

---

# **cython-sounddevice**

*Release 0.0.1*

**Apr 17, 2020**



---

# Contents

---

<b>1</b>	<b>cython-sounddevice</b>	<b>1</b>
1.1	Description . . . . .	1
1.2	Links . . . . .	1
1.3	Usage . . . . .	1
1.4	Dependencies . . . . .	1
1.5	Installation . . . . .	2
1.5.1	Linux . . . . .	2
1.5.2	Windows . . . . .	2
1.5.3	MacOS . . . . .	2
1.6	License . . . . .	2
1.6.1	Reference . . . . .	2
<b>2</b>	<b>Indices and tables</b>	<b>7</b>
	<b>Index</b>	<b>9</b>



### 1.1 Description

Python bindings for the `PortAudio` library to interface with audio streams. This project was inspired by `python-sounddevice`, but uses `Cython` instead of `CFFI`.

This allows for use in other `Cython` projects needing audio I/O without the performance penalty of the switching between Python and C/C++ contexts. All of the necessary classes, functions and data types have shared declarations for this purpose.

### 1.2 Links

- Documentation
  - <https://cython-sounddevice.readthedocs.io/en/latest/>
- Source Code
  - <https://github.com/nocarryr/cython-sounddevice>

### 1.3 Usage

*TODO*

### 1.4 Dependencies

- `Cython`  $\geq 0.29.1$
- `PortAudio`

## 1.5 Installation

*TODO*

### 1.5.1 Linux

```
sudo apt-get install portaudio19-dev
```

### 1.5.2 Windows

*TODO*

### 1.5.3 MacOS

*TODO*

## 1.6 License

See the [LICENSE](#) file for license information (GPLv3).

### 1.6.1 Reference

**cysounddevice.devices module**

**PortAudio class**

**HostApilInfo class**

**DeviceInfo class**

**cysounddevice.streams module**

**Stream class**

**StreamInfo class**

**StreamCallback class**

**C-API**

**CallbackUserData**

Container for data used in `_stream_callback()`

int **input\_channels**

Number of input channels

**int output\_channels**  
Number of output channels

*SampleBuffer*\* **in\_buffer**  
Pointer to a *SampleBuffer* to write input data to

*SampleBuffer*\* **out\_buffer**  
Pointer to a *SampleBuffer* to read output data from

**int \_stream\_callback** (const void\* *in\_bfr*, void\* *out\_bfr*, unsigned long *frame\_count*, const PaStreamCallbackTimeInfo\* *time\_info*, PaStreamCallbackFlags *status\_flags*, void\* *user\_data*)  
Callback function that reads and writes input/output data using the *SampleBuffer* pointers stored in *user\_data* as *CallbackUserData*

## cysounddevice.buffer module

### StreamBuffer class

### StreamInputBuffer class

### StreamOutputBuffer class

## C-API

### SampleBuffer

A buffering structure with preallocated memory for use in *\_stream\_callback*

*BufferItem*\* **items**  
Buffer array of *BufferItem*

Py\_ssize\_t **length**  
Number of *items* to allocate

Py\_ssize\_t **itemsize**  
Size in bytes per sample

Py\_ssize\_t **item\_length**  
Number of samples to allocate for each *BufferItem* (block size)

Py\_ssize\_t **nchannels**  
Number of channels

Py\_ssize\_t **write\_index**  
Index of the next item to use for writing

Py\_ssize\_t **read\_index**  
Index of the next item to use for reading

BLOCK\_t **current\_block**  
The current block of samples

int **read\_available**  
Number of items available to read from

int **write\_available**  
Number of items available to write to

### BufferItem

A single item used to store data for *SampleBuffer*

*SampleTime\_s* **start\_time**

The time of the first sample in the item's buffer, as reported by PortAudio

Py\_ssize\_t **index**

Index of the item within its parent *SampleBuffer*

Py\_ssize\_t **length**

Number of samples the item contains

Py\_ssize\_t **itemsize**

Size in bytes per sample

Py\_ssize\_t **nchannels**

Number of channels

Py\_ssize\_t **total\_size**

The total size in bytes to allocate “ length \* itemsize \* nchannels “

char \***bfr**

Pointer to the preallocated buffer

*SampleBuffer\** **sample\_buffer\_create** (*SampleTime\_s* start\_time, Py\_ssize\_t length, Py\_ssize\_t nchannels, Py\_ssize\_t itemsize)

Creates a *SampleBuffer* and child items (*BufferItem*), allocating all required char buffers.

void **sample\_buffer\_destroy** (*SampleBuffer\** bfr)

Deallocates the given *SampleBuffer* and all of its child items.

int **sample\_buffer\_write** (*SampleBuffer\** bfr, const void \*data, Py\_ssize\_t length)

Copy the given data to the next available item in the given *SampleBuffer*. If no items are available to write (the buffer is full), no data is copied.

Returns 1 if successful

*SampleTime\_s\** **sample\_buffer\_read** (*SampleBuffer\** bfr, char \*data, Py\_ssize\_t length)

Copy data from the next available item into the given buffer.

**Returns:** A *SampleTime\_s* pointer to the *BufferItem.start\_time* describing the source timing of the data. If no data is available, returns NULL.

*SampleTime\_s\** **sample\_buffer\_read\_sf32** (*SampleBuffer\** bfr, float[:, :] data)

Copy stream data from a *SampleBuffer* into a float array

Deinterleaves the stream and casts it to 32-bit float. A typed memoryview may be used.

The sample format must be paFloat32.

**Returns:** A *SampleTime\_s* pointer to the *BufferItem.start\_time* describing the source timing of the data. If no data is available, returns NULL.

## cysounddevice.types module

### SampleTime class

### C-API

#### SampleFormat

PaSampleFormat **pa\_ident**

Py\_ssize\_t **bit\_width**



bint **is\_signed**  
bint **is\_float**  
bint **is\_24bit**  
void\* **dtype\_ptr**

**SampleTime\_s**

PaTime **pa\_time**  
Time in seconds  
PaTime **time\_offset**  
Time offset in seconds  
SAMPLE\_RATE\_t **sample\_rate**  
Sample rate  
Py\_ssize\_t **block\_size**  
Number of samples per block  
BLOCK\_t **block**  
Block count  
Py\_ssize\_t **block\_index**  
Index within the block



## CHAPTER 2

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`



## Symbols

`_stream_callback` (*C function*), 3

## B

`ButtonItem` (*C type*), 3  
`ButtonItem.bfr` (*C member*), 4  
`ButtonItem.index` (*C member*), 4  
`ButtonItem.itemsize` (*C member*), 4  
`ButtonItem.length` (*C member*), 4  
`ButtonItem.nchannels` (*C member*), 4  
`ButtonItem.start_time` (*C member*), 3  
`ButtonItem.total_size` (*C member*), 4

## C

`CallbackUserData` (*C type*), 2  
`CallbackUserData.in_buffer` (*C member*), 3  
`CallbackUserData.input_channels` (*C member*), 2  
`CallbackUserData.out_buffer` (*C member*), 3  
`CallbackUserData.output_channels` (*C member*), 2

## S

`sample_buffer_create` (*C function*), 4  
`sample_buffer_destroy` (*C function*), 4  
`sample_buffer_read` (*C function*), 4  
`sample_buffer_read_sf32` (*C function*), 4  
`sample_buffer_write` (*C function*), 4  
`SampleBuffer` (*C type*), 3  
`SampleBuffer.current_block` (*C member*), 3  
`SampleBuffer.item_length` (*C member*), 3  
`SampleBuffer.items` (*C member*), 3  
`SampleBuffer.itemsize` (*C member*), 3  
`SampleBuffer.length` (*C member*), 3  
`SampleBuffer.nchannels` (*C member*), 3  
`SampleBuffer.read_available` (*C member*), 3  
`SampleBuffer.read_index` (*C member*), 3  
`SampleBuffer.write_available` (*C member*), 3  
`SampleBuffer.write_index` (*C member*), 3

`SampleFormat` (*C type*), 4  
`SampleFormat.bit_width` (*C member*), 4  
`SampleFormat.dtype_ptr` (*C member*), 5  
`SampleFormat.is_24bit` (*C member*), 5  
`SampleFormat.is_float` (*C member*), 5  
`SampleFormat.is_signed` (*C member*), 4  
`SampleFormat.pa_ident` (*C member*), 4  
`SampleTime_s` (*C type*), 5  
`SampleTime_s.block` (*C member*), 5  
`SampleTime_s.block_index` (*C member*), 5  
`SampleTime_s.block_size` (*C member*), 5  
`SampleTime_s.pa_time` (*C member*), 5  
`SampleTime_s.sample_rate` (*C member*), 5  
`SampleTime_s.time_offset` (*C member*), 5