Contents

1 A simple audio path specification language

1.1 Introduction

Audio paths are specified using a small language, dubbed *AuPaL* (Audio Path Language). This language is specified to make use of a compact binary representation, while attempting to keep it readable by using ASCII command codes.

1.1.1 DCP register

Audio path specifications must be sent to Streaming Board register 82.

1.1.2 *AuPaL* compiler

There is a compiler, named `aupalc.py`, for transforming a text representation of *AuPaL* to proper binary *AuPaL*. It can be used for creating test vectors and for *AuPaL* generator verification. Requires Python 3.

2 Audio paths mini DSL specification

2.1 Command codes

2.1.1 Command code "I" — Instance

Introduce physical appliance with a well-known ID under a given name. In general, the appliance IDs sent via DCP register 87 are well-known IDs. IDs for appliances which do not host a Streaming Board (e.g., amplifiers) are to be defined.

Note that—in congruence with reality—it is possible for any appliance type to be present multiple times. For instance, multiple amplifiers of the same type could be connected to a player. Each such physical appliance must be introduced with its own `I` command.

As a special case, the appliance ID and the appliance name may both be empty strings. In this case, all known devices and its audio connections are erased (shortcut for multiple `i` commands).

Parameters:

- Appliance ID: *ASCIIZ* — A well-known ID from the models database.
- Appliance name: *ASCIIZ* — An arbitrary, preferably short, non-empty name to identify the specific appliance.

Examples:

- `IMP3100HV\0self\0`
- `IPA3100HV\0pa\0`
- `I\0\0IMP3100HV\0self\0`

2.1.2 Command code "i" — Instance removed

Report disconnection of previously available appliance. All previously declared audio connections from and to that appliance are automatically removed.

Parameters:

- Appliance name: *ASCIIZ* — Name of an appliance which has disappeared.

Examples:

- `ipa\0`

2.1.3 Command code "C" – Physical audio connection from sink to source

Declare a physical connection from one appliance's output (defined as audio sink in the model for that appliance) to another appliance's input (defined as an audio source in that appliance's model).

Connections involving unknown sink IDs or source IDs are simply ignored. If these IDs become available by later declarations, then the connections will not be made.

Parameters:

- Sink ID: *ASCIIZ* — Name of an audio sink, qualified with an appliance name.
- Source ID: *ASCIIZ* — Name of an audio source, qualified with an appliance name.

Examples:

- `Cself.analog_line_out\0pa.analog_in_1\0`

2.1.4 Command code "c" – Remove physical audio connection

State that a physical connection from one appliance's output to another appliance's input has been unmade. This is the counterpart to the `C` command.

Connections involving unknown sink IDs or source IDs are simply ignored.

Parameters:

- Sink ID: *ASCIIZ* — Name of an audio sink, qualified with an appliance name. This parameter may also be an appliance name or the empty string, in which case the parameter acts as a filter for outgoing connections from a specific appliance (non-empty appliance name), or all appliances (empty string).
- Source ID: *ASCIIZ* — Name of an audio source, qualified with an appliance name. Similar to the sink ID, this parameter may also be an appliance name or the empty string to act as a filter.

Examples:

- `cself.analog_line_out\0pa.analog_in_1\0`
- `cself\0pa\0`
- `c\0\0`

2.1.5 Command code "S" — Set values

Report values of any controls in an appliance, using names and types as defined in the model for the appliance.

Values of controls which are defined in the appliance model, but not sent in this command, are explicitly forgotten and marked as unknown. This is handy in cases where the model defines controls which are not yet supported by the appliance firmware.

Note that this property makes this command an absolute one. It is not suited for incremental value updates where only one or few controls are to be reported; see commands `U` and `u` for commands supporting this use case.

Controls not defined in the appliance model are simply ignored. This allows interoperation of appliance firmware and Streaming Board even if their common understanding of appliance internals run out of sync for some reason.

Controls with values set out of range (according to the model) are marked as unknown.

Parameters:

- Element ID: *ASCIIZ* — Name of an element, qualified with an appliance name.
- Controls count: *byte* — Number of controls to be set for the element. Set this parameter to 0 to forget the values of all controls in the specified element. In the unlikely case that more than 255 controls need to be set for a single element, use `S` for the first 255 controls, followed by `U` commands for the remaining controls.
- Control ID 1: *ASCIIZ* — Name of a control in the specified element.
- Value 1: *variant* — Value of the specified control, using the type defined in the model.
- Control ID 2: *ASCIIZ* — Name of the seconds control, if any.
- …

Examples:

- `Sself.dsp\0\x02filter\0siir_bezier\0phase_invert\0b\x01`
- `Sself.dsd_out_filter\0\x01mode\0snormal\0`

2.1.6 Command code "U" — Update values

Like `S`, but keep all values for controls not mentioned in the `U` command the way they were before. This command is an incremental version of `S`.

Parameters are exactly the same as for `S`, except for the control count: if this is set to 0, then `U` acts as a NOP and does not forget any values.

2.1.7 Command code "u" — Update single value

Short version of `U` for single value updates.

This is the shortest possible encoding for reporting incremental updates of single values. However, better use `U` for reporting elements with two or more controls changed.

Parameters:

- Element ID: *ASCIIZ* — Name of an element, qualified with an appliance name.
- Control ID: *ASCIIZ* — Name of a control in the specified element.
- Value: *variant* — Value of the specified control, using the type defined in the model.

Examples:

- `uself.dsp\0filter\0siir_bezier\0`

- `uself.dsd_out_filter\0mode\0snormal\0`

2.1.8 Command code "d" — Unset single value

Reset a the value of a single control to unknown state.

Parameters:

- Element ID: *ASCIIZ* — Name of an element, qualified with an appliance name.
- Control ID: *ASCIIZ* — Name of a control in the specified element.

Examples:

- `dself.dsd_out_filter\0mode\0`

2.2 Variant types

2.2.1 Encoding of variant types

Variant types are encoded using a type code followed by the value encoded according to the type code. Type codes are defined based on D-Bus types (see https://dbus.freedesktop.org/doc/dbus-specification.html#idm487).

Here is a complete list of supported types:

Type code	Type description
s	*ASCIIZ* string, a plain ASCII string with a 0 terminator
b	Boolean value, either 0 or 1 stored as an 8-bit value
Y	Signed 8-bit value
y	Unsigned 8-bit value
n	Signed 16-bit value
q	Unsigned 16-bit value
i	Signed 32-bit value
u	Unsigned 32-bit value
x	Signed 64-bit value
t	Unsigned 64-bit value
D	T+A 14-bit fix point value

2.2.2 Mapping of control types to variant types

Control type	Type code	Notes
on_off	b	off = 0, on = 1
choice	s	Choices are sent verbatim
range	*any*	The type is the `value_type` defined in the model