

Developing LADSPA Plugins with Csound

Walsh, Lazzarini

csLADSPA

LADSPA plugin development with Csound

csLADSPA

LADSPA plugin development with Csound

Csound

- Csound is one of the most powerful and popular audio programming languages available to electroacoustic composers today.
- Its origins can be traced directly back to Max Mathews in Bell Labs and since its inception it has grown to become one of the most extensive computer music toolkits in development.
- Csound is free, cross platform, efficient and very well supported by a team of dedicated developers.

csLADSPA

LADSPA plugin development with Csound

More on Csound

- Csound is a command line program. Users write text files which contain instruments and a score.
- Csound instruments are created by connecting little 'black boxes' called 'opcodes'(operator code).
- Connecting the opcodes together creates your 'signal processor' and different connections will create different types of instruments. Several instruments can then be packed into an 'orchestra'
- Once the instruments are complete users compile them into audio.

csLADSPA

LADSPA plugin development with Csound

The Csound Host API

- An API (application programming interface) is an interface provided by a computer system, library or application, which provides users with a way of accessing functions and routines particular to the control program.
- API's provide developers with a means of harnessing an existing applications functionality within a host application.
- The Csound API can be used to control an instance of Csound through a series of different calling functions. In short, the Csound API makes it possible to harness all the power of Csound in your own applications.

csLADSPA

LADSPA plugin development with Csound

Audio plugins

- Plugins are 'libraries' which extend the functionality of a particular software known as the 'host application'.
- The 'host application' must be capable of loading these so-called libraries. Usually the host will search a particular folder and look for libraries contained within it. If the host finds a library it recognises it will load it.
- The advantage of using plugin is that third party developers can enhance the features of a particular application without having to rebuild it from scratch.

LADSPA

- Linux Audio Developers Simple Plugin API
- It came to fruition in early 2000 following the combined efforts of Paul Davis, Richard W.E. Furse, and Stefan Westerfield.
- They created a framework for developing audio plugins which can be loaded by a wide range of host applications.
- LADSPA plugins can run on most major platforms, not just *nix operating systems.

csLADSPA

LADSPA plugin development with Csound

csLADSPA

- csLADSPA is a tool for developing audio plugins
- It simplifies the process of developing audio plugins by allowing users to write plugin code in Csound rather than in a lower level language such as C.
- csLADSPA provides a link between the Csound API, LADSPA and the host application.
- csLADSPA remains true to one of the main objectives of the LADSPA project i.e., to create a 'simple' architecture for the development of plugins...

csLADSPA

LADSPA plugin development with Csound

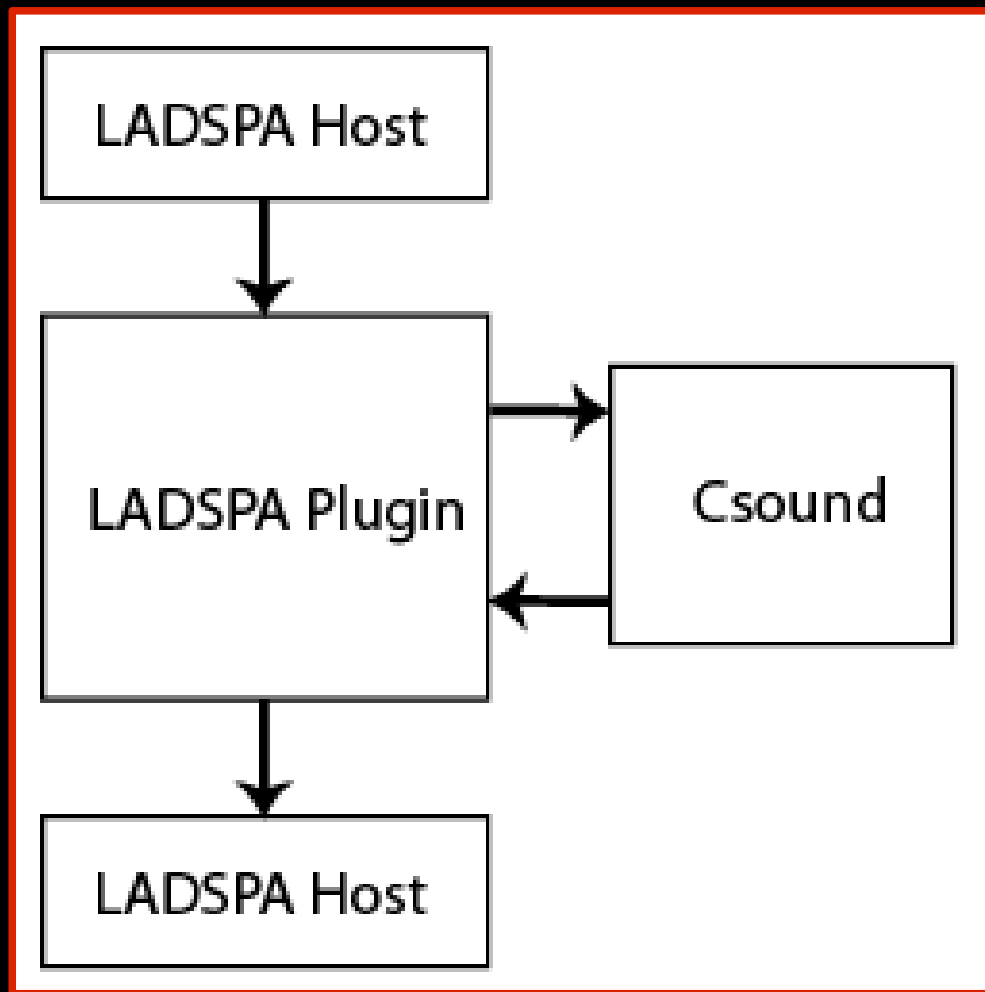
Inside csLADSPA

- The main difference between csLADSPA and a typical LADSPA plugin is that csLADSPA uses Csound instruments to process the audio rather than a dedicated processing function.
- Whenever a LADSPA host application is launched it loads the csLADSPA plugin. Once the csLADSPA is loaded it searches set directories for Csound files.
- For each Csound instrument it finds csLADSPA loads a unique plugin which will interface with that particular instrument.

csLADSPA

LADSPA plugin development with Csound

- A basic model of how the csLADSPA work is shown in below



csLADSPA

LADSPA plugin development with Csound

Getting started with csLADSPA

- The csLADSPA plugin and the Csound source files must reside in the folder pointed to by the LADSPA_PATH environment variable.
- In order to keep things simple, csLADSPA only works with the unified Csound file format.

csLADSPA Tags

- In order for csLADSPA to load Csound files as plugins the user must specify some basic information about the plugin in their Csound files.
- This is done in a special <csLADSPA> section of the Csound file.
- Within the csLADSPA section of the Csound file users can specify the control ports they will need in order to interact with their Csound code when running the plugin through a host.

- Every csLADSPA plugin must specify the following:

Tags	Description
Name	The name of the plugin as it will appear in the host application
Maker	Author of plugin
UniqueID	ID given to plugin, each plugin should use a unique ID.
Copyright	Copyright/Licence notice

- If users wish to add controls to their plugins they can do so using the following tags:

Tags	Description
ControlPort	The name of the control as it appears when the plugin is run and the name of the channel which Csound will retrieve the data on. The two names should be separated by a ' ' symbol.
Range	Plugin max/min range. Again the two values are separated by a ' ' symbol. If users wish to controls to respond logarithmically they can add a '&log' after they specify the range values.

Examples

```
<csLADSPA>
Name=Gain Plugin
Maker=John Doe
UniqueID=1049
Copyright=None
ControlPort=Gain|gain
Range=0|2
</csLADSPA>
<CsoundSynthesizer>
<CsInstruments>
sr = 44100
ksmps = 10
nchnls = 1

instr 1
    kGain chnget "gain"
    ain in
    out ain*kGain
endin

</CsInstruments>
<CsScore>
i1 0 3600
</CsScore>
</CsoundSynthesizer>
```

C/C++ equivalent

csLADSPA

LADSPA plugin development with Csound

```
<csLADSPA>
Name=MyReverb
Maker=Rory Walsh
UniqueID=2049
Copyright=None
ControlPort=ReverbTime|rtime
Range=0|5
</csLADSPA>
<CsoundSynthesizer>
<CsInstruments>
sr = 44100
ksmps = 10
nchnls = 1

instr 1
    ktime chnget "rtime"
    ain in
    arev reverb ain, ktime
    out arev
endin

<CsScore>
</CsInstruments>
i1 0 3600
</CsScore>
</CsoundSynthesizer>
```

csLADSPA

LADSPA plugin development with Csound

```
<cLADSPA>
Name=Flanger
Maker=Rory Walsh
UniqueID=1099
Copyright=None
ControlPort=Flange Depth|depth
Range=0|5
ControlPort=Flange Rate|rate
Range=0|10
</cLADSPA>
<CsoundSynthesizer>
<CsOptions>
-d
</CsOptions>
<CsInstruments>
sr = 44100
ksmps = 10
nchnls = 1

instr 1
kdelttime chnget "depth"
krate chnget "rate"
ain in
a1 oscil kdelttime, 1, 1
ar vdelay3 ain, a1+kdelttime, 1000
out ar+ain
endin

</CsInstruments>
<CsScore>
f1 0 1024 10 1
i1 0 3600
</CsScore>
</CsoundSynthesizer>
```

csLADSPA

LADSPA plugin development with Csound

```

<cLADSPA>
Name=PVSBlur
Maker=Rory Walsh
UniqueID=9991
Copyright=None
ControlPort=Max Delay|del
Range=0|1
ControlPort=Blur Time|blur
Range=0|10
</cLADSPA>
<CsoundSynthesizer >
<CsOptions>
-d
</CsOptions>
<CsInstruments>
sr = 44100
ksmps = 10
nchnls = 1

instr 1
imaxdel chnget "del"
iblurtime chnget "blur"
asig in
fsig pvsanal asig, 1024, 256, 1024, 1
ftps pvsblur fsig, 0.2, 0.2
atps pvsynth ftps
out atps
endin

</CsInstruments>
<CsScore>
f1 0 1024 10 1
i1 0 3600
</CsScore>
</CsoundSynthesizer >

```

csLADSPA

LADSPA plugin development with Csound

Conclusion

- To date csLADSPA has been used both as a creative tool and as a pedagogical utility used in the teaching of DSP techniques.
- It has been fully tested in real-time using Jack-Rack and in non-realtime mode using Audacity.
- With regards to future developments the authors are currently working on a better error-checking system and multichannel support is also being investigated.