ASIO-README.txt

This document contains information to help you compile PortAudio with

ASIO support. If you find any omissions or errors in this document

please notify us on the PortAudio mailing list.

NOTE: The Macintosh sections of this document are provided for historical

reference. They refer to pre-OS X Macintosh. PortAudio no longer

supports pre-OS X Macintosh. Steinberg does not support ASIO on Mac OS X.

Building PortAudio with ASIO support

------------------------------------

To build PortAudio with ASIO support you need to compile and link with

pa\_asio.c, and files from the ASIO SDK (see below), along with the common

PortAudio files from src/common/ and platform specific files from

src/os/win/ (for Win32).

If you are compiling with a non-Microsoft compiler on Windows, also

compile and link with iasiothiscallresolver.cpp (see below for

an explanation).

For some platforms (MingW, Cygwin/MingW), you may simply

be able to type:

./configure --with-host\_os=mingw --with-winapi=asio [--with-asiodir=/usr/local/asiosdk2]

make

and life will be good. Make sure you update the above with the correct local

path to the ASIO SDK.

For Microsoft Visual C++ there is an build tutorial here:

http://www.portaudio.com/trac/wiki/TutorialDir/Compile/WindowsASIOMSVC

Obtaining the ASIO SDK

----------------------

In order to build PortAudio with ASIO support, you need to download

the ASIO SDK (version 2.0 or later) from Steinberg. Steinberg makes the ASIO

SDK available to anyone free of charge, however they do not permit its

source code to be distributed.

NOTE: In some cases the ASIO SDK may require patching, see below

for further details.

http://www.steinberg.net/en/company/developer.html

If the above link is broken search Google for:

"download steinberg ASIO SDK"

Building the ASIO SDK on Windows

--------------------------------

To build the ASIO SDK on Windows you need to compile and link with the

following files from the ASIO SDK:

asio\_sdk\common\asio.cpp

asio\_sdk\host\asiodrivers.cpp

asio\_sdk\host\pc\asiolist.cpp

You may also need to adjust your include paths to support inclusion of

header files from the above directories.

The ASIO SDK depends on the following COM API functions:

CoInitialize, CoUninitialize, CoCreateInstance, CLSIDFromString

For compilation with MinGW you will need to link with -lole32, for

Borland compilers link with Import32.lib.

Non-Microsoft (MSVC) Compilers on Windows including Borland and GCC

-------------------------------------------------------------------

Steinberg did not specify a calling convention in the IASIO interface

definition. This causes the Microsoft compiler to use the proprietary

thiscall convention which is not compatible with other compilers, such

as compilers from Borland (BCC and C++Builder) and GNU (gcc).

Steinberg's ASIO SDK will compile but crash on initialization if

compiled with a non-Microsoft compiler on Windows.

PortAudio solves this problem using the iasiothiscallresolver library

which is included in the distribution. When building ASIO support for

non-Microsoft compilers, be sure to compile and link with

iasiothiscallresolver.cpp. Note that iasiothiscallresolver includes

conditional directives which cause it to have no effect if it is

compiled with a Microsoft compiler, or on the Macintosh.

If you use configure and make (see above), this should be handled

automatically for you.

For further information about the IASIO thiscall problem see this page:

http://www.rossbencina.com/code/iasio-thiscall-resolver

Building the ASIO SDK on (Pre-OS X) Macintosh

---------------------------------------------

To build the ASIO SDK on Macintosh you need to compile and link with the

following files from the ASIO SDK:

host/asiodrivers.cpp

host/mac/asioshlib.cpp

host/mac/codefragements.cpp

You may also need to adjust your include paths to support inclusion of

header files from the above directories.

(Pre-OS X) Macintosh ASIO SDK Bug Patch

---------------------------------------

There is a bug in the ASIO SDK that causes the Macintosh version to

often fail during initialization. Below is a patch that you can apply.

In codefragments.cpp replace getFrontProcessDirectory function with

the following one (GetFrontProcess replaced by GetCurrentProcess).

bool CodeFragments::getFrontProcessDirectory(void \*specs)

{

FSSpec \*fss = (FSSpec \*)specs;

ProcessInfoRec pif;

ProcessSerialNumber psn;

memset(&psn,0,(long)sizeof(ProcessSerialNumber));

// if(GetFrontProcess(&psn) == noErr) // wrong !!!

if(GetCurrentProcess(&psn) == noErr) // correct !!!

{

pif.processName = 0;

pif.processAppSpec = fss;

pif.processInfoLength = sizeof(ProcessInfoRec);

if(GetProcessInformation(&psn, &pif) == noErr)

return true;

}

return false;

}

###