Subject: Sound device access system based on PortAudio
Posted by dolik.rce on Thu, 04 Nov 2010 20:54:16 GMT
View Forum Message <> Reply to Message

Hi everyone!

Some time ago, I set myself a goal to bring support for sound into U++. Today, I would like to present you first preview of my work in this direction. I chose PortAudio library, which supports great selection of supported platforms and backends (see table below) and the MIT license should be compatible with BSD. This library provides only access to sound devices. to get something really usable, it will have to be extended with some other code that would handle reading, writing and processing of sound files in various formats.

The attached zip archive contains three packages. The biggest one, plugin/portaudio contains sources of Portaudio + few files that take care about proper integration into the u++ build system. Next package, called simply Sound provides U++ wrapper classes that encapsulate the C code from the library. And finally the last package, SoundExample is a simple testing app that gives you some info about available sound APIs, allows to run simple sine wave generator on various sound devices with various options and also contains a simple recording test.

How to compile this: plugin/portaudio can be configured to use various backend via flags (ALSA, JACK, OSS, ASIHPI, DSOUND, WMME, WASAPI, WDMKS). If you don't supply any flags, default one apply (OSS+ALSA on linux, WMME+WASAPI on win32). You can specify multiple flags at once. I recomend using the dot syntax, e.g. "GUI .ALSA .OSS" in order to prevent unnecessary recompilation of everything. For some of the backends to compile/work it might be necessary to have available some libraries or other dependencies (also listed in the table below).

Usual disclaimer applies: This is alpha code, might not be stable and/or might not work as expected. The api in Sound is definitely not stable yet, it will change soon and often. There is no documentation for now, but I promise to write it as soon as the API stabilizes.

Here is a list of supported backends and some info. If you feel like testing on some platforms I didn't have access to (and there is a majority of those), feel free to do so and let me know so I can update the table.

API	Platform	State	Tested	Requirements
jack	unix	compiles	not tested	libjack
oss	unix	works	Arch Linux	none (?)
alsa	unix	works	Arch Linux	libalsa
asihpi	unix	implemented	not tested	???
asio	win&mac	not implemented	---	ASIO SDK, ASIO Drivers, registration
coreaudio	mac	not supported	---	---
dsound	win	implemented	not tested	DirectX SDK
wmme	win	works	wine 1.3.4	none (?)
wasapi	win	probably works	wine 1.3.4	Vista SDK (only for MSVC)
wdmks	win	implemented	not tested	DirectX SDK

More info about the backends (best links I could dig up...):
jack: http://jackaudio.org/
oss: http://www.opensound.com/oss.html
alsa: http://www.alsa-project.org/
asihpi: http://www.audioscience.com/internet/download/linux_drivers. htm
asio: http://www.steinberg.net/en/company/developer.html
coreaudio: http://developer.apple.com/technologies/mac/audio-and-video. html
dsound: http://en.wikipedia.org/wiki/DirectSound
wmme: http://en.wikipedia.org/wiki/Windows_legacy_audio_components
#Multimedia_Extensions_.28MME.29
wasapi: http://msdn.microsoft.com/en-us/library/dd371455(VS.85).aspx
wdmks: http://www.portaudio.com/trac/browser/portaudio/trunk/src/ho stapi/wdmks/readme.txt

I'm looking forward to your comments

Best regards,
Honza

EDIT: Attachment removed in favor of newer version below.

Subject: Re: Sound device access system based on PortAudio
Posted by koldo on Thu, 04 Nov 2010 22:15:00 GMT
View Forum Message <> Reply to Message

Hello Honza

Good effort. I promise to test it tomorrow.

Subject: Re: Sound device access system based on PortAudio
Posted by koldo on Fri, 05 Nov 2010 22:49:26 GMT
View Forum Message <> Reply to Message

Hello Honza

I have tried in windows.

Using MinGW with WMME+WASAPI it plays sound, but it seems not to record.

Compiling with MSC I get "fatal error C1189: #error : RTTI must be enabled". It seems to happen as <Core/config.h> is included in a .c file.

Sorry for the test .

Subject: Re: Sound device access system based on PortAudio
Posted by dolik.rce on Fri, 05 Nov 2010 23:42:31 GMT
View Forum Message <> Reply to Message

Hi Koldo,

No need to be sorry, that is what tests are for

I tested with mingw in wine only, so I expected some problems. For MSC, there should be some way to turn the RTTI on (U++ itself uses it), any hints anybody?

I'll try to get some more extensive tests to figure out while it doesn't record. Meanwhile you can try to add compiler options for the plugin/portaudio package -DPA_LOG_API_CALLS and/or -DPA_ENABLE_DEBUG_OUTPUT, which turn on very verbose output to stdout (on windows you'll have to redirect it to file). It might give you some ideas about what is wrong. If you don't mind spending the time of course

Honza

Subject: Re: Sound device access system based on PortAudio
Posted by koldo on Sat, 06 Nov 2010 16:34:01 GMT
View Forum Message <> Reply to Message

Hello Honza

Here I enclose you the output.

File Attachments
1) out.txt, downloaded 1085 times

Subject: Re: Sound device access system based on PortAudio
Posted by dolik.rce on Wed, 10 Nov 2010 21:32:08 GMT
View Forum Message <> Reply to Message

Hi Koldo,

I didn't find out much from the output. Btw: what version of windows are you using?

Here I attach an updated version. The error reporting should be now better and I also left some debug logging on, so you can check the log-file after run.

I guess I will have to try the MSC builds myself... It turns out that the results are very compiler dependent. E.g. some features don't work correctly with Clang

Honza

EDIT: Attachment removed in favor of newer version below.

Subject: Re: Sound device access system based on PortAudio
Posted by koldo on Sun, 14 Nov 2010 20:46:09 GMT
View Forum Message <> Reply to Message

Hello Honza

I use XP SP3.

To make MSC run I just comment line #include <Core/config.h> in pa_upp.h. If not MSC complains because it is a C++ include used in .C files.

Well, here I enclose the output, truncated as the argument >out.txt does not serve.

The program plays but it does no record.

File Attachments

1) Out.7z, downloaded 317 times

Subject: Re: Sound device access system based on PortAudio
Posted by dolik.rce on Sun, 14 Nov 2010 21:27:36 GMT
View Forum Message <> Reply to Message

koldo wrote on Sun, 14 November 2010 21:46To make MSC run I just comment line #include <Core/config.h> in pa_upp.h. If not MSC complains because it is a C++ include used in .C files.
Fixed, thanks for info. Actually it is quite surprising as config.h contains only preprocessor commands, no code, so I would expect it to work both in C and in C++. Seems MSC has to always do something different

Also, how is it working now for you? I mean mainly the recording problem. Did you get any error messages in the log file or in the app?

Honza

EDIT: Attachment removed in favor of newer version below.

Subject: Re: Sound device access system based on PortAudio
Posted by koldo on Sun, 14 Nov 2010 21:38:13 GMT
View Forum Message <> Reply to Message

Hello Honza

The reason is here, from config.h:

```
#ifdef _MSC_VER
 #define COMPILER_MSC 1
 #ifndef _CPPRTTI
  #error  RTTI must be enabled !!!
```

Subject: Re: Sound device access system based on PortAudio
Posted by dolik.rce on Thu, 25 Nov 2010 20:25:48 GMT

View Forum Message <> Reply to Message

Hi everyone,

Good news, we are getting close to something potentially useful here I added basic support for .wav files and proof of concept function PlayWav(const char* filename). There is also an example how to read informations about the file. The wave file parsing is taken from Timothy J. Weber website. It is written in a code style amazingly similar to U++ and has IMHO quite permissive license (at least it is the closes thing to license I could find):Quote:PERMISSION

Copyright is retained by Timothy J. Weber. License is granted to use this source code for any purpose. That should be OK, for all possible uses within U++, right? The only sub-optimal thing about the code is that it uses STL, namely string, stack and vector. It probably can be converted to U++ later...

Now, before I'll continue adding more formats and standardizing the interface for playing audio files, I would like to discuss it here with you. My idea is to have a system similar to image formats - self registering plugins, each handling one audio format (or one family of formats). The question is how should the interfaces look.

The few ideas I have so far:Each format should support simple PlayXYZ(const char*) function, preferably non-blocking.
There should be a Play(const char*) function, that would recognize file type and call appropriate PlayXYZ().
Each format should provide an interface for reading samples from file and also for storing samples to file (decoding/encoding). I'm not sure if the should be separate (e.g. WavEncoder,WavDecoder) or joint (e.g. WavAudio) providing both.
All formats should be accessible in uniform way, that means probably common base class.
The audio files are by nature rather big, so it might be good idea to provide streaming capabilities (maybe even based on Stream class?) both for reading and writing.
What are your opinions, ideas, recommendations or hints? Any inputs are more than welcomed.

Furthermore, I would like to ask if there is some simple way to make a function non-blocking without requiring MT flag? The only way I can see is using the platform specific threading. Using callbacks is probably not an option here, as the audio playback requires quite high performance...

I'm looking forward to your feedback!
Honza

PS: The PlayWav is just a proof of concept, so don't expect much It blocks, until the sound is played, which is pretty bad behavior (the GUI doesn't respond). Also, it is quite easy to break and can't play everything (no support for compressed wavs or resampling if the content doesn't match your hardware capabilities)... But it is a start

File Attachments

1) sound.zip, downloaded 322 times

Subject: Re: Sound device access system based on PortAudio
Posted by dolik.rce on Mon, 13 Dec 2010 12:16:22 GMT

View Forum Message <> Reply to Message

Really no discussion or opinions on the topic of sound? I guess it is not because it is perfect already so it must be because nobody is interested enough If there is no interest I should probably aim my work in some more useful direction (do not take this as threat please).

Honza

PS: Some news about how the last version works on win32 would be helpful to...

Subject: Re: Sound device access system based on PortAudio
Posted by alendar on Mon, 13 Dec 2010 19:12:40 GMT

View Forum Message <> Reply to Message

I'm definitely interested. I'm looking for a simple way to play my own sounds for notify the user (me) when activities are complete.

EDIT:
I downloaded and tried to compile, but no luck.

I'm on XP SP3, with MSC9.

I'll download the portaudio package pa_stable_v19...
It may need a a dll/lib.

EDIT:
No bin distro, so getting Visual Studio 2005. Should compile??

Subject: Re: Sound device access system based on PortAudio
Posted by jerson on Tue, 14 Dec 2010 01:05:53 GMT

View Forum Message <> Reply to Message

I am very keen to use it, but, being relatively new to UPP, I did not interject. I have compiled and been successful in using the soundexample. I tried this on WinXP and mingw. I hope it can grow further. Sometimes I just gape in awe at the knowledge you guys have.

Subject: Re: Sound device access system based on PortAudio
Posted by dolik.rce on Tue, 14 Dec 2010 11:28:07 GMT
View Forum Message <> Reply to Message

Hi guys,
Thanks for the feedback

alendar wroteI'll download the portaudio package pa_stable_v19...
It may need a a dll/lib.

EDIT:
No bin distro, so getting Visual Studio 2005. Should compile??
No it should contain everything necessary in the source. It is posibly to link dynamically against library, but the package I published uses static linking. So the problem is probably somewhere else.

The biggest problem for me is that I don't have much access to windows systems, so I ussually test with mingw in wine That is why I am constantly asking you how it works with other systems and compilers.

I guess I should install MSVC in wine too to test the packages more thoroughly. Also I'm considering to upload it to bazaar, even though it is still just a stub.

jerson wroteI am very keen to use it, but, being relatively new to UPP, I did not interject. I have compiled and been successful in using the soundexample. I tried this on WinXP and mingw. I hope it can grow further. Sometimes I just gape in awe at the knowledge you guys have.Don't be mistaken, I gather the knowledge on the way I am glad to hear that at least the mingw install works.

Best regards,
Honza