

Copyright 2009, DVBLLogic

For additional information on DVBLink visit our website [www.dvblogic.com](http://www.dvblogic.com)

## Introduction

This document describes what DVBLink is and how it works. It provides top level technical details of DVBLink operations and describes the functionality of DVBLink SDK.

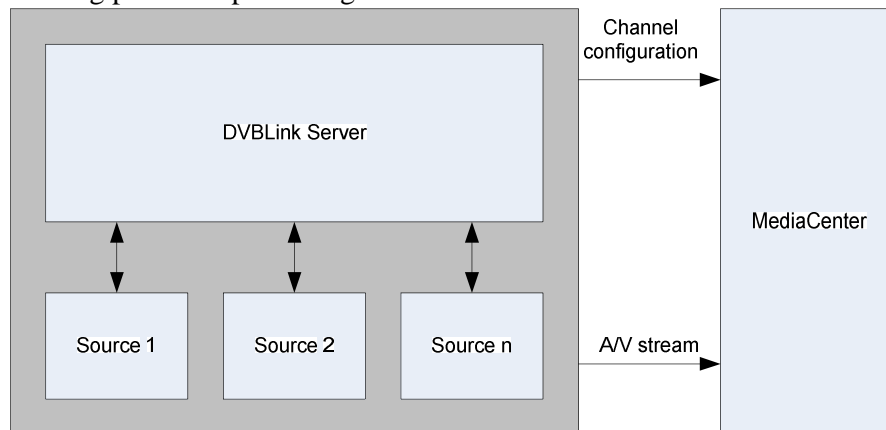
## What is DVBLink?

DVBLink is the software package, which brings streaming A/V content from originally non-compliant sources to Windows Media Center. It does this by emulating DVB-S tuner device. This tuner device is virtual as it does not control any real hardware on itself. Instead it acts as a pass-through between MediaCenter and DVBLink Server. The latter is actually responsible for controlling real signal sources. As MediaCenter natively supports DVB-S tuners user gets full control over incoming A/V content, including live watching, pause live TV, one-touch and timer recordings, EPG listings etc.

## How does DVBLink work?

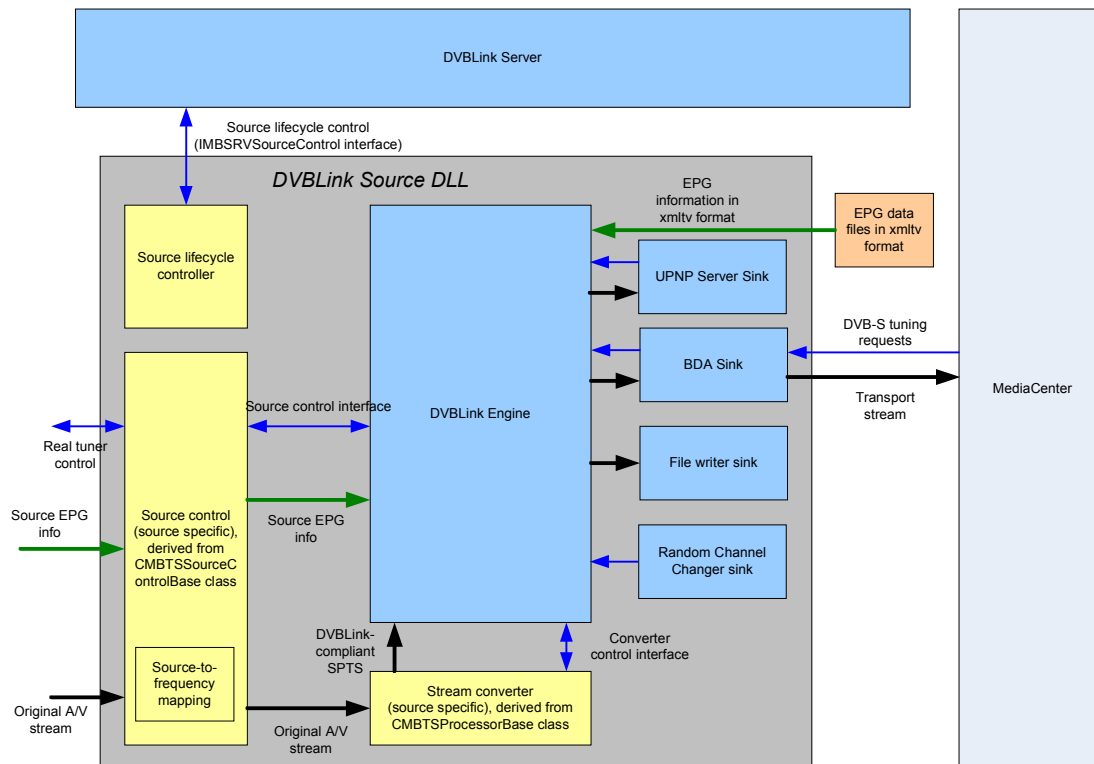
### *Top-level overview*

The following picture depicts a high-level overview of DVBLink:



DVBLink Server is a core part of DVBLink software. This server controls the signal sources and provides MediaCenter with the channel configuration information and with the actual A/V stream. All sources talk to DVBLink Server via predefined interfaces. DVBLink Server simultaneously supports multiple sources of the same or different types.

The following picture provides a zoom into internal organization of DVBLink:



The DVBLink generic framework components are shown in blue color in the picture above and source-specific components - in yellow.

The DVBLink service communicates with DVBLink source via *Source Lifecycle control* interface. This is a very top level interface, providing functions StartSource, ShutdownSource, Standby and Resume. It also provides information to the source on where it is located and where its settings can be found in registry.

The main functionality of DVBLink is provided by *DVBLink engine* library. This library abstracts the signal source from actual details of communicating with MediaCenter. MediaCenter communication is realized by BDA sink component, which sends channel change requests to DVBLink core library in form of DVB-S tuning requests and forwards transport stream to MediaCenter. Next to BDA sink DVBLink core library can handle a number of other possible sinks. File writer sink and Random Channel Changer sink are implemented for debug purposes. DVBLink features also UPNP server sink, which streams the signal to UPNP-enabled clients. DVBLink engine library prioritizes channel change requests from different sinks and broadcasts stream to all of them.

The following chapters elaborate on a number of important aspects of how DVBLink works – streaming mechanisms and tuner control.

### ***DVBLink as UPNP server***

As it was said in the previous chapter DVBLink acts not only as a stream source for MediaCenter, but also as a UPNP server, built on basis of Platinum UPNP SDK. The UPNP-enabled clients can browse the list of channels and play back the live stream as received from DVBLink. The following UPNP clients have been tested and validated with DVBLink UPNP server:

- Nero ShowTime v4
- PowerDVD v7 and v8

- Sony Playstation 3

## Guide a la Carte

DVBLink Engine implements core functionality of Guide a la Carte feature – translation of custom EPG information format into EIT packets. These packets are multiplexed into outgoing transport stream, which is sent to sinks. MediaCenter can extract this information from the stream and present in Guide.

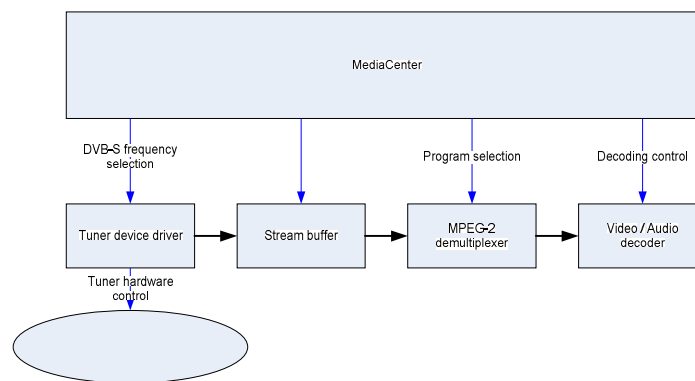
To use Guide a la Carte feature, source should derive a source-specific class from CMBTSEPGSource and create an instance of it on every channel change.

## Streaming mechanisms

### MediaCenter live streaming

The streaming mechanism of DVBLink is designed to provide seamless integration with MediaCenter streaming mechanisms and the way it communicates with tuner device.

The following picture shows how MediaCenter controls the processing pipeline for the signal coming from tuner device (some small details are omitted for better readability):



When recording or live stream playback is requested by user, MediaCenter builds the pipeline, consisting of tuner device filter, stream buffer, MPEG-2 demultiplexer and A/V decoder filter.

Tuner device filter consists internally of capture and tuner filters, which control streaming and tuning respectively. Stream buffer filter provides time-shift behavior.

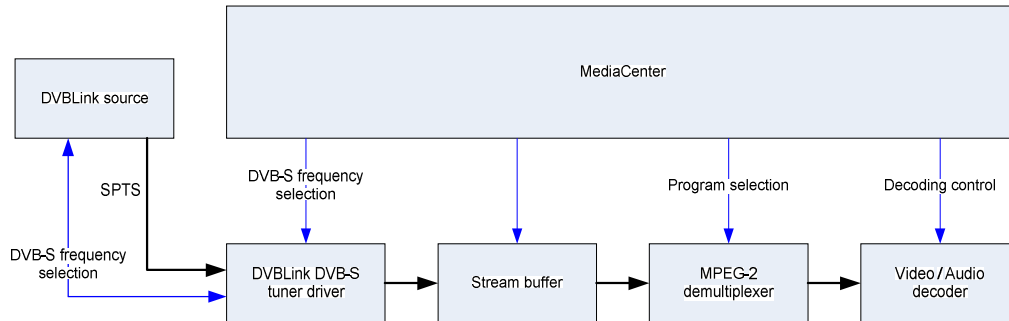
MPEG-2 demux filter selects the audio and video streams, which belong to a particular program, which then are decoded by A/V decoder filter(s).

For different tuner hardware different tuner device drivers are written by their manufacturers. The rest of the pipeline is fixed (with certain tools it is possible to select another A/V decoder).

DVB-S transport stream may carry more than one program (service) inside. Tuner device drivers provide such a stream for each DVB-S modulation frequency. The selection of a particular program from the transport stream is handled by demux. As MediaCenter always uses its own demultiplexer (Microsoft MPEG-2 demultiplexer), the tuning requests for different programs inside the same transport stream are not visible for the 3<sup>rd</sup> party software (including tuner device driver).

### DVBLink-MediaCenter streaming and tuner control

The following picture shows DVBLink MediaCenter streaming:



DVBLink is visible to MediaCenter as a DVB-S tuner. However this tuner is virtual as it does not control any hardware directly. Instead it forwards tuning requests to DVBLink source. It accepts the stream from this DVBLink source and sends it further down the graph for processing. It should be noted that DVBLink source runs in a user mode, while DVBLink tuner is the kernel mode component. DVBLink implements a special communication mechanism between the two with zero data copying for better performance.

DVBLink source must provide *SPTS* (single program transport stream) to DVBLink tuner driver. This is because (as it was said in the previous chapter) program selection inside the transport stream is done by demux and is not visible to DVBLink. To be able to select a correct channel at the real signal source DVBLink has to maintain one-to-one relationship between real source program and DVB-S frequency. Another important consequence of such tuner control mechanism is that DVBLink source has to map DVB-S frequencies on channels of the real A/V source. This mapping is source-specific and may or may not require interaction with user to do it.

## DVBLink internal streaming organization

Source-specific Source Control is derived from CMBTSSourceControlBase class. It accepts A/V stream in its original format. Source Control runs in user mode. Source Control outputs the stream to Stream Converter in one of the predefined formats.

These formats are:

- Transport stream
- PES (transport stream packets with the stripped transport stream packet header and adaptation field stripped off)
- Program stream
- Section stream – PSI sections already reconstructed from the original transport stream

The rest of the DVBLink components, including Stream Converter can subscribe to one of the streams above. Source Control does not have to implement all of the stream types above. It should only implement the stream type, which is the best for Stream Converter to work with.

The Stream Converter is derived from CMBTSProcessorBase class. It implements conversion of the original stream to DVBLink-compliant SPTS, restoring as much as possible of the original stream information. DVBLink compliancy means that

- Output stream has to be DVB-S compliant MPEG-2 transport stream
- Output stream has to be SPTS
- Output TS has to have unique Network ID, transport stream ID and Service ID for each DVB-S frequency according to the following formula:
  - $NID=ONID=TID=SID=DVB-S \text{ freq} / 10000$
- Network Provider name is set to “DVBLink”

The resulting stream is sent to CMBTSReadyCallback class via its OnStreamReady() function. DVBLink engine broadcasts it then to all connected sinks.

### ***DVBLink tuner control mechanisms***

The whole DVBLINK source works internally with DVB-S frequencies. It means that DVBLINK engine and Source Control always receive and handle DVBS tuning requests. When a new tuning request comes from one of the sinks, DVBLINK core library prioritizes them.

If tuning request is taken for handling, DVBLINK engine first stops the streaming (StopStreaming() call on Stream Converter interface). Then it sends tuning request to Source Control and starts timeout timer. Source control is responsible for mapping of the DVB-S frequency to a particular channel of the real source and changing of the channel itself. When channel is changed, Source Control broadcasts this event to all subscribers through CMDfilter mechanism.

When DVBLINK engine receives this event from stream control it returns the call to a sink, which has requested the channel change. However the streaming is not started yet. DVBLINK engine starts polling the Source Control every 200 ms, waiting for actual streaming to start (calling GetStreamInfo() function of the Source Control). As soon as this function returns 1, DVBLINK engine assumes that actual streaming is started and starts the Stream Converter by calling its StartStreaming() function (information about new stream is passed as a parameter of this function - STSStreamInfo structure).

### ***DVBLink transport and program stream processing helpers***

DVBLink engine provides a number of basic routines, which assist in processing of transport or program streams. These routines are described in mb\_tsinfo.h file and expose functions to

- Parse out the content of TS packets
- Parse out the content of EIT packets
- Parse the payload of SDT, PAT, PMT and SDT sections from the transport stream packets
- Generate transport stream packets for PAT, PMT, SDT and NIT sections and PES buffers
- Etc.

## **DVBLink SDK**

### ***General***

DVBLink SDK allows 3<sup>rd</sup> party companies and developers to create their own DVBLINK compatible signal sources. In principle, everything that produces digital streaming signal, which is reasonably close to transport stream, and can switch channels can be made available in Windows MediaCenter – cable boxes, satellite settopboxes, Dreambox, IPTV streams, UPNP streams, custom signal sources etc. DVBLINK SDK abstracts developer from actual details of MediaCenter communication and provides comprehensive set of tools (header files, library, documentation and sample code) to write custom DVBLINK compatible signal source with minimum time and effort.

## **Installation**

To install DVBLink SDK unzip DVBLinkSDK.zip file into one of directories on your hard disk. When this operation completes the following directories will be created in your destination folder:

- DVBLinkSDK – SDK root folder
  - Docs – folder with documentation
    - html – folder with documentation of the code in HTML format
  - DVBLinkService – folder with installation of DVBLink Service
  - Inc – SDK header files
  - Lib – DVBLink Engine library
  - Sample – root folder for DVBLink sample code
    - DVBLinkSDKSample – DVBLink SDK sample
    - Resources – files, needed to run the sample
    - StartTest – debug helper project

## **Build environment**

DVBLink SDK was created with Microsoft Visual C++ Express 2005. It will also work with Microsoft Visual Studio 2005 or newer.

The code does not contain any Microsoft Visual Studio-specific code and should be usable by any C++ compiler / linker.

## **DVBLink Service**

Each DVBLink signal source plug-in is controlled by DVBLink Server. This service communicates with plug-in via IMBSRVSourceControl interface, which must be exposed by each plug-in. The following list describes requirements, which DVBLink compatible source must implement to be properly integrated with DVBLink Server:

- Plug-in must be a DLL, which exports the following functions:
  - extern "C" IMBSRVSourceControl\* \_\_stdcall  
MBSRV\_DLLGetSourceControl();
  - extern "C" void \_\_stdcall  
MBSRV\_DLLReleaseSourceControl(IMBSRVSourceControl\* src\_ctrl);
- Plug-in must reside in a dedicated folder under <<DVBLink Service installation path>>\Templates\Sources
- Plug-in must have a dedicated registry key under HKEY\_LOCAL\_MACHINE\SOFTWARE\DVBLink\Templates\Sources. The name of this registry key must be the same as the name of the folder where plug-in resides. The registry key should have at least two values set:
  - *SourceControl* string value, holding the name of the plug-in DLL itself
  - *StartAsService* DWORD value, set to 1

During plug-in initialization DVBLink Server provides it with the pointers to its physical location and its location in registry so that plug-in may retrieve its specific settings.

## **How to create custom DVBLink compatible source**

This chapter gives a short step-wise overview of how custom DVBLink compatible source should be created (Microsoft Visual Studio build environment is taken as example):

1. Create new dynamic link library (DLL) project in Microsoft Visual C++ Express.

2. Create def file and fill it in with the MBSRV\_DLLGetSourceControl and MBSRV\_DLLReleaseSourceControl the function. Also implement this functions in the plug-in's code.
3. Implement plug-in's IMBSRVSourceControl interface
4. Inherit custom signal source control class from CMBTSSourceControlBase class and implement it
5. Inherit signal converter class from CMBTSProcessorBase and implement it
6. Inherit plug-in settings class from CMBGELibSettings and implement it
7. Implement calls to MCEBridgeInitSignalSource and MCEBridgeTerminateSignalSource functions inside your IMBSRVSourceControl handler.
8. Test new source behavior, using DVBLink debugging tools
9. Test new source behavior under real DVBLink Server
10. Deploy new DVBLink compatible source

### ***DVBLink debugging tools***

DVBLink SDK provides the following debugging tools to assist in testing new DVBLink compatible signal source:

- StartTest program. This program is available as a source code under <<DVBLinkSDK installation dir>>\samples\StartTest. It allows testing a DVBLink source plug-in without DVBLink Service itself – from the command line. This program loads the plug-in's control DLL and calls its control interface functions in the same way as DVBLink Service does. *To use the StartTest program with your source DLL do not forget to change the DLL name and Registry key name for your particular source!*
- File writer sink. This DVBLink sink writes the transport stream, which is sent to BDA drivers, to a file. This file can be used for visual inspection of the stream. The output transport stream file is called dvblink\_out.ts and is located in the sink's directory - <<DVBLink Server installation path>>\Sinks\File. Please note that output transport stream file is re-created each time the DVBLink source plug-in is initialized.
- Random Channel Changer. This DVBLink sink randomly goes through all mapped DVB-S frequencies and requests channel changes for all of them. This sink is not part of standard DVBLink Server installation and should be installed from <<DVBLink SDK installation path>>\Tools. To install the random channel changer sink, copy *rndchchanger* folder and its contents to <<DVBLink Service installation path>>\Sinks and execute *rndchchanger.reg* file to enter registry information for this sink. The channel change period (in seconds) can be adjusted via *SimulChChangePeriod* DWORD value in the sink's registry. *Do not forget to disable this sink before using DVBLink source plug-in with MediaCenter.*

It is also suggested to use TS Reader and VLC Player tools for visual inspection of incoming and produced stream.

### ***Sample***

DVBLink SDK contains a sample code for DVBLink compatible source – DVBLinkSDKSample. This sample can be found in <<DVBLink SDK installation folder>>\Sample.

DVBLink SDKSample shows all important aspects of DVBLink compatible source functionality – life cycle control, channel change control, stream conversion and

integration with DVBLink Engine. For its functionality DVBLink sample maps two transport stream files to two DVB-S frequencies and starts streaming of these files on channel change requests from MediaCenter.

To build the sample you need Microsoft Visual C++ Express 2005 or Microsoft Visual Studio 2005 or newer.

To run the sample with StartTest program you need to copy 1stmusic.ts and startv.ts files to reside next to DVBLinkSDKSample.dll file.

To run the sample under DVBLink Service, you need to

- Execute DVBLinkSDKSample.reg file from <<DVBLink SDK installation folder>>\Sample\Resources
- Create DVBLinkSDKSample directory under <<DVBLink Service installation path>>\Templates\Sources
- Copy 1stmusic.ts and startv.ts files to this folder
- Copy DVBLinkSDKSample.dll to this folder
- Initiate DVBLinkSDKSample source using DVBLink Source Manager
- Restart DVBLink Service

### ***Channels synchronization in MediaCenter***

To synchronize DVBLink channels with MediaCenter channel database use MediaCenter DVBLink addin.

### ***Support***

For all questions, bug reports, remarks etc. please visit our forums at [www.dvblogic.com](http://www.dvblogic.com).

We can also be reached by e-mail [info@dvblogic.com](mailto:info@dvblogic.com)