

DAB+ Radcap



Manual

Introduction

The DAB+ Radcap receives and decodes the entire contents of a DAB+/DAB ensemble, rendering each audio service as a virtual Windows audio capture device for use with multi-channel recording or monitoring software. Broadcast data services, including DLS text and MOT slideshows, are also available through a simple application programming interface.

The card supports both legacy DAB MP2 audio coding as well as the new HE-AAC v2 encoding used with DAB+ broadcasts.

Multiple cards can be installed, allowing simultaneous monitoring or recording of several ensembles.

A sample application with source code is provided, allowing monitoring of DAB+/DAB audio and programme-associated data.

System Requirements

Platform:	Windows XP SP2 or later (32-bit and 64-bit versions)
Processor:	1GHz Pentium 4 or better
Memory:	256MB minimum
Other:	Sound card or motherboard sound port for monitoring

Specifications

Tuning range:	Band III (174-240MHz)
DAB format:	Mode 1
RF input:	BNC connector
Number of services:	Unlimited (audio input device limit of 32 on XP and Server 2003)
Audio decoding:	MP2, HE-AAC v2
Audio sampling rate:	48kHz 16-bit stereo (other rates supported automatically through the Windows sampling rate converter)

Installation

The DAB+ Radcap card uses static-sensitive components. Observe the usual precautions against static electricity when handling the card and do not touch the PCI edge connector contacts.

Ensure that Windows XP with at least Service Pack 2 or a later operating system is installed on the PC. The DAB+ Radcap cannot be used on any earlier versions of Windows as they don't support dynamic audio sub-devices. It is recommended that the latest Service Pack and security updates be installed.

Switch off the PC and unplug the power lead before inserting the card into any vacant PCI slot. The card can be used in either 5V or 3.3V standard 32-bit slots or PCI-X 64-bit slots.

Restart the PC and allow Windows to boot up.

Windows XP, Server 2003, Vista, Server 2008 - Windows will report that new hardware has been found and the New Hardware wizard will start. Insert the driver CD supplied with the card and proceed through the wizard. Allow Windows to search for the driver – do NOT specify a driver location or file name.

Windows 7, Server 2008 R2 – Windows no longer searches removable media for drivers. Open Device Manager, where the Radcap will be listed under Other Devices as a Multimedia Audio Controller. Right-click on it, select *Update Driver Software*, then click on *Browse my computer for driver software* and click on the Browse button to navigate to the driver's location. Click on *Next* to install the driver.

Windows may warn that the driver being installed has not been certified by Microsoft, which is true. Click on **Continue** to complete the installation. The drivers are digitally signed by Innes Corporation as required by 64-bit Windows Vista and later systems.

Some modern processors offer **hyperthreading**, which is a limited form of multiprocessing. However, some elements of the processor, such as the floating point unit and memory cache, are shared between the executing threads, and this can cause a high priority thread, such as an audio processing thread, to be blocked by a lower priority thread that's using the shared resource. This can result in skipping during audio recording and gaps during playback. If this problem occurs, we recommend that hyperthreading be disabled in the motherboard BIOS. Note that this problem doesn't occur with true multi-core processors.

Antenna

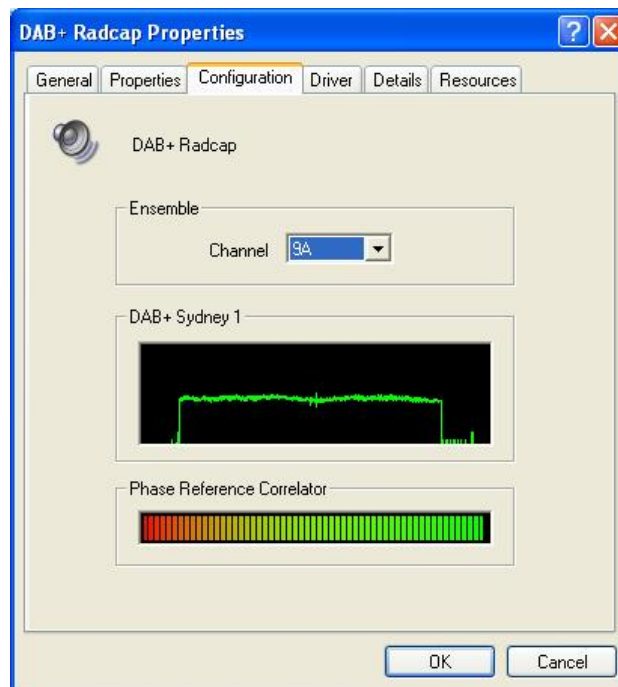
The DAB+ Radcap requires an external vertically polarised antenna to receive the stations. The type of antenna needed depends on the signal levels in the area in which it is being used. A splitter may be used to feed a single antenna into multiple cards, but a masthead amplifier may then be needed to compensate for the splitter losses. If using an amplifier, take care not to overdrive the cards as performance will be severely degraded. The minimum gain needed to provide good reception on the weakest ensemble being monitored should be used, and in some applications one or more directional antennas may be required.

We strongly recommend fitting an external lightning suppressor to minimise the risk of damage to the card.

Do not use a horizontally polarised TV antenna with the DAB+ Radcap, as the DAB signals are likely to be overloaded by adjacent television services.

Configuration

To set the card's ensemble channel, open **Control Panel** and double-click on **Sounds and Multimedia**, click on the **Hardware** tab, select **DAB+ Radcap** and click on **Properties**. In the Properties window click on the **Configuration** tab and select the required channel from the drop-down list.



If a signal is present, the ensemble RF spectrum and phase reference correlator level are shown, with the broadcast ensemble name appearing on the spectrum box.

If multiple cards are installed in the PC, each must be set to a different ensemble channel.

Once the ensemble is set and the card is receiving a satisfactory signal, the audio services on the ensemble will appear as Windows audio input devices, using the names broadcast for the services.

Windows XP and Server 2003 limit the total number of audio input devices to 32. Please be aware of this limit, particularly if there are other audio input devices present in the PC. This limit has been removed in Windows Vista and later operating systems.

Recording

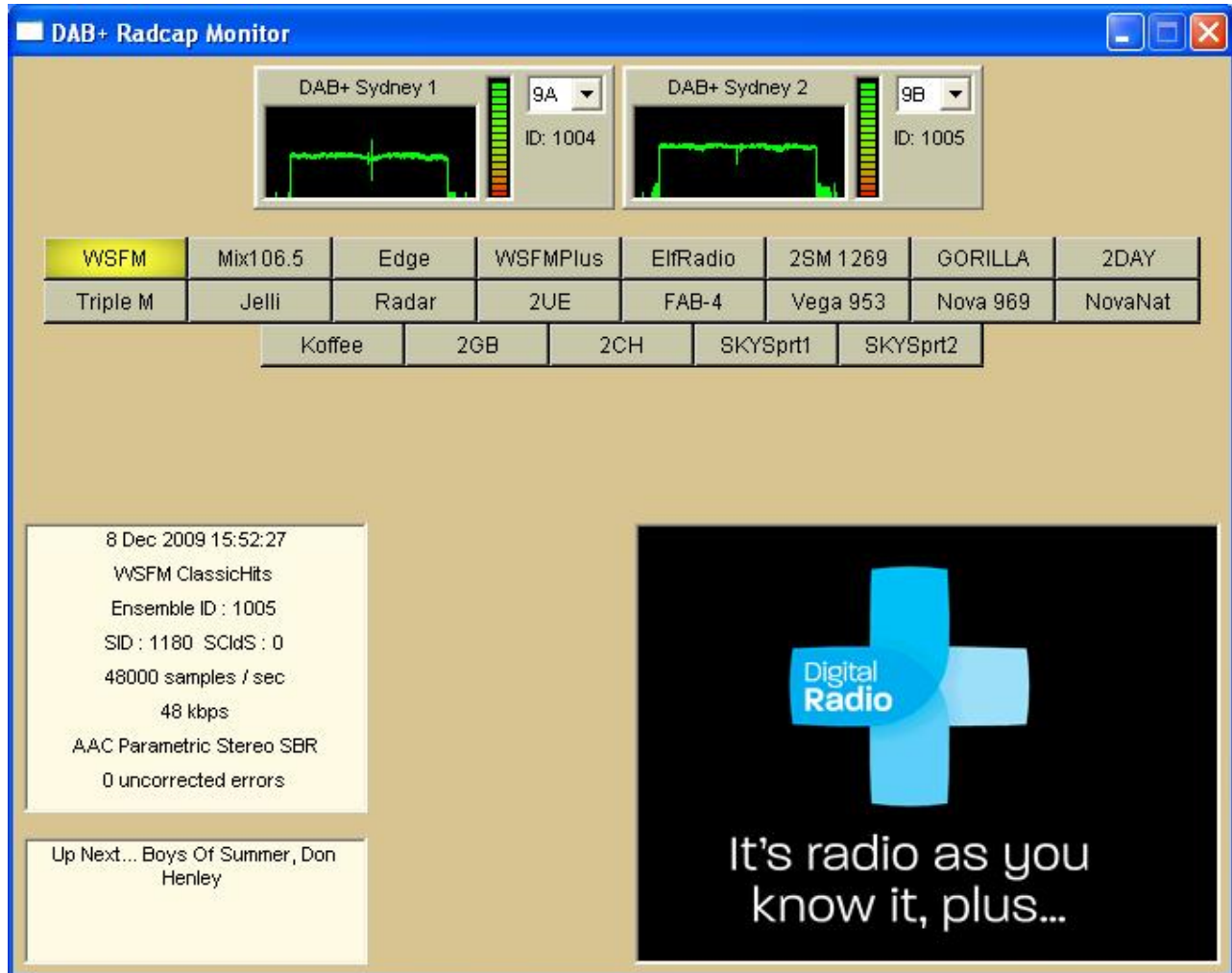
Any application that records from standard wave input devices can be used to record the audio streams from the DAB+ Radcap.

Make sure the recording software you are using allows you to select the audio input devices.

A recording level and mute control are provided for each service through the devices' mixer ports.

If recording a large number of channels using any form of audio compression, be sure to allow sufficient processing power to both capture the audio and compress it. The Performance Monitor in Windows' Task Manager provides a useful guide to CPU loading. Regrettably at this time the Windows audio subsystem does not support recording directly in the DAB+ / DAB native AAC or MP2 compressed formats.

Monitor Application



A simple monitor application is included in the distribution package, along with its C++ source code utilising the DAB+ Radcap programming API documented in Appendix A.

The application displays a control panel for each card and creates buttons for each audio service. When a button is clicked, it plays the audio through the default output device while displaying information obtained from the service and any DLS text and MOT images being broadcast.

Each card panel displays the ensemble name and identifier, along with the phase reference correlator level and signal spectrum. A drop-down list allows the ensemble channel to be set.

The buttons corresponding to each audio service are sorted by service ID number (SId). The application periodically scans the list of wave capture devices, adding, removing or renaming buttons as required.

Support

For all support matters go to www.innescorp.com.au and click on Support – Help Desk. Software and driver updates may also be made available from time to time and these will be placed on the DAB+ Radcap page of the product catalogue.

Appendix A – Programming the DAB+ Radcap

On the driver CD under **\\DAB Radcap\\API** you will find DABRadcap.dll, DABRadcap.exp, DABRadcap.lib and DABRadcap.h, which can be used with application programs to obtain information about each service while reading the transmitted FPAD and XPAD information. The functions exported by the DLL are grouped into card-centric and service-centric operations.

The card-centric functions require a card identifier number in the range 0 to CardCount-1, and support operations related to the hardware, including setting and retrieving the tuner channel, monitoring the phase reference symbol correlator and signal spectrum, and retrieving the tuned ensemble identifier and name.

The service-centric functions are accessed using a handle returned by DabRadcapOpen (). This function in turn takes a Windows wave capture device ID number. Be aware that wave ID numbers are dynamic, and can change as devices are added or removed, or if the user changes the default capture device. The handle returned by DabRadcapOpen () will remain associated with a given service regardless of any changes to the wave ID numbers, however once the handle is closed, it must not be assumed that calling this function again with the same wave ID will access the same service as previously obtained.

For applications using DirectSoundCapture, the system property DSPROPERTY_DIRECTSOUNDDEVICE_DESCRIPTION returns the wave input ID number for a given DirectSoundCapture GUID. Refer to System Property Sets in the DirectSound documentation for details. Be aware that, like wave input ID numbers, DirectSoundCapture GUIDs may change as devices are added or removed and should not be assumed to be static within a session.

Overlapped I/O should be used with the DabRadcapGetPAD () function, as this does not return data immediately but waits until the next block of data is received. Multiple buffers should be used to avoid loss of data. Refer to the PadThread () function in the sample monitor application to see how this is implemented.

Card-centric Functions

UINT __stdcall DabRadcapGetNumberOfCards ();

Return value

If the function succeeds, the return value is the number of DAB+ Radcap cards installed in the PC. If the function fails, or if no cards are installed, the return value is 0. To get extended error information, call **GetLastError**.

Comments

Subsequent calls to card-centric functions should pass a card number in the range zero to one less than the return value of this function.

UINT __stdcall DabRadcapGetCardChannel (
 UINT CardNum);

Parameters

CardNum

Card identifier in the range zero to one less than the number of DAB+ Radcap cards.

Return value

The return value is the channel to which the card is tuned. The channel numbers used in this function correspond to European DAB channel designators and nominal centre frequencies according to the table in Appendix B.

BOOL __stdcall DabRadcapSetCardChannel (
 UINT CardNum,
 UINT Channel);

Parameters

CardNum

Card identifier in the range zero to one less than the number of DAB+ Radcap cards.

Channel

Channel number (refer to Appendix B).

Return value

If the function succeeds the return value is TRUE.

If the function fails, the return value is FALSE. To get extended error information, call **GetLastError**.

BOOL `__stdcall DabRadcapGetCardEnsembleInformation` (
 UINT CardNum,
 EnsembleIdentification *pInfo);

Parameters

CardNum

Card identifier in the range zero to one less than the number of DAB+ Radcap cards.

pInfo

Pointer to an EnsembleIdentification structure which receives the requested information.

Return value

If the function succeeds the return value is TRUE.

If the function fails, the return value is FALSE. To get extended error information, call **GetLastError**.

Comments

This function returns the Ensemble ID, label and label mask that is being broadcast. If this information is unavailable, the Ensemble ID is set to zero.

UINT `__stdcall DabRadcapGetCardPhaseReferenceCorrelator` (
 UINT CardNum);

Parameters

CardNum

Card identifier in the range zero to one less than the number of DAB+ Radcap cards.

Return value

The return value is the phase reference symbol correlator level, in the range 0 to 100 where 100 corresponds to perfect correlation.

```
BOOL __stdcall DabRadcapGetCardSpectrum (  
    UINT CardNum,  
    float *SpectrumArray,  
    SIZE_T SpectrumArraySize);
```

Parameters

CardNum

Card identifier in the range zero to one less than the number of DAB+ Radcap cards.

SpectrumArray

Pointer to an array of 2048 floats into which the spectrum data is written.

SpectrumArraySize

The size in bytes of the array to which SpectrumArray points.

Return value

If the function succeeds the return value is TRUE.

If the function fails, the return value is FALSE. To get extended error information, call **GetLastError**.

Comments

The values written into SpectrumArray represent decibels in the range 0 to 60. The first element in the array represents the lowest frequency in the tuner passband. The frequency spacing is 1kHz, thus the returned data spans a frequency range of 2048kHz.

Service-centric Functions

HANDLE __stdcall DabRadcapOpen (
 UINT waveInID);

Parameters

waveInID
 waveIn device identifier.

Return value

If the function succeeds, the return value is a handle for use with other DabRadcap API calls. If the function fails, the return value is INVALID_HANDLE_VALUE. To get extended error information, call **GetLastError**.

Comments

Use the **CloseHandle** function to close an object handle that DabRadcapOpen returns.

BOOL __stdcall DabRadcapGetEnsembleInformation (
 HANDLE hDev,
 EnsembleIdentification *pInfo);

Parameters

hDev
 Handle returned by **DabRadcapOpen**.
pInfo
 Pointer to an EnsembleIdentification structure which receives the requested information.

Return value

If the function succeeds the return value is TRUE.

If the function fails, the return value is FALSE. To get extended error information, call **GetLastError**.

Comments

This function returns the Ensemble ID, label and label mask that is being broadcast. If this information is unavailable, the Ensemble ID is set to zero.

```
BOOL __stdcall DabRadcapGetServiceInformation (  
    HANDLE hDev,  
    ServiceInfo *pInfo);
```

Parameters

hDev
Handle returned by **DabRadcapOpen**.

pInfo
Pointer to a ServiceInfo structure which receives the requested information.

Return values

If the function succeeds the return value is TRUE.

If the function fails, the return value is FALSE. To get extended error information, call **GetLastError**.

```
BOOL __stdcall DabRadcapGetPAD (  
    HANDLE hDev,  
    PAD *pPad,  
    OVERLAPPED *pOverlap);
```

Parameters

hDev
Handle returned by **DabRadcapOpen**.

pPAD
Pointer to a PAD structure which receives the next broadcast FPAD and XPAD contents.

pOverlap
Pointer to an OVERLAPPED structure. For overlapped operations, **DabRadcapGetPAD** returns immediately, and the event object is signalled when the operation has been completed. Otherwise, the function does not return until the operation has been completed or an error occurs.

Return values

If the function succeeds the return value is TRUE.

If the operation fails or is pending, the return value is FALSE. To get extended error information, call **GetLastError**, which returns ERROR_IO_PENDING if an overlapped operation is pending.

Comments

Refer to *Radio Broadcasting Systems; Digital Audio Broadcasting (DAB) to mobile, portable and fixed receivers* (ETSI EN 300 401) for details on how to interpret FPAD and XPAD information. Refer to *Digital Radio Broadcasting; Multimedia Object Transfer (MOT) protocol* (ETSI EN 301 324) for details on decoding MOT information.

The audio stream must be open and running in order to receive PAD data, otherwise this call will fail and **GetLastError** will return ERROR_NOT_READY.

Because PAD information is transmitted frequently (every 24ms for DAB or 120ms for DAB+), the use of multiple buffers and overlapped I/O is strongly recommended to avoid loss of data.

```
BOOL __stdcall DabRadcapGetServiceStatus (  
    HANDLE hDev,  
    ServiceStatus *pStatus );
```

Parameters

hDev

Handle returned by **DabRadcapOpen**.

pStatus

Pointer to a *ServiceStatus* variable into which the result is to be written.

Return values

If the function succeeds the return value is TRUE.

If the operation fails the return value is FALSE. To get extended error information, call **GetLastError** ().

Comments

If this function returns FALSE or if the status is *ServiceStatusLost*, the application should immediately cease any recording operations, close any open waveIn handle on this device and call **CloseHandle** to close hDev.

A status value of *ServiceStatusAbsent* typically indicates a momentary loss of signal.

A status value of *ServiceStatusPresent* indicates that the signal is being correctly received.

```
UINT __stdcall DabRadcapGetUncorrectedErrorCount (  
    HANDLE hDev );
```

Parameters

hDev

Handle returned by **DabRadcapOpen**.

Return value

The function returns the number of uncorrected errors since the stream was opened.

```
BOOL __stdcall DabRadcapGetDateTime (  
    HANDLE hDev,  
    DABDateTimeInfo *pDateTime );
```

Parameters

hDev

Handle returned by **DabRadcapOpen**.

pDateTime

Pointer to a DABDateTimeInfo structure into which the current UTC and local time is to be written.

Return values

If the function succeeds the return value is TRUE.

If the operation fails the return value is FALSE. To get extended error information, call GetLastError ().

```
BOOL __stdcall DabRadcapGetFormat (  
    HANDLE hDev,  
    DABFormat *pFormat );
```

Parameters

hDev

Handle returned by **DabRadcapOpen**.

pFormat

Pointer to a DABFormat structure into which the current audio format parameters are written.

Return values

If the function succeeds the return value is TRUE.

If the operation fails the return value is FALSE. To get extended error information, call GetLastError ().

Comments

The audio stream must be open and running in order to receive DABFormat data, otherwise this call will fail and **GetLastError** will return ERROR_NOT_READY.

Appendix B – DAB Channel Numbers

Channel Number	Channel Designator	Centre Frequency (MHz)
0	5A	174.928
1	5B	176.640
2	5C	178.352
3	5D	180.064
4	6A	181.936
5	6B	183.648
6	6C	185.360
7	6D	187.072
8	7A	188.928
9	7B	190.640
10	7C	192.352
11	7D	194.064
12	8A	195.936
13	8B	197.648
14	8C	199.360
15	8D	201.072
16	9A	202.928
17	9B	204.640
18	9C	206.352
19	9D	208.064
20	10A	209.936
21	10N	210.096
22	10B	211.648
23	10C	213.360
24	10D	215.072
25	11A	216.928
26	11N	217.088
27	11B	218.640
28	11C	220.352
29	11D	222.064
30	12A	223.936
31	12N	224.096
32	12B	225.648
33	12C	227.360
34	12D	229.072
35	13A	230.784
36	13B	232.496
37	13C	234.208
38	13D	235.776
39	13E	237.488
40	13F	239.200

Appendix C – Copyright Notices

The DAB+ Radcap driver, API library, sample programs and PCB artwork are Copyright © 2009 Innes Corporation Pty Ltd. Some elements of the driver software are attributable to other individuals or companies as follows:

MP2 Audio Decoder

kjmp2 MP2 Decoder Copyright © 2006 Martin J. Fiedler, used with permission.

Reed-Solomon Error Correction

Reed-Solomon decoder Copyright © 2004 Phil Karn, KA9Q, used under the terms of the GNU LESSER GENERAL PUBLIC LICENSE Version 2.1, February 1999. The library source code and licence text is included in the DAB+ Radcap distribution package.

aacPlus-v2 Audio Decoder

FAAD2 aacPlus-v2 Decoder Copyright © 2003-2005 M. Bakker, Nero AG used under commercial licence.