HOWTO: Get An Avermedia DVB-T working under Linux

Table of Contents

Assumptions and Introduction

It is assumed that the reader understands the basic structure of the Linux Kernel DVB drivers and the general principles of Digital TV.

One significant difference between Digital TV and Analogue TV that the unwary (like myself) should consider is that, although the component structure of budget DVB-T cards are substantially similar to Analogue TV cards, they function in substantially different ways.

The purpose of an Analogue TV is to receive and display an Analogue Television signal. An Analogue TV signal (otherwise known as composite video) is an analogue encoding of a sequence of image frames (25 per second) rasterised using an interlacing technique. Interlacing takes two fields to represent one frame. Computers today are at their best when dealing with digital signals, not analogue signals and a composite video signal is about as far removed from a digital data stream as you can get. Therefore, an Analogue TV card for a PC has the following purpose:

* Tune the receiver to receive a broadcast signal
* demodulate the broadcast signal
* demultiplex the analogue video signal and analogue audio signal (note some countries employ a digital audio signal embedded within the modulated composite analogue signal - NICAM.)
* digitize the analogue video signal and make the resulting datastream available to the data bus.

The digital datastream from an Analogue TV card is generated by circuitry on the card and is often presented uncompressed. For a PAL TV signal encoded at a resolution of 768x576 24-bit color pixels over 25 frames per second - a fair amount of data is generated and must be processed by the PC before it can be displayed on the video monitor screen. Some Analogue TV cards for PCs have onboard MPEG2 encoders which permit the raw digital data stream to be presented to the PC in an encoded and compressed form - similar to the form that is used in Digital TV.

The purpose of a simple budget digital TV card (DVB-T,C or S) is to simply:

* Tune the received to receive a broadcast signal.
* Extract the encoded digital datastream from the broadcast signal.
* Make the encoded digital datastream (MPEG2) available to the data bus.

The significant difference between the two is that the tuner on the analogue TV card spits out an Analogue signal, whereas the tuner on the digital TV card spits out a compressed encoded digital datastream. As the signal is already digitised, it is trivial to pass this datastream to the PC databus with minimal additional processing and then extract the digital video and audio datastreams passing them to the appropriate software or hardware for decoding and viewing.

The Avermedia DVB-T

The Avermedia DVB-T is a budget PCI DVB card. It has 3 inputs:

* RF Tuner Input
* Composite Video Input (RCA Jack)
* SVIDEO Input (Mini-DIN)

The RF Tuner Input is the input to the tuner module of the card. The Tuner is otherwise known as the "Frontend". The Frontend of the Avermedia DVB-T is a Microtune 7202D. A timely post to the linux-dvb mailing list ascertained that the Microtune 7202D is supported by the sp887x driver which is found in the dvb-hw CVS module.

The DVB-T card is based around the BT878 chip which is a very common multimedia bridge and often found on Analogue TV cards. There is no on-board MPEG2 decoder, which means that all MPEG2 decoding must be done in software, or if you have one, on an MPEG2 hardware decoding card or chipset.

Getting the card going

In order to fire up the card, it is necessary to load a number of modules from the DVB driver set. Prior to this it will have been necessary to download these drivers from the linuxtv CVS server and compile them successfully.

Depending on the card's feature set, the Device Driver API for DVB under Linux will expose some of the following device files in the /dev tree:

* /dev/dvb/adapter0/audio0
* /dev/dvb/adapter0/ca0
* /dev/dvb/adapter0/demux0
* /dev/dvb/adapter0/dvr0
* /dev/dvb/adapter0/frontend0
* /dev/dvb/adapter0/net0

* /dev/dvb/adapter0/osd0
* /dev/dvb/adapter0/video0

The primary device nodes that we are interested in (at this stage) for the Avermedia DVB-T are:

* /dev/dvb/adapter0/dvr0
* /dev/dvb/adapter0/frontend0

The dvr0 device node is used to read the MPEG2 Data Stream and the frontend0 node is used to tune the frontend tuner module.

At this stage, it has not been able to ascertain the functionality of the remaining device nodes in respect of the Avermedia DVBT. However, full functionality in respect of tuning, receiving and supplying the MPEG2 data stream is possible with the currently available versions of the driver. It may be possible that additional functionality is available from the card (i.e. viewing the additional analogue inputs that the card presents), but this has not been tested yet. If I get around to this, I'll update the document with whatever I find.

To power up the card, load the following modules in the following order:

* modprobe bttv (normally loaded automatically)
* modprobe dvb-bt8xx (or place dvb-bt8xx in /etc/modules)

Insertion of these modules into the running kernel will activate the appropriate DVB device nodes. It is then possible to start accessing the card with utilities such as scan, tzap, dvbstream etc.

The frontend module sp887x.o, requires an external firmware. Please use the command "get_dvb_firmware sp887x" to download it. Then copy it to /usr/lib/hotplug/firmware or /lib/firmware/ (depending on configuration of firmware hotplug).

Receiving DVB-T in Australia

I have no experience of DVB-T in other countries other than Australia, so I will attempt to explain how it works here in Melbourne and how this affects the configuration of the DVB-T card.

The Digital Broadcasting Australia website has a Reception locatortool which provides information on transponder channels and frequencies. My local transmitter happens to be Mount Dandenong.

The frequencies broadcast by Mount Dandenong are:

Table 1. Transponder Frequencies Mount Dandenong, Vic, Aus.

Broadcaster	Channel	Frequency
ABC	VHF 12	226.5 MHz

TEN	VHF 11	219.5 MHz
NINE	VHF 8	191.625 MHz
SEVEN	VHF 6	177.5 MHz
SBS	UHF 29	536.5 MHz

The Scan utility has a set of compiled-in defaults for various countries and regions, but if they do not suit, or if you have a pre-compiled scan binary, you can specify a data file on the command line which contains the transponder frequencies. Here is a sample file for the above channel transponders:

```
# Data file for DVB scan program
#
# C Frequency SymbolRate FEC QAM
# S Frequency Polarisation SymbolRate FEC
# T Frequency Bandwidth FEC FEC2 QAM Mode Guard Hier
T 226500000 7MHz 2/3 NONE QAM64 8k 1/8 NONE
T 191625000 7MHz 2/3 NONE QAM64 8k 1/8 NONE
T 219500000 7MHz 2/3 NONE QAM64 8k 1/8 NONE
T 177500000 7MHz 2/3 NONE QAM64 8k 1/8 NONE
T 536500000 7MHz 2/3 NONE QAM64 8k 1/8 NONE
```

The defaults for the transponder frequency and other modulation parameters were obtained from www.dba.org.au.

When Scan runs, it will output channels.conf information for any channel's transponders which the card's frontend can lock onto. (i.e. any whose signal is strong enough at your antenna).

Here's my channels.conf file for anyone who's interested:

```
ABC HDTV:226500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_3_4:FEC_3_4:QAM_64
:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_16:HIERARCHY_NONE:2307:0:560
ABC TV Melbourne:226500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_3_4:FEC_3_
4:QAM_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_16:HIERARCHY_NONE:512:65
0:561
ABC TV 2:226500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_3_4:FEC_3_4:QAM_64
:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_16:HIERARCHY_NONE:512:650:562
ABC TV 3:226500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_3_4:FEC_3_4:QAM_64
:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_16:HIERARCHY_NONE:512:650:563
ABC TV 4:226500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_3_4:FEC_3_4:QAM_64
:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_16:HIERARCHY_NONE:512:650:564
ABC DiG Radio:226500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_3_4:FEC_3_4:Q
AM_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_16:HIERARCHY_NONE:0:2311:56
6
TEN Digital:219500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_3_4:FEC_1_2:QAM
_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_16:HIERARCHY_NONE:512:650:158
5
TEN Digital 1:219500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_3_4:FEC_1_2:Q
AM_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_16:HIERARCHY_NONE:512:650:1
586
TEN Digital 2:219500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_3_4:FEC_1_2:Q
AM_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_16:HIERARCHY_NONE:512:650:1
587
TEN Digital 3:219500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_3_4:FEC_1_2:Q
AM_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_16:HIERARCHY_NONE:512:650:1
588
```

TEN Digital:219500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_3_4:FEC_1_2:QAM
_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_16:HIERARCHY_NONE:512:650:158
9
TEN Digital 4:219500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_3_4:FEC_1_2:Q
AM_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_16:HIERARCHY_NONE:512:650:1
590
TEN Digital:219500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_3_4:FEC_1_2:QAM
_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_16:HIERARCHY_NONE:512:650:159
1
TEN HD:219500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_3_4:FEC_1_2:QAM_64:T
RANSMISSION_MODE_8K:GUARD_INTERVAL_1_16:HIERARCHY_NONE:514:0:1592
TEN Digital:219500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_3_4:FEC_1_2:QAM
_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_16:HIERARCHY_NONE:512:650:159
3
Nine Digital:191625000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_3_4:FEC_1_2:QA
M_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_16:HIERARCHY_NONE:513:660:10
72
Nine Digital HD:191625000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_3_4:FEC_1_2
:QAM_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_16:HIERARCHY_NONE:512:0:1
073
Nine Guide:191625000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_3_4:FEC_1_2:QAM_
64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_16:HIERARCHY_NONE:514:670:1074
7 Digital:177500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_2_3:FEC_2_3:QAM_6
4:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_8:HIERARCHY_NONE:769:770:1328
7 Digital 1:177500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_2_3:FEC_2_3:QAM
_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_8:HIERARCHY_NONE:769:770:1329
7 Digital 2:177500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_2_3:FEC_2_3:QAM
_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_8:HIERARCHY_NONE:769:770:1330
7 Digital 3:177500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_2_3:FEC_2_3:QAM
_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_8:HIERARCHY_NONE:769:770:1331
7 HD Digital:177500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_2_3:FEC_2_3:QA
M_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_8:HIERARCHY_NONE:833:834:133
2
7 Program Guide:177500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_2_3:FEC_2_3
:QAM_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_8:HIERARCHY_NONE:865:866:
1334
SBS HD:536500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_2_3:FEC_2_3:QAM_64:T
RANSMISSION_MODE_8K:GUARD_INTERVAL_1_8:HIERARCHY_NONE:102:103:784
SBS DIGITAL 1:536500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_2_3:FEC_2_3:Q
AM_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_8:HIERARCHY_NONE:161:81:785
SBS DIGITAL 2:536500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_2_3:FEC_2_3:Q
AM_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_8:HIERARCHY_NONE:162:83:786
SBS EPG:536500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_2_3:FEC_2_3:QAM_64:
TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_8:HIERARCHY_NONE:163:85:787
SBS RADIO 1:536500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_2_3:FEC_2_3:QAM
_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_8:HIERARCHY_NONE:0:201:798
SBS RADIO 2:536500000:INVERSION_OFF:BANDWIDTH_7_MHZ:FEC_2_3:FEC_2_3:QAM
_64:TRANSMISSION_MODE_8K:GUARD_INTERVAL_1_8:HIERARCHY_NONE:0:202:799

Known Limitations

At present I can say with confidence that the frontend tunes via /dev/dvb/adapter{x}/frontend0 and supplies an MPEG2 stream via /dev/dvb/adapter{x}/dvr0. I have not tested the functionality of any other part of the card yet. I will do so

over time and update this document.

There are some limitations in the i2c layer due to a returned error message inconsistency. Although this generates errors in dmesg and the system logs, it does not appear to affect the ability of the frontend to function correctly.

Further Update

dvbstream and VideoLAN Client on windows works a treat with DVB, in fact this is currently serving as my main way of viewing DVB-T at the moment. Additionally, VLC is happily decoding HDTV signals, although the PC is dropping the odd frame here and there - I assume due to processing capability - as all the decoding is being done under windows in software.

Many thanks to Nigel Pearson for the updates to this document since the recent revision of the driver.

February 14th 2006