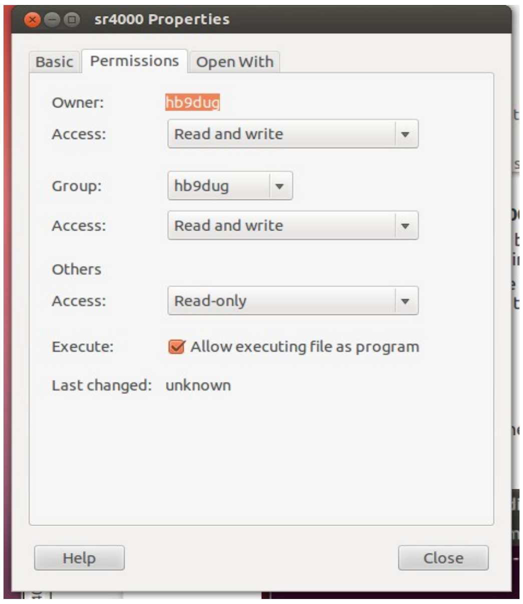
You will find a table summarizing the bit-rate available for a given SR and FEC on the F4DAY web site.

sr4000-ffmpeg script file

You will find the script file **sr4000-ffmpeg** in the zip file: **config_files.zip.** This script uses the cmd ffmpeg.

The script file integrates all the blocks enumerated above. The processes (programs) communicate between them using pipes (fifo buffers) created with the CMD **mkfifo.**

When we stop the script file, we do a little bit of cleanup by killing all the processes with the CMD **killall** and we remove the temporary pipes with the CMD **rm.**



Owner: hb9dug
Access: Read and write
Group: hb9dug
Access: Read and write
Others
Access: Read-only
Execute: ✓ Allow executing file as program
Last changed: unknown

Before you can execute a script file, you need to make it executable.

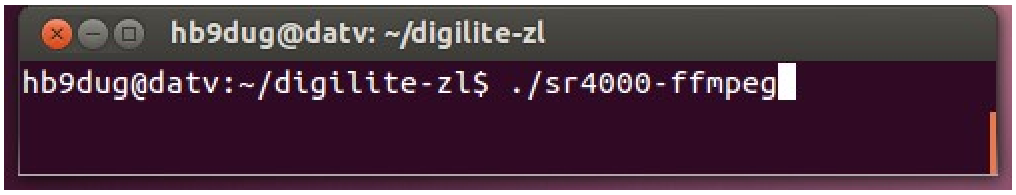
For that, open the digilite-zl directory and right-click on either sr4000-ffmpeg, test_pvr-usb2_video0 or test_pvr-usb2_video1.

On the pull-down menu, select **Properties** and on the TAB **Permissions**, tick **Execute** and **Close**.

To run a script, from the directory which contains the script, type:

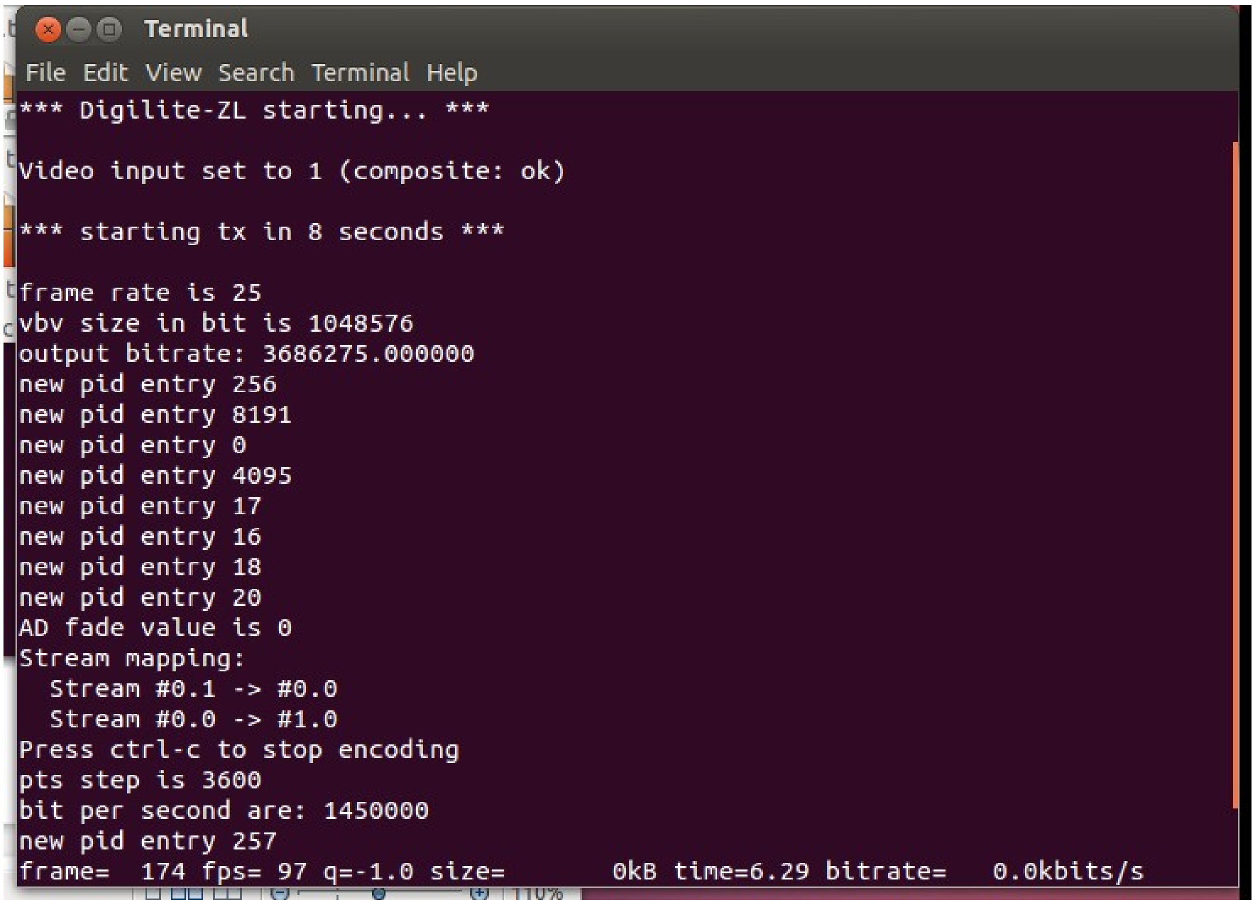
./ <script_name>

To run the script file, move to the directory /home/username/digilite-zl and type the CMD: **./sr4000-ffmpeg**



```
hb9dug@datv: ~/digilite-zl
hb9dug@datv:~/digilite-zl$ ./sr4000-ffmpeg
```

If everything goes well as it should... you will see a terminal window like below:



```
*** Digilite-ZL starting... ***

Video input set to 1 (composite: ok)

*** starting tx in 8 seconds ***

frame rate is 25
vbv size in bit is 1048576
output bitrate: 3686275.000000
new pid entry 256
new pid entry 8191
new pid entry 0
new pid entry 4095
new pid entry 17
new pid entry 16
new pid entry 18
new pid entry 20
AD fade value is 0
Stream mapping:
  Stream #0.1 -> #0.0
  Stream #0.0 -> #1.0
Press ctrl-c to stop encoding
pts step is 3600
bit per second are: 1450000
new pid entry 257
frame=  174 fps= 97 q=-1.0 size=       0kB time=6.29 bitrate=   0.0kbits/s
```

On the Digilite-ZL DE0-nano board:

- when the USB cable is connected, the LED0 flash at a one second rate to tell you that Digilite-ZL is ready and waiting for a transport stream.
- When LED1 is steady on, Digilite-ZL confirms receiving a transport stream at the correct SR/FEC and modulating.

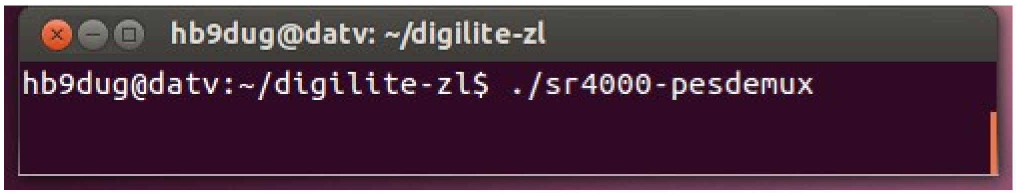
Now, you can connect your DVB-S Satellite Tuner and see your first Digilite-ZL live transmission!

sr4000-pesdemux script file

You will find the script file **sr4000-pesdemux** in the zip file: **config_files.zip.** This script uses the cmd pesdemux. From OpenCaster3.1.4.

To run the script file, move to the directory /home/username/digilite-zl and type the CMD:

./sr4000-ffmpeg



```
hb9dug@datv:~/digilite-zl$ ./sr4000-pesdemux
```

How to allow non-root access to /ttyUSB0

In the standard ubuntu configuration, only root has access to the USB interface so when you will execute the script sr4000-ffmpeg or sr4000-pesdemux, you will get an error message like "permission denied".

The quick fix, you run the script with root privilege: **sudo ./sr4000-ffmpeg or ./sr4000-pesdemux**

The correct fix... Copy the file **10-ftdi.rules** in the directory **/etc/udev/rules.d**. Move to the digilite-zl directory and execute in a terminal window the following CMD:

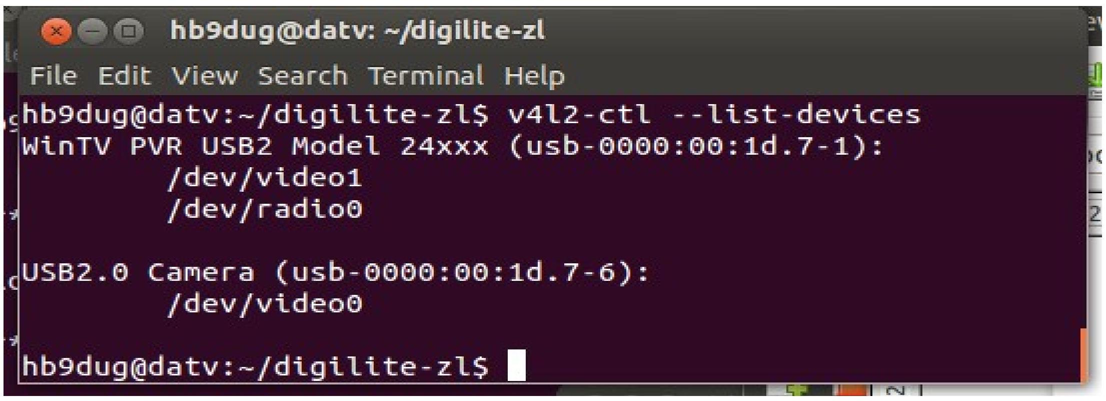
sudo cp 10-ftdi.rules /etc/udev/rules.d
sudo chmod 644 /etc/udev//rules.d/10-ftdi.rules

and restart your computer.

Intermediate tests, just in case...

If you face issues getting out your transport stream to the Digilite-ZL hardware, you should first check that the Hauppauge card is running correctly.

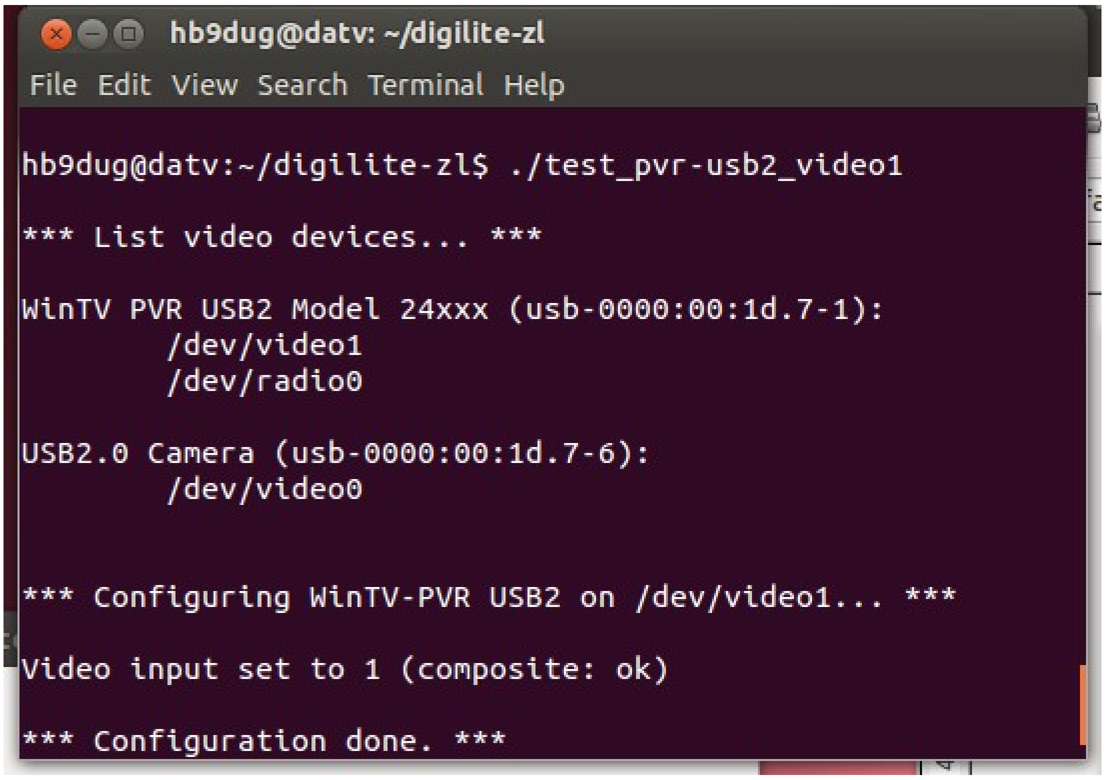
1. Check the video device name for your Hauppauge card with the CMD:
   v4l2-ctl --list-devices


hb9dug@datv: ~/digilite-zl
File Edit View Search Terminal Help

```
hb9dug@datv:~/digilite-zl$ v4l2-ctl --list-devices
WinTV PVR USB2 Model 24xxx (usb-0000:00:1d.7-1):
        /dev/video1
        /dev/radio0

USB2.0 Camera (usb-0000:00:1d.7-6):
        /dev/video0

hb9dug@datv:~/digilite-zl$
```

2. Run the script file **test_pvr-usb2_video0** or **test_pvr-usb2_video1** according to the video device name for your card.

   This script will only configure the Hauppauge card for the video composite input.

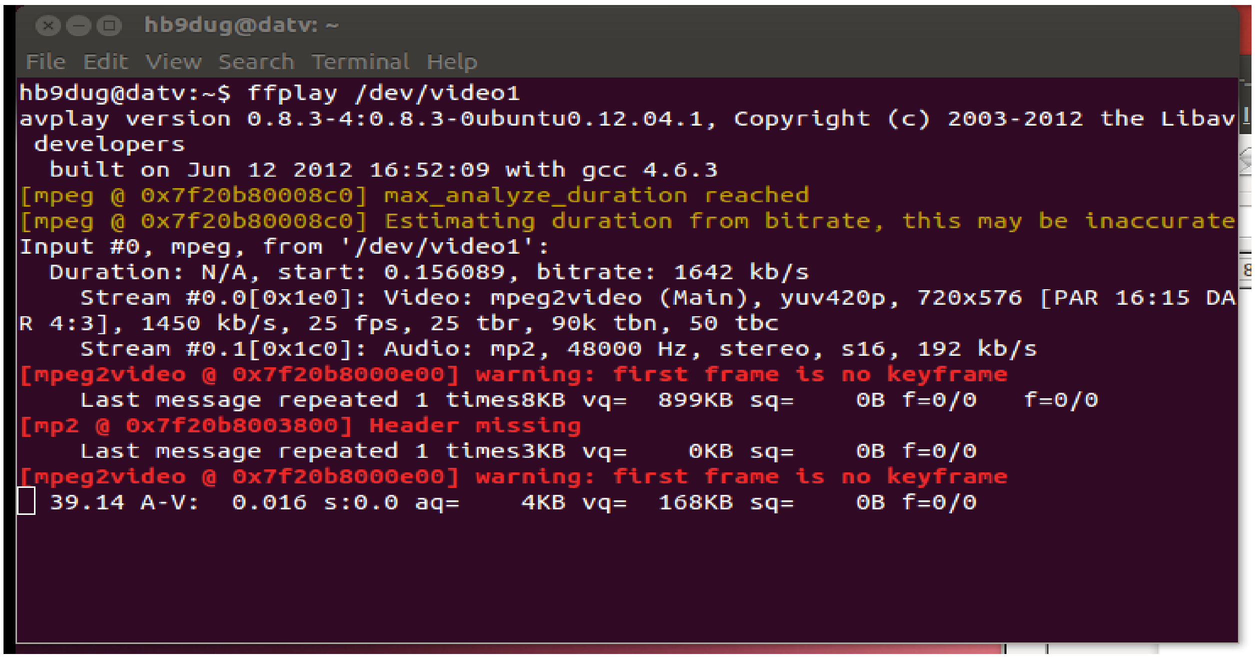


hb9dug@datv: ~/digilite-zl
File Edit View Search Terminal Help

```
hb9dug@datv:~/digilite-zl$ ./test_pvr-usb2_video1

*** List video devices... ***

WinTV PVR USB2 Model 24xxx (usb-0000:00:1d.7-1):
        /dev/video1
        /dev/radio0

USB2.0 Camera (usb-0000:00:1d.7-6):
        /dev/video0


*** Configuring WinTV-PVR USB2 on /dev/video1... ***

Video input set to 1 (composite: ok)

*** Configuration done. ***
```

   Finally, run the CMD: **ffplay /dev/video0** or **ffplay /dev/video1** in a terminal window.

   The window will show you the parameters configured for the card and in a second window you will see the images of your camera as well as the sound.



```
hb9dug@datv:~$ ffplay /dev/video1
avplay version 0.8.3-4:0.8.3-0ubuntu0.12.04.1, Copyright (c) 2003-2012 the Libav
 developers
  built on Jun 12 2012 16:52:09 with gcc 4.6.3
[mpeg @ 0x7f20b80008c0] max_analyze_duration reached
[mpeg @ 0x7f20b80008c0] Estimating duration from bitrate, this may be inaccurate
Input #0, mpeg, from '/dev/video1':
  Duration: N/A, start: 0.156089, bitrate: 1642 kb/s
    Stream #0.0[0x1e0]: Video: mpeg2video (Main), yuv420p, 720x576 [PAR 16:15 DA
R 4:3], 1450 kb/s, 25 fps, 25 tbr, 90k tbn, 50 tbc
    Stream #0.1[0x1c0]: Audio: mp2, 48000 Hz, stereo, s16, 192 kb/s
[mpeg2video @ 0x7f20b8000e00] warning: first frame is no keyframe
    Last message repeated 1 times8KB vq=  899KB sq=    0B f=0/0   f=0/0
[mp2 @ 0x7f20b8003800] Header missing
    Last message repeated 1 times3KB vq=    0KB sq=    0B f=0/0
[mpeg2video @ 0x7f20b8000e00] warning: first frame is no keyframe
  39.14 A-V:  0.016 s:0.0 aq=    4KB vq=  168KB sq=    0B f=0/0
```

Credits and disclaimers...

The solution described in this document and implemented is totally based on the work of Christian F1FAU, Rob M0DTS and Charles G4GUO. Without theirs in deep descriptions on generating a live MPEG-2 transport stream with Linux tools and Hauppauge hardware, I wouldn't have succeeded in this project.

Please note that the author is not a Linux expert and is therefore asking for your understanding if the description and script files are very basic.

The author is a member of the swissATV, a swiss association dedicated to ham television and HB9TV who operates a network of ATV/DATV relays.

HB9DUG Michel

hb9dug@vtxnet.ch