

# Configuração de dispositivos de áudio: Raspberry Pi 3

**Autora:** Bárbara Circe

19 de agosto de 2021

# Sumário

<b>1</b>	<b>Introdução</b>	<b>4</b>
1.1	Conhecendo seu Raspberry Pi por dentro	4
1.2	Gerenciamento de áudio e outros pacotes	6
1.3	Ferramentas de ajuda	7
<b>2</b>	<b>GUI-desktop</b>	<b>8</b>
2.1	Gnome Alsamixer	9
<b>3</b>	<b>PortAudio</b>	<b>10</b>
<b>4</b>	<b>ALSA</b>	<b>10</b>
4.1	Checando dependências	11
4.2	Configurando o ALSA: alsamixer	12
4.3	Configurando o ALSA: asound.config	16
<b>5</b>	<b>PulseAudio</b>	<b>18</b>
5.1	Configurando o PulseAudio	20
5.2	Expondo o PulseAudio para o ALSA	23
<b>6</b>	<b>Conclusões</b>	<b>27</b>
	<b>Referências</b>	<b>28</b>

## Lista de Códigos

1	Comandos que estampam informações sobre o sistema. . . . .	5
2	Comandos para atualizar o sistema operacional. . . . .	6
3	Sintaxe e exemplo de uso do comando <code>arecord</code> para captura de áudio. . . . .	8
4	Linhas de código para testar alto-falantes com ruído branco e arquivo <code>.wav</code> de áudio, respectivamente. . . . .	8
5	Instalação ou atualização do Gnome Alsa-mixer. . . . .	9
6	Ordem a ser executada na linha de comando para instalação do PortAudio. . .	10
7	Comando para atualização do pacote <code>alsa-utils</code> . . . . .	11
8	Comandos que listam dispositivos para reprodução e captura de áudio, respectivamente. . . . .	12
9	Comandos para abrir o <i>mixer</i> do ALSA e salvar suas configurações, respectivamente. . . . .	13
10	Linhas de comando para se chegar ao arquivo <code>asound.config</code> . . . . .	16
11	Comandos para procurar o arquivo <code>asound.config</code> . . . . .	16
12	Maneiras de testar a captura e reprodução pelo seu dispositivo de áudio. . . .	17
13	Comandos para checar, instalar ou atualizar o PulseAudio. . . . .	19
14	<code>dd</code> . . . . .	20
15	Sintaxe para alteração de <i>inputs</i> e <i>outputs</i> para <code>pacmd</code> e <code>pactl</code> . . . . .	22
16	Exemplos de como modificar <i>sinks</i> e <i>sources</i> com <code>pacmd</code> e <code>pactl</code> . . . . .	22
17	Códigos para checar a relação "pulse-alsa". . . . .	23
18	Comandos para acessar o arquivo <code>asound.config</code> . . . . .	25
19	Perfis de exposição do PulseAudio ao ALSA, usando o <i>plugin</i> "pulse" (adaptado de [1]). . . . .	25

# 1 Introdução

Este Tutorial *não* possui a intenção de auxiliar no primeiro contato do usuário com o Raspberry Pi, limitando-se ao passo a passo do reconhecimento e utilização de dispositivos de áudio. Para a utilização correta deste documento, algumas informações fazem-se necessárias:

- identificação correta do modelo;
- verificação da versão do tipo de sistema operacional instalado;
- uma pesquisa prévia a respeito do dispositivo que você deseja conectar.

A seguir será abordado como saber qual modelo, sistema operacional e quais códigos necessários para obtenção dessas informações, além de especificar o Raspberry Pi utilizado para a construção deste Tutorial.

## 1.1 Conhecendo seu Raspberry Pi por dentro

O Raspberry Pi<sup>1</sup> é um pequeno computador que pode existir em diversos modelos<sup>2</sup>, como os exibidos na Figura 1. Cada um desses computadores utilizam um cartão SD como forma de armazenamento e os modelos mais recentes possuem diversos tipos de entradas como USB, Ethernet, HDMI, GPIO (do inglês *General Purpose Input/Output*) e suporte para Wi-Fi, Bluetooth e HATs (*Hardware Attached on Top*). Normalmente o sistema operacional recomendado para utilização em Raspberry Pi é chamado de Raspbian, uma distribuição Linux baseada em Debian e exclusiva para o produto.

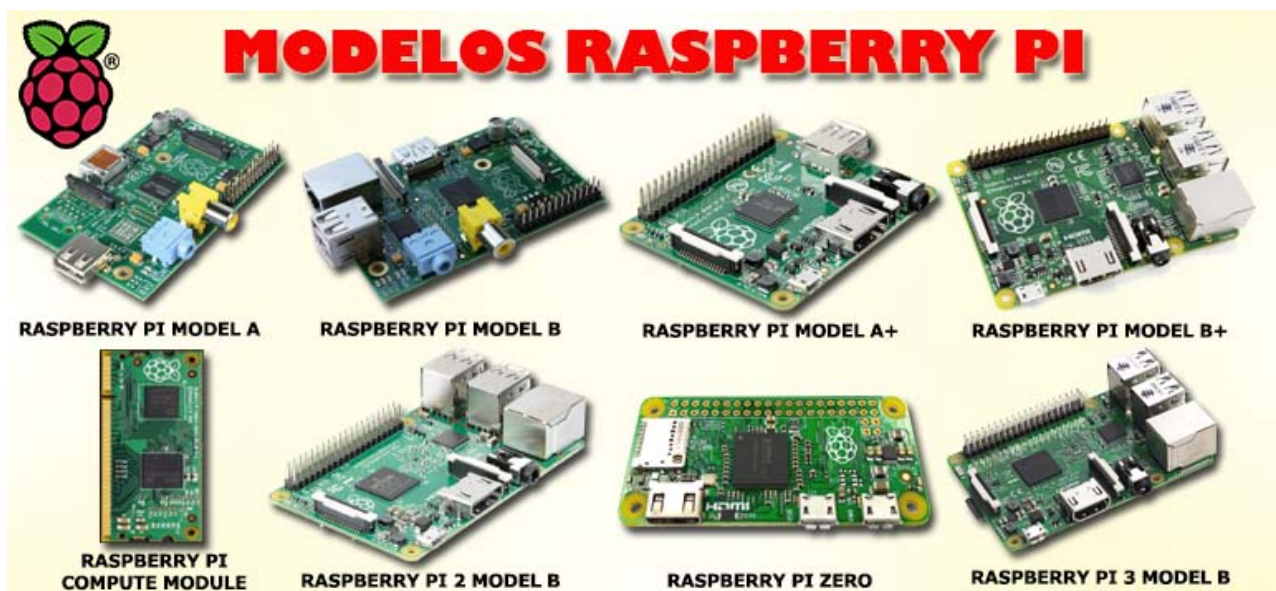


Figura 1: Diferentes modelos de Raspberry Pi [2].

Para não haver dúvidas, simbolizaremos com \$ no início de cada linha um comando a ser escrito no *bash* do Raspbian. As linhas em que não há o símbolo \$ são o *output* das

<sup>1</sup>Mais informações no site <https://www.raspberrypi.org/>.

<sup>2</sup>Poucos dias antes da publicação da primeira versão desse Tutorial, foi lançado o Raspberry Pi 4: <https://www.raspberrypi.org/products/raspberry-pi-4-model-b> (e vale muito a pena conferir as melhorias).

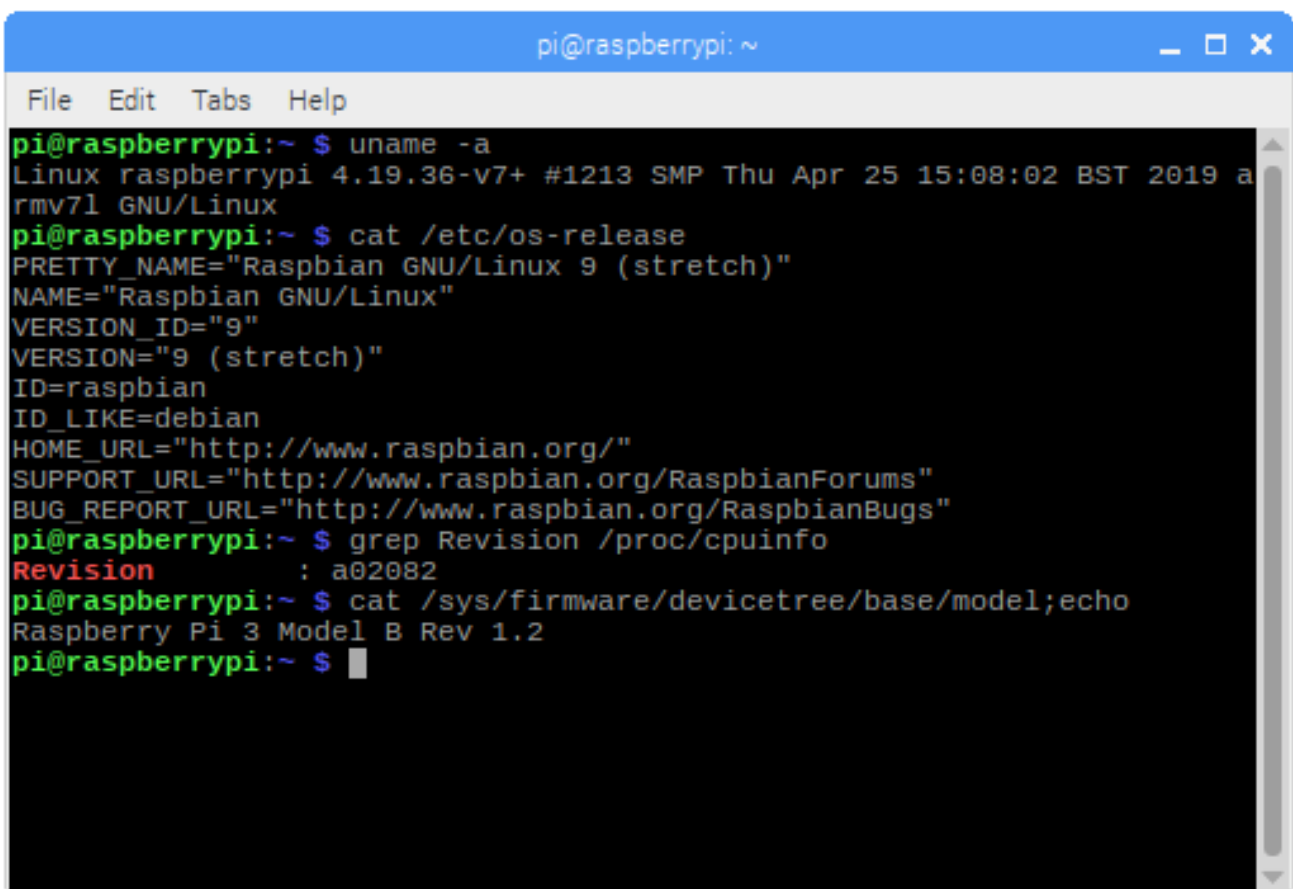
entradas inseridas no *bash*, ou podem tratar-se de códigos inseridos no editor ou no console do Python (que tem aparência diferente). Alguns comandos que nos permitem conhecer as características do Raspberry podem ser encontrados a seguir, no Código 1.

```
1 $ uname -a
2 $ cat /etc/os-release
3 $ grep Revision /proc/cpuinfo
4 $ cat /sys/firmware/devicetree/base/model; echo
```

Código 1: Comandos que estampam informações sobre o sistema.

Abra o terminal com o comando `Ctrl+Alt+T` e digite o conteúdo do Código 1. O resultado da inserção dos comandos pode ser visto na Figura 2. O primeiro comando (`uname -a`) imprime na tela informações do sistema: versão do *kernel*, *hardware* e outras características. Como resposta, obtemos "armv7l", que é a arquitetura utilizada para este modelo de Raspberry, significando dentre outras coisas, um processador de 32-bit. A estrutura ARM é a base da maioria dos processadores do RPi (abreviação de Raspberry) porém apenas os modelos armv8 introduzem a opção 64-bit (como é o caso do Raspberry Pi 4).

O segundo comando, (`cat /etc/os-release`) informa o tipo e a versão do sistema operacional utilizado, que neste caso é o Raspbian Stretch versão 9. Há ainda o Raspbian Jessie, que é um modelo mais antigo e nesse caso, recomenda-se atualizar o sistema (que será mostrado nas próximas páginas) ou buscar métodos específicos para essa versão de sistema operacional (SO).



```
pi@raspberrypi: ~
File Edit Tabs Help
pi@raspberrypi:~ $ uname -a
Linux raspberrypi 4.19.36-v7+ #1213 SMP Thu Apr 25 15:08:02 BST 2019 a
rmv7l GNU/Linux
pi@raspberrypi:~ $ cat /etc/os-release
PRETTY_NAME="Raspbian GNU/Linux 9 (stretch)"
NAME="Raspbian GNU/Linux"
VERSION_ID="9"
VERSION="9 (stretch)"
ID=raspbian
ID_LIKE=debian
HOME_URL="http://www.raspbian.org/"
SUPPORT_URL="http://www.raspbian.org/RaspbianForums"
BUG_REPORT_URL="http://www.raspbian.org/RaspbianBugs"
pi@raspberrypi:~ $ grep Revision /proc/cpuinfo
Revision          : a02082
pi@raspberrypi:~ $ cat /sys/firmware/devicetree/base/model; echo
Raspberry Pi 3 Model B Rev 1.2
pi@raspberrypi:~ $
```

Figura 2: Resultado do *bash* após inserção dos comandos vistos no Código 1.

O terceiro comando (`grep Revision /proc/cpuinfo`) informa o código de revisão do seu RPi. Com esta informação você pode descobrir o modelo e mais informações como a memória RAM e o país de origem. Caso você esteja apenas interessada ou interessado em saber o modelo do seu RPi, o quarto comando prontamente acaba com sua dúvida. Neste caso, o modelo utilizado para a criação deste Tutorial é o Raspberry Pi B.

```
1 $ sudo apt-get update
2 $ sudo apt-get upgrade #cuidado!
```

Código 2: Comandos para atualizar o sistema operacional.

Considera-se uma boa prática a atualização do sistema e seus pacotes. Caso seu sistema não esteja em dia, orienta-se a atualização conforme descrito no Código 2, porém, *cuidado!* Recomenda-se fazer uma cópia de segurança antes de atualizar todo sistema. Alguns problemas podem decorrer desta atualização, como por exemplo um *login loop*, que impede a utilização normal da interface gráfica do *desktop*. Caso ocorra, a autora deste Tutorial poderá te ajudar (também aconteceu com ela).

## 1.2 Gerenciamento de áudio e outros pacotes

Há diferentes formas de utilização de áudio com Raspberry Pi: podemos utilizar a saída *on-board* através do *jack* de áudio da placa, ou por GPIO, seja por cabos ou utilização de HATs. Também é possível adicionar dispositivos via USB, como interfaces ou placas mais complexas de áudio, ou conectar uma tela através da porta HDMI. Além disso tudo, a *forma de utilização* dos dispositivos de áudio também pode ter outras dependências: por exemplo, se você quer utilizar uma placa de áudio que obedeça a comandos do Python, alguns pacotes para gerenciamento dessa relação fazem-se necessários, como o módulo `sounddevice` e o `PortAudio`.

Durante a escolha da melhor maneira de gerenciar, encontramos muitas limitações em diferentes métodos. Como um breve exemplo podemos citar:

- o RPi possui apenas saída, em que podem haver problemas com tempo real e projetos que incluam latências muito baixas;
- aparentemente, o mesmo áudio *on-board* possui apenas 11 bits [3];
- HATs são uma ótima opção, porém é muito mais fácil encontrar aqueles que apenas gerenciam a saída ao invés de gerenciar ambos; (a Referência [4] pode te dar um bom norte);
- algumas interfaces de áudio não são aceitas pelo sistema, porém se o `alsamixer` (ou `Gnome AlsaMixer`) reconhecerem mesmo não reproduzindo ou gravando nada ainda, é um bom sinal: o problema pode ser a falta de algum servidore (PulseAudio, JACK) ou de uma biblioteca como o `PortAudio`.

As Referências [3] e [5] trazem informações valiosas e que podem poupar bastante do seu tempo: um repositório de *software* de áudio para RPi (além de detalhes sobre sua utilização) e dispositivos USB que verificadamente funcionam com o Raspberry Pi, respectivamente.

## 1.3 Ferramentas de ajuda

Aqui apresentaremos diferentes maneiras que podem auxiliar a se chegar no reconhecimento e utilização do dispositivo de áudio que você deseja conectar. Alguns comandos que podem ser úteis para a investigação:

- i. `<nome-do-comando> --help:`  
estampa na tela a ajuda e as opções de utilização do comando.  
Exemplo: `alsamixer --help;`
- ii. `man <nome-do-comando>:`  
abre o manual de referência do comando desejado.  
Exemplo: `man pulseaudio;`
- iii. `lsusb:`  
mostra na tela dispositivos USB conectados ao RPi;
- iv. `sudo find <diretório-de-pesquisa> -name <nome-do-arquivo.extensão>:`  
procura um tipo de arquivo pelo nome em um certo diretório. Substituindo o nome do arquivo por um asterisco, buscaremos todos aqueles que possuem a mesma extensão.  
Exemplo 1: `sudo find / -name asound.config.`  
Exemplo 2: `sudo find / -name *.config` (todos os arquivos .config);
- v. `whereis <arquivos-ou-programas>:`  
printa na tela diretórios que contenham arquivos ou programas com o nome procurado.  
Exemplo: `whereis python3.5;`
- vi. `cd <diretório>:`  
seleciona o diretório inserido como sendo o ativo.  
Exemplo: `cd /home/pi/Audio;`
- vii. `dir:`  
comando que mostra na tela o conteúdo do diretório ativo.
- viii. `nano:`  
ferramenta de leitura e edição de arquivos de texto.  
Exemplo: `nano asound.config`
- ix. `sudo:`  
comando que força execução de linhas do *bash* como "super usuário"<sup>a</sup>

<sup>a</sup>O comando `sudo` vem da expressão *superuser do*.

Será recorrente dizer que é uma boa prática testar a saída ou entrada do dispositivo de áudio após alterações nas configurações: como testar no *bash* do Linux (conforme os Códigos 3 e 8), no Youtube e no Python. Em algumas ocasiões podemos fazer funcionar no Youtube mas não no Python, por exemplo, e por isso testar de diferentes maneiras é importante para a investigação.



```

1 $ #sintaxe:
2 $ arecord -D plughw:<indice-dispositivo> --duration=<seg> -f cd
3 /diretorio/arquivo.wav
4 $ #exemplo:
5 $ arecord -D plughw:1 --duration=7 -f cd /home/pi/teste.wav

```

Código 3: Sintaxe e exemplo de uso do comando `arecord` para captura de áudio.

As porções do Código 3 (acima) que precisam de um pouco mais de explicação são: a) `-D`: significa dispositivo (*device*) indicado pelo comando b) `plughw:1`, que seleciona o dispositivo pelo número do índice (nesse caso o índice 1, que significa a escolha do *segundo* dispositivo de áudio); c) o pedaço `-f cd` configura o formato de gravação como sendo 16-bit, frequência de amostragem 44100 Hz e estéreo.

Já o Código 4 mostra duas maneiras de se testar alto-falantes: uma com ruído branco, não muito recomendável mas podendo vir a servir (cuidado com o susto!), e utilizando-se um arquivo `.wav` (e não `.mp3`!) [6]. Na primeira linha a porção `-c2` significa testar dois canais simultaneamente (caso hajam dois), enquanto na segunda o comando `aplay` reproduz um áudio `.wav`. Os números de `plughw:1,0` e `hw:1,0` significam o primeiro *output* (representado por "0") do segundo dispositivo (representado por "1"). A diferença entre "plughw" e "hw" é que o primeiro é um *plugin* de conversão que evita ter de explicitar tantos parâmetros (como formato e frequência de amostragem), como deve ser feito no segundo; é através do "plughw" que o ALSA consegue se conectar com o PulseAudio (caso esteja instalado), como se fosse um *plugin*.

```

1 $ speaker-test -c2 -D plughw:1,0 #ruído branco
2 $ aplay --device=plughw:1,0 /seu/diretorio/teste.wav

```

Código 4: Linhas de código para testar alto-falantes com ruído branco e arquivo `.wav` de áudio, respectivamente.

Após checar pacotes e ter à disposição ferramentas úteis apresentadas nessa Subseção, podemos enfim começar nossa jornada para selecionar o dispositivo de áudio que desejamos. Recomenda-se seguir a ordem seguinte (com exceção da Subseção 2.1, que é opcional), pois o leitor ou a leitora não precisará mexer em configurações mais complexas caso seu problema se dissolva com soluções mais simples. As Seções seguintes apresentarão diferentes –e complementares– métodos para reconhecimento e uso do dispositivo desejado, tendo sido ordenadas de soluções mais simples a maneiras mais complexas de se resolver.

## 2 GUI-desktop

A primeira e mais simples tentativa é ir no seu *desktop* e buscar reconhecimento e/ou alteração do dispositivo via seu gerenciador de ambientes. Diversos tipos de GUI para desktop existem, como o Raspbian Lite e o PIXEL, sendo todos bastante parecidos no que diz respeito ao controle de dispositivos de áudio. O positivo do GUI é que você já pode tentar mudar seu dispositivo de áudio ou pelo menos saber se ele está sendo reconhecido a partir da sua interface gráfica de fácil utilização.

Dependendo de outras configurações de certos pacotes ou do próprio RPi (veremos à frente), esta simples solução pode não ter efeito (porque os programas se sobrepõem) mas não se preocupe: nas seções seguintes veremos por que isso acontece e também outras maneiras



de configurar o dispositivo de áudio. Na sequência veremos como o Gnome através da sua unidade para áudio, o Gnome Alsa mixer, deve ser utilizado para o reconhecimento e controle do dispositivo.

## 2.1 Gnome Alsa mixer

O Gnome é um gerenciador de ambientes para desktop utilizado em plataformas Linux. Uma parte dele, o Gnome Alsa mixer, é um gerenciador de dispositivos de áudio (a partir do pacote `alsa-utils`) em forma de GUI. Com ele é possível alterar o dispositivo selecionado de uma maneira muito semelhante ao Windows, assim como aumentar, abaixar ou deixar no mute saídas ou entradas. O Código 5 contém os comandos necessários para a instalação ou atualização do Gnome Alsa mixer.

```
1 $ sudo apt-get update
2 $ sudo apt-get install gnome-alsamixer #instalar
3 $ sudo apt-get upgrade gnome-alsamixer #atualizar
```

Código 5: Instalação ou atualização do Gnome Alsa mixer.

A Figura 3 apresenta a interface gráfica dessa ferramenta. No canto superior direito podemos ver um ícone em forma de alto-falante: clicamos com o botão esquerdo do mouse e as opções para chegar nessa tela aparecem.

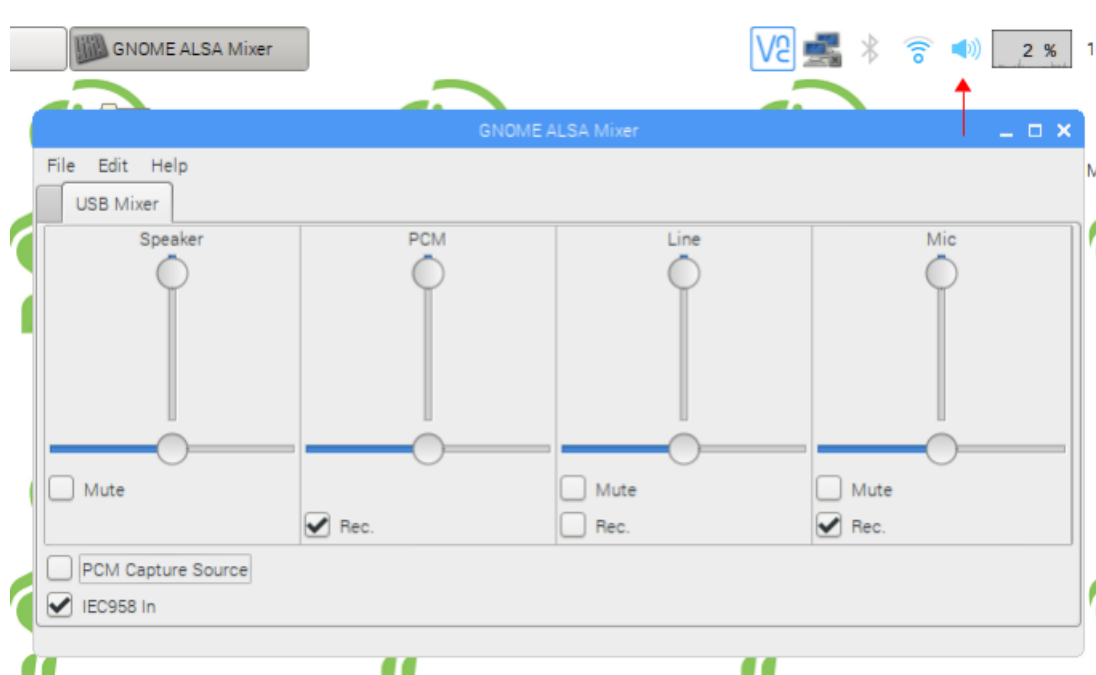


Figura 3: Interface gráfica do Gnome Alsa mixer.

Caso você não tenha o Gnome ou Gnome Alsa mixer instalado, não tem problema, pois além de existir inúmeras opções de interfaces, o ALSA e seus pacotes (atualizados, de preferência) é o que realmente precisamos dentro do Raspbian. O Gnome apenas apresenta uma interface gráfica que pode tornar certas operações mais fáceis de serem realizadas.

Neste método devemos:

- 1) verificar se o dispositivo aparece nas opções para ser selecionado;
- 2) verificar se suas saídas e entradas estão com nomes corretos e
- 3) se *não estão* no modo *mute*.

Após seguir esses passos (o que com certeza você deve ter feito antes), lembramos que testar o áudio em sites como YouTube é uma boa prática. Não resolvendo ou caso o gerenciador não consiga selecionar – e manter – o dispositivo, podemos passar para a próxima tentativa, descrita a seguir.

### 3 PortAudio

O PortAudio<sup>3</sup> é uma biblioteca gratuita para gravação, reprodução e processamento de áudio. É escrito em C e tem capacidade de executar em vários sistemas operacionais e plataformas, possibilitando escrever e compilar programas simples de áudio em 'C' e 'C++' [7]. Alguns dos programas que utilizam essa biblioteca são o Python (para os módulos PyAudio e sounddevice, por exemplo), Audacity e o ITA-Toolbox. A instalação do PortAudio segue a seguinte ordem, documentada e compartilhada pelo colega Leonardo Jacomussi (que auxiliou diversas vezes na construção deste documento):

- 1) Faça o download do arquivo no site: <http://portaudio.com/download.html>;
- 2) Extraia os arquivos do download, abra a pasta e encontre o arquivo 'README.configure.txt', nele terá todas as instruções de como instalar;
- 3) Instale conforme Código 6 e logo após reinicie.

```

1 $ sudo apt-get install autoconf
2 $ sudo apt-get install build-essential
3 $ sudo apt-get install pkg-config
4 $ sudo apt-get install libtool libasound-dev
5 $ cd /home/pi/Downloads/portaudio #exemplo de diretorio
6 $ autoreconf -if
7 $ sudo make install

```

Código 6: Ordem a ser executada na linha de comando para instalação do PortAudio.

### 4 ALSA

O ALSA (*Advanced Linux Sound Architecture*) é "um conjunto de *drivers* de *hardware*, bibliotecas e utilitários que fornecem funcionalidade de áudio e MIDI para o sistema operacional Linux" [8]. Esse subsistema (ALSA) estrutura sobretudo a relação entre o sistema operacional

<sup>3</sup><http://www.portaudio.com/>

e *hardware*, como sugere a Figura 4. Utilizar apenas o ALSA, sem nenhum outro servidor de áudio é comumente chamado de "uso nativo", entretanto, há muitos programas que precisam de bibliotecas como o PortAudio (ou servidores como o PulseAudio para controle).

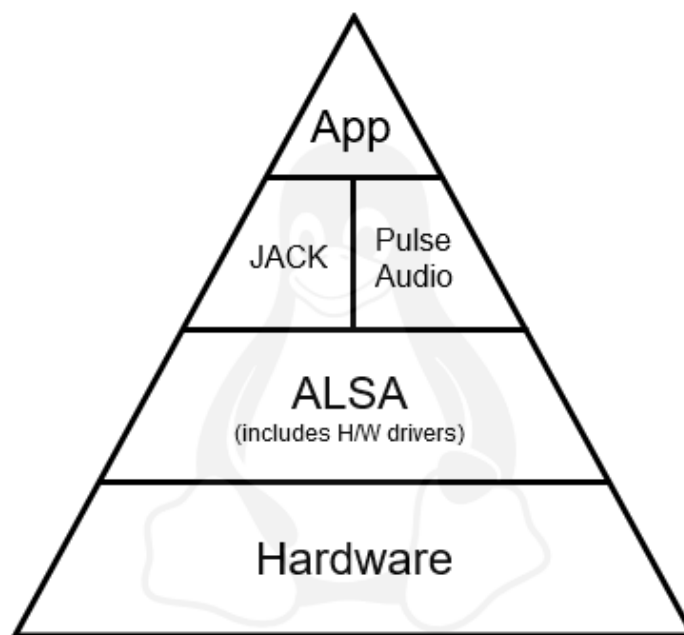


Figura 4: Relação aplicativo-*hardware* hierarquizada. Retirado de: <https://www.learn.digitalaudio.com/how-linux-audio-works-vs-windows-audio-2017>.

A Figura 4 esquematiza a relação entre aplicativo e *hardware*, passando pelo servidor de áudio (JACK ou PulseAudio) e pelo ALSA. Nota-se o caráter complementar das etapas de comunicação da pirâmide e ter essa noção é importante para a investigação de possíveis falhas, além de *sempre* ler atentamente códigos de erros e *changelogs*. A seguir será descrita a checagem de pacotes e configurações puramente ALSA, porém tendo em mente que certas aplicações podem ter dependências de outras bibliotecas (como o PortAudio, por exemplo).

## 4.1 Checando dependências

O `libasound2`, `alsa-oss` e `alsa-utils` são bibliotecas e pacotes de baixo nível que já vem com o Raspbian [9], apoiando a estrutura de controle de áudio (ALSA). Com ele podemos acessar várias funções com os comandos `amixer`, `alsamixer`, `alsactl`, `aplay`, `arecord`, `speaker-test` etc, que serão explicados à frente. O Código 5 ensina como instalar<sup>4</sup> ou atualizar o ALSA. Neste caso, como não há necessidade de instalação do pacote, apresenta o comando para sua atualização. A versão utilizada para criação deste Tutorial foi 1.1.3-1.

```

1 $ sudo apt-get install libasound2 alsa-utils #instalar
2 $ sudo apt-get upgrade libasound2 alsa-utils #atualizar

```

Código 7: Comando para atualização do pacote `alsa-utils`.

<sup>4</sup>Para instalação completa, visite: [https://alsa.opensrc.org/Quick\\_Install](https://alsa.opensrc.org/Quick_Install).

## 4.2 Configurando o ALSA: alsamixer

Este é um modo de gerenciar dispositivos de áudio a partir do terminal, utilizando um *mixer* do ALSA, que faz parte do núcleo do Linux. Para iniciar, abra o terminal utilizando o atalho `Ctrl + Alt + T` e digite o conteúdo do Código 8. Para cada linha, a primeira palavra refere-se a um comando do ALSA e o argumento `-l` apresenta a lista simples. Pode-se encontrar outros argumentos interessantes acessando o *help* de cada um desses comandos.

```
1 $ aplay -l
2 $ arecord -l
```

Código 8: Comandos que listam dispositivos para reprodução e captura de áudio, respectivamente.

O resultado do *bash* deve parecer como na Figura 5. A linha `aplay -l` revela uma lista de dispositivos para reprodução de áudio e `arecord -l` uma lista dos que capturam áudio. Podemos notar que os dispositivos são identificados como "card 0" ou "card 1". Muitas vezes (e não todas) iremos ver as expressões *card* e *sound card* referindo-se ao que, neste Tutorial, chamamos de dispositivos de áudio<sup>5</sup>. Neste caso, o primeiro deles é o HAT AMP2 da HiFiBerry<sup>6</sup>, e o segundo é dispositivo de áudio USB da C-Media CM6206<sup>7</sup>.

```
pi@raspberrypi:~ $ aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: sndrpihifiberry [snd_rpi_hifiberry_dacplus], device 0: HiFiBerry pcm512x-hifi-0 []
  Subdevices: 1/1
  Subdevice #0: subdevice #0
card 1: Device [USB Sound Device], device 0: USB Audio [USB Audio]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
pi@raspberrypi:~ $ arecord -l
**** List of CAPTURE Hardware Devices ****
card 1: Device [USB Sound Device], device 0: USB Audio [USB Audio]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
pi@raspberrypi:~ $
```

Figura 5: Lista de dispositivos de reprodução e de captura de áudio, respectivamente.

Ainda na Figura 5 podemos notar que o dispositivo identificado como "card 0" possui

<sup>5</sup>Faz-se com a intenção de elucidar que os dispositivos podem se conectar ao RPi de diferentes formas (e.g. HATs, via USB ou via *bluetooth* para caixas de som).

<sup>6</sup>Mais informações:

<https://www.hifiberry.com/blog/introducing-the-amp2-more-power-higher-sample-rates-more-affordable/>

<sup>7</sup>Informações e *datasheet*: [https://www.cmedia.com.tw/products/USB20\\_FULL\\_SPEED/CM6206](https://www.cmedia.com.tw/products/USB20_FULL_SPEED/CM6206)

apenas controle para a saída de áudio (*playback*), pois não aparece na lista dos que capturam áudio. Sendo assim, focaremos no reconhecimento da placa de áudio USB da C-Media, ou "card 1", pois esta possui capacidade tanto de *input* quanto de *output*. A organização do ALSA fica mais compreensível ao se conceituar certos nomes recorrentes, como:

- **card**: é um índice que define a placa de som utilizada, seja ela física (como tipos *onboard*, via USB etc) ou virtual como dispositivos MIDI virtuais [10].
- **device**: dispositivos do *card*. Podem ser, por exemplo, dispositivos de áudio digital (PCM), de controle (*mixers*) ou MIDI [10].
- **subdevice**: está associado com as portas do *device*, como por exemplo capturar áudio a partir de canais simultâneos [11].

Se seu dispositivo *não apareceu na lista*, não se desespere. Utilizando o Raspberry Pi aprendemos "pequenos truques", como ligar a tela do pc antes de ligar seu RPi para evitar erros. Verifique também o cabo do seu dispositivo: mesmo que ele esteja funcionando no Windows, por exemplo, pode ser que algum pequeno erro decorrendo do mau estado do cabo não consiga ser corrigido pelo Raspbian.

Após inúmeras tentativas sem sucesso, e ninguém em nenhum fórum comentando sobre seu dispositivo funcionando com RPi, você já deve saber qual é a má notícia. Para saber se ele está sendo reconhecido de outro jeito, teste o comando `lsusb` (caso seja USB) e veja se aparece seu nome. Caso tenha aparecido como na Figura 5, seguiremos para o próximo passo.

```
1 $ alsamixer #controle e modifique
2 $ sudo alsactl store #salve
```

Código 9: Comandos para abrir o *mixer* do ALSA e salvar suas configurações, respectivamente.

O Código 9 mostra a sequência de comandos para controlar e salvar configurações relativas a dispositivos de áudio. A primeira linha (`alsamixer`) dá origem à tela de controle (o *mixer* do ALSA). A Figura 6 ilustra esse resultado.

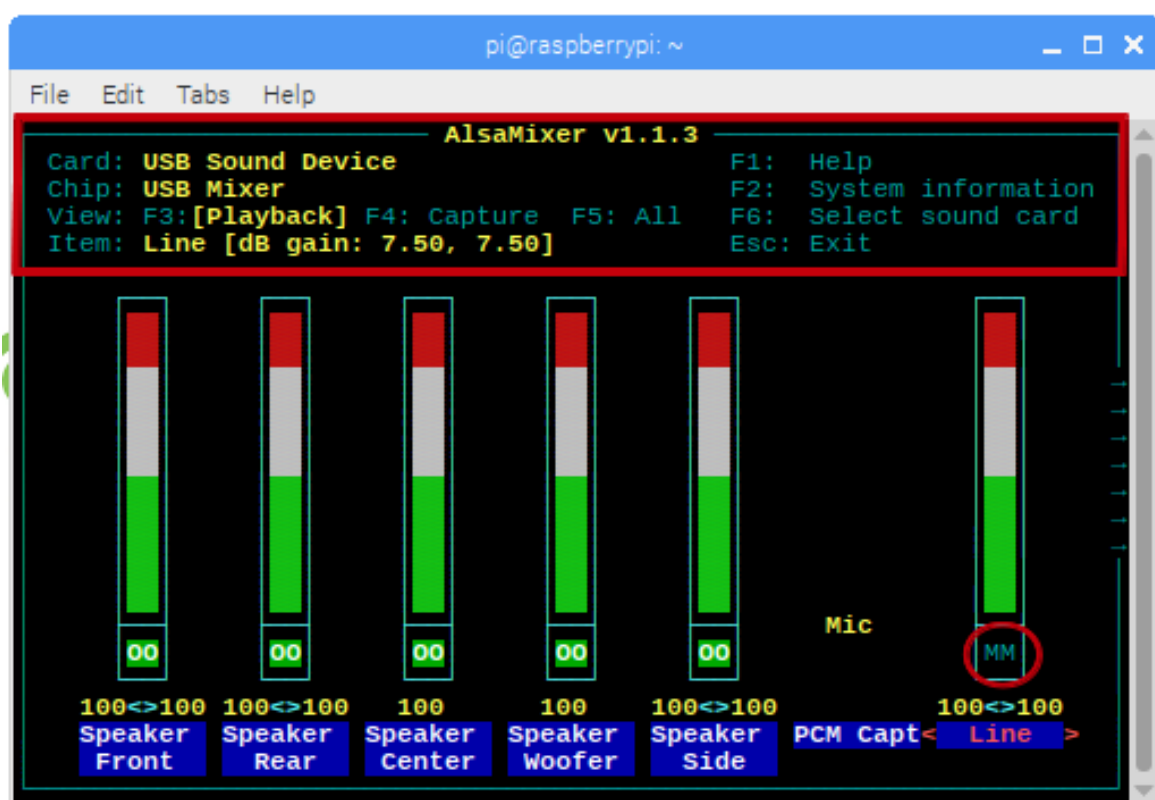


Figura 6: Controles do alsamixer.

O destaque em vermelho na parte de cima da Figura 6 identifica o nome do dispositivo e controles que podem ser utilizados para a navegação nesse *mixer*. Circulado em vermelho e na parte inferior, ainda na Figura 6, vemos as letras "MM", simbolizando que aquele canal está mudo. Para ativar ou desativar o *mute* devemos selecionar o canal (seja ele saída ou entrada) e apertar a tecla M. As setas do teclado  $\uparrow \downarrow \leftarrow \rightarrow$  são utilizadas para mexer nas opções e para selecionarmos o dispositivo, basta apertamos a tecla F6, como ilustra a Figura 7.

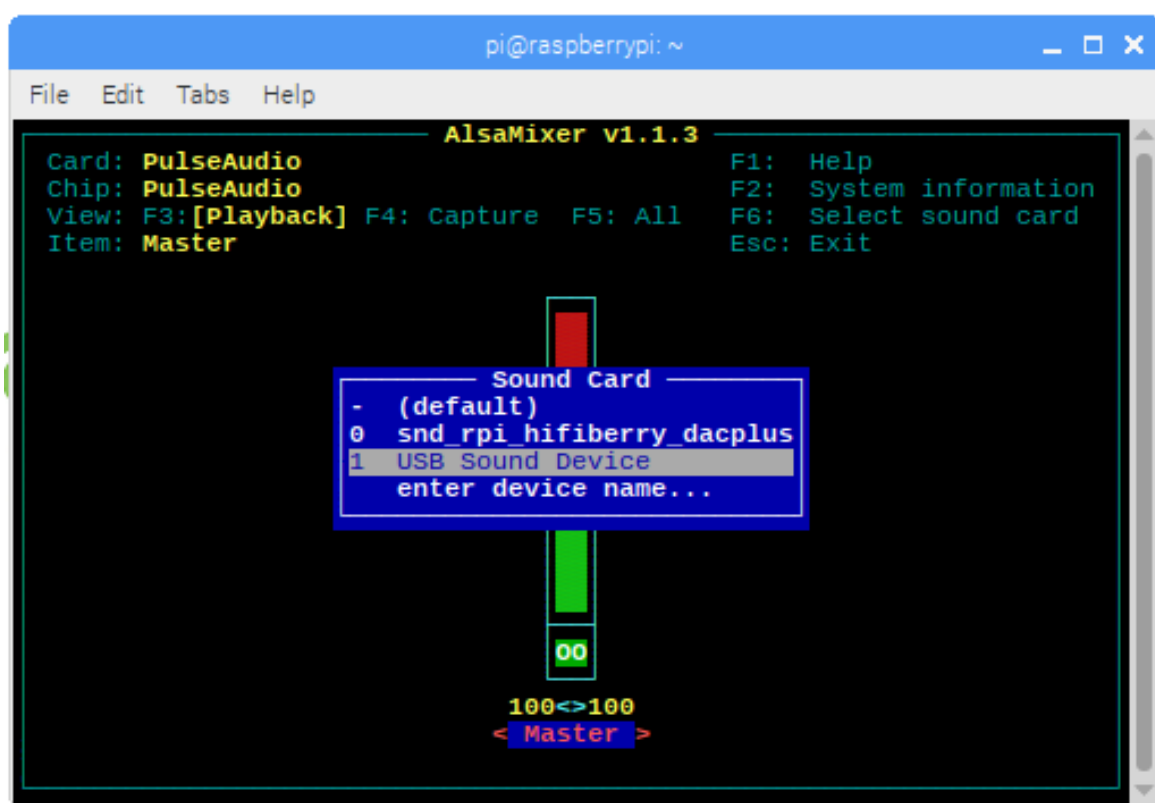


Figura 7: Seleção do dispositivo no alsamixer.

Podemos ver que, como já dito, estão sendo reconhecidos dois dispositivos: o HAT da a HiFiBerry e o dispositivo USB C-Media, acompanhados dos seus respectivos índices de identificação de *sound cards*. Após realizar as modificações necessárias, digitaremos o conteúdo da segunda linha do Código 9: `sudo alsactl store`. Esta linha de código é importante pois *garante* que as configurações que você acabou de editar sejam salvas. O comando `sudo` neste caso é estritamente necessário, para que sejam salvas as alterações para todos os usuários e suas permissões.

Neste método, utilizando o `alsamixer`, devemos:

- 1) verificar dispositivos disponíveis de captura e reprodução de áudio, conforme o Código 8;
- 2) abrir *mixer* do ALSA com o comando `alsamixer` e configurar o necessário, como:
  - selecionar o dispositivo de áudio desejado, utilizando a tecla F6;
  - aumentar (zerar) o ganho dos canais de entrada ou saída;
  - tirar canais do mudo;
  - selecionar qual entrada recebe o mic (line ou mic, por exemplo).
- 3) salvar todas as configurações com o comando `sudo alsactl store` do Código 9;
- 4) testar com o Código 3.



### 4.3 Configurando o ALSA: asound.config

Outra forma de configurar o ALSA é de maneira "roots". Acessamos via *bash* a pasta que contém o arquivo `asound.config` e o alteramos. O Código 10 enumera a sequência a ser seguida para encontrar e abrir o arquivo.

```
1 $ cd /etc\  
2 $ dir # mostra conteudo da pasta, que deve conter asound.config  
3 $ sudo nano asound.config #para abrir arquivo
```

Código 10: Linhas de comando para se chegar ao arquivo `asound.config`.

O arquivo em questão é crucial para se modificar o dispositivo de áudio padrão. Se, por alguma razão ele não esteja na pasta esperada, pode-se usar qualquer um dos comandos do Código 11. Caso ele tenha sido excluído, recomenda-se recriar um arquivo de mesmo nome na pasta original.

```
1 $ #Caso nao esteja na pasta /etc\  
2 $ whereis asound.config #uma forma de procurar  
3 $ sudo \ find -name asound.config #outra forma de procurar
```

Código 11: Comandos para procurar o arquivo `asound.config`.

O resultado da linha `sudo nano asound.config` do Código 10 pode ser visto abaixo (Figura 8). A sigla "pcm" significa *Pulse Code Modulation*, sendo a forma padrão para se representar digitalmente sinais de áudio em dispositivos, tratando-se de um método utilizado para amostrar e quantizar um sinal de áudio antes analógico. O PCM associado ao tipo "hw" dá ao ALSA a habilidade de controlar o *hardware* "card 1" criando um *sound card* virtual [12], dando acesso ao *device* e aos *subdevices* definido pela lógica do ALSA.

Já "ctl" está associado ao controle do *hardware* (como um *mixer*). A palavra *default* representa que essas são as configurações padrão, podendo-se substituir essa palavra por outras para criar novos perfis PCM e CTL personalizados para o ALSA, como será mostrado nas Seções seguintes.

```

pi@raspberrypi: /etc
GNU nano 2.7.4 File: asound.config

pcm.!default{
    type hw
    card 1
}

ctl.!default{
    type hw
    card 1
}

^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify
^X Exit      ^R Read File ^\ Replace   ^U Uncut Text ^T To Spell

```

Figura 8: Conteúdo do arquivo `asound.config` após alteração.

Como uma primeira tentativa podemos mudar ambos os números dos *cards* para aqueles que representa o dispositivo que desejamos fazer funcionar, aparentando como na Figura 8. Para testar se seu dispositivo está marcado como o *default*, pode-se utilizar as linhas de Código 12 [12]:

```

1 $ arecord -D default /diretorio/teste.wav #capturando audio
2 $ aplay -D default /diretorio/teste1.wav #reproduzindo
3 $ # ou
4 $ arecord -D plughw:1 /diretorio/teste.wav
5 $ aplay -D plughw:1 /diretorio/teste1.wav

```

Código 12: Maneiras de testar a captura e reprodução pelo seu dispositivo de áudio.

O índice "1" logo após "plughw" deve ser o índice correspondente ao *card* que simboliza o dispositivo que você deseja testar, caso você tenha desejo de comparar com o *default*. Dependendo de que tipo de aplicação você deseja testar, possivelmente este método pode resolver.

Neste método de configuração do ALSA via `asound.config`, devemos:

- 1) encontrar e abrir o arquivo `asound.config` (Código 10);
- 2) alterar os índices para aqueles que correspondam ao *card* que desejamos ativar;
- 3) testar via *bash* (Código 12) e outros meios.

Uma outra maneira de alterar qual dispositivo será padrão, é digitando no *bash* o endereço `cd /usr/share/alsa` e abrir o arquivo desejado com o comando `sudo nano alsa.conf` e alterar o índice das linhas "defaults.ctl.card 0" e "defaults.pcm.card 0". Há ainda inúmeras maneiras de configurar o ALSA que você pode facilmente encontrar na internet e algumas delas ainda serão apresentadas neste documento.

Como já dito anteriormente, o ALSA é parte do núcleo do Linux, e estabelece um ligação que, às vezes, precisa de uma "mãozinha" para funcionar da forma desejada em certas aplicações (como no Chromium e no Python, por exemplo). Veremos nas Subseções seguintes, outras maneiras de configuração de áudio que começam a se misturar com o ALSA, de forma a permitir uma personalização mais robusta e ampla.

## 5 PulseAudio

O PulseAudio<sup>8</sup> é um servidor de som multi-plataformas que costuma vir junto com gerenciadores de ambiente de desktop (como o Gnome, por exemplo) [13]. É considerado um *middleware*, pois tem a capacidade de agir mediando *hardware* e suas aplicações [1]. Pode rodar em diversos modelos de sistemas operacionais (alguns mais antigos), como MacOS X e Windows XP. A Figura 9 é bastante interessante, e demonstra o fluxo de processos e estágios de *hardware* e *software* somados à interação humano-máquina. O PulseAudio nesse infográfico se localizaria na região do *software* (em laranja), onde está escrito "*middleware*".

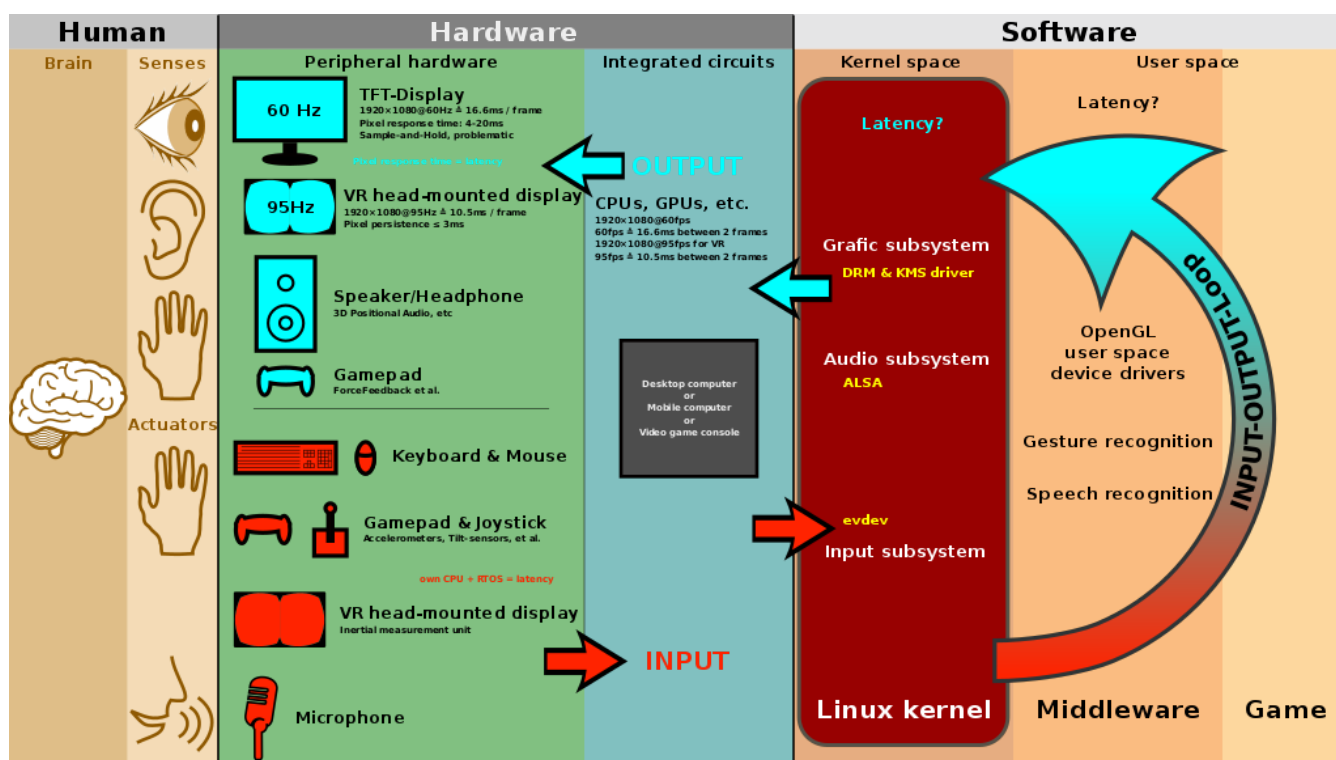


Figura 9: Infográfico de processos de interação humano-máquina através de jogos em plataformas Linux. (Por Shmuel Csaba Otto Traian; <https://commons.wikimedia.org/w/index.php?curid=31418026>)

O Código 13 apresenta, respectivamente, como checar se o PulseAudio está instalado e qual sua versão, seguido dos comandos para atualizar e instalar caso necessário. Atualmente

<sup>8</sup>Informações oficiais no site: <https://www.freedesktop.org/wiki/Software/PulseAudio/>.

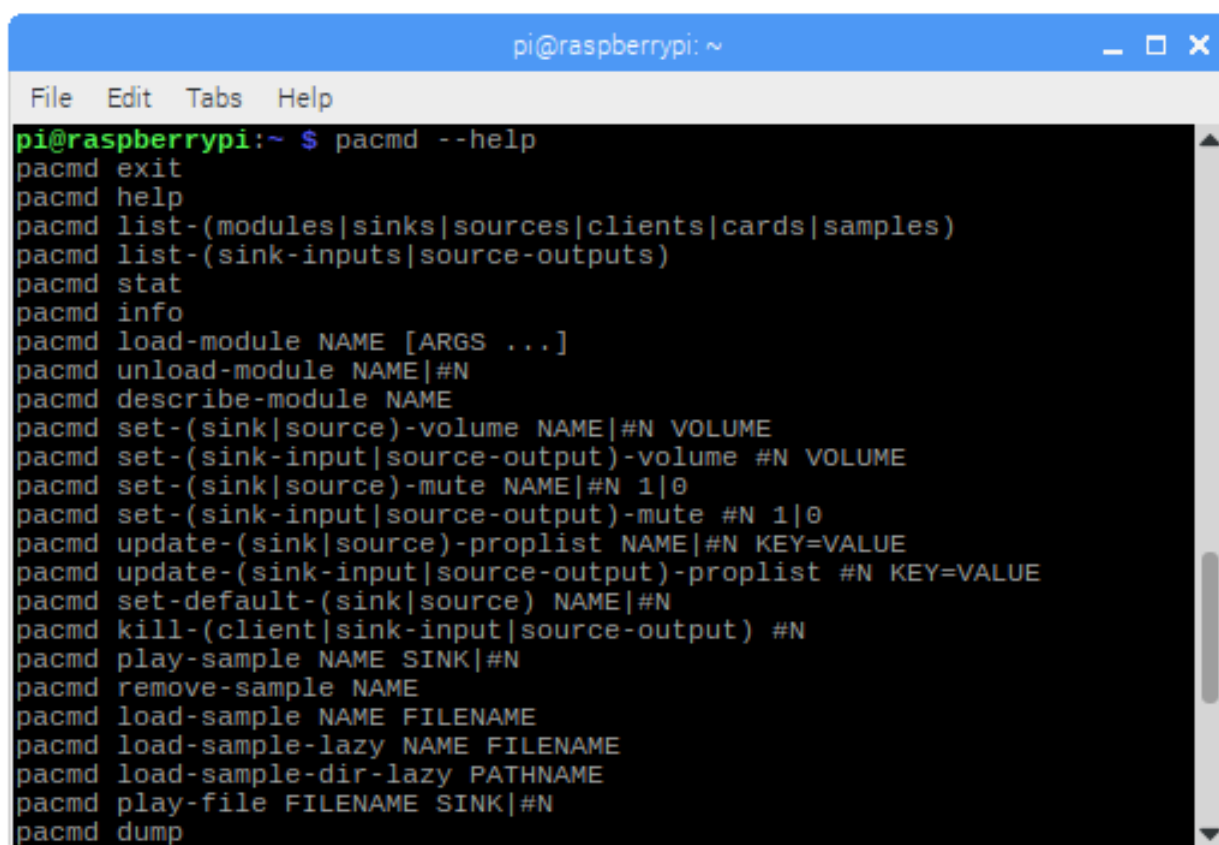
a última versão estável do PulseAudio é a 13.0, lançada em 13 setembro de 2019, entretanto, a versão 9 do Raspbian Stretch necessita de construção manual para versões posteriores a 10.0.

```
1 $ pulseaudio --version
2 pulseaudio 10.0
3 $ sudo apt-get upgrade pulseaudio
4 $ sudo apt-get install pulseaudio
```

Código 13: Comandos para checar, instalar ou atualizar o PulseAudio.

Há dois principais comandos do PulseAudio que nos permitem configurá-lo com o propósito de alterar dispositivos de áudio, perfil dos *sound cards* e seus canais de entrada e saída. São eles:

- i. `pacmd`: reconfigura o servidor de áudio durante seu tempo de execução, muitas vezes sendo necessária a reinicialização do servidor.
- ii. `pactl`: controla o servidor de áudio ativo (sessão em execução). É uma parte do `pacmd` que altera configurações de áudio, sem a necessidade de reiniciar o PulseAudio.



```
pi@raspberrypi: ~
File Edit Tabs Help
pi@raspberrypi:~ $ pacmd --help
pacmd exit
pacmd help
pacmd list-(modules|sinks|sources|clients|cards|samples)
pacmd list-(sink-inputs|source-outputs)
pacmd stat
pacmd info
pacmd load-module NAME [ARGS ...]
pacmd unload-module NAME|#N
pacmd describe-module NAME
pacmd set-(sink|source)-volume NAME|#N VOLUME
pacmd set-(sink-input|source-output)-volume #N VOLUME
pacmd set-(sink|source)-mute NAME|#N 1|0
pacmd set-(sink-input|source-output)-mute #N 1|0
pacmd update-(sink|source)-proplist NAME|#N KEY=VALUE
pacmd update-(sink-input|source-output)-proplist #N KEY=VALUE
pacmd set-default-(sink|source) NAME|#N
pacmd kill-(client|sink-input|source-output) #N
pacmd play-sample NAME SINK|#N
pacmd remove-sample NAME
pacmd load-sample NAME FILENAME
pacmd load-sample-lazy NAME FILENAME
pacmd load-sample-dir-lazy PATHNAME
pacmd play-file FILENAME SINK|#N
pacmd dump
```

Figura 10: Resposta do *bash* ao comando `pacmd -help`.

Para conhecer um pouco mais sobre a infinidade de opções de ambos os comandos citados acima, são utilizados os códigos: `pacmd -help`, `pactl -help`. A Figura 10 apresenta a tela de *help* do comando `pacmd` do PulseAudio. Para ler os manuais, os seguintes códigos podem ser utilizados: `man pulseaudio`, `man pacmd` e `man pactl`.

## 5.1 Configurando o PulseAudio

O PulseAudio é um servidor de áudio que auxilia o ALSA a lidar com vários programas em várias instâncias, sendo por isso classificado como *middleware*. Possui a habilidade de lidar com diferentes dispositivos e seus canais para rodar seus aplicativos, enquanto o ALSA dialoga mais diretamente com o *hardware*. Veremos formas de configurar como modificar apenas pelo PulseAudio ou ainda o expor ao ALSA.

```
1 $ pactl info
2 $ pacmd list-cards
```

Código 14: dd

A Figura 11 exemplifica o resultado da linha do Código 14 (`pactl info`), printando na tela que configurações estão sendo seguidas e que dispositivo é o escolhido padrão para o PulseAudio. Neste caso temos o mesmo nome do *card* destacado na Figura 12 e algumas outras informações, como o formato *s16le* (sinal amostrado com 16 *bits* e padrão *little-endian*<sup>9</sup>), dois canais de saída (*ch2*) e frequência de amostragem de 44100 Hz.

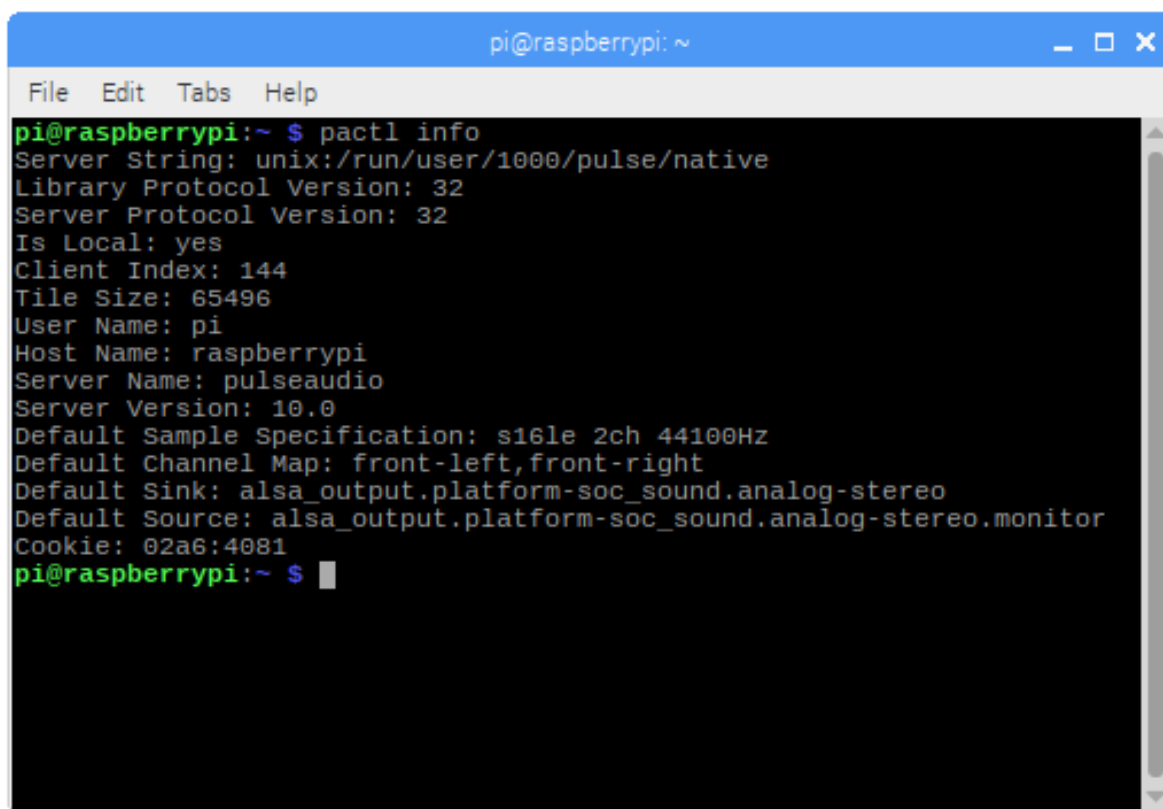


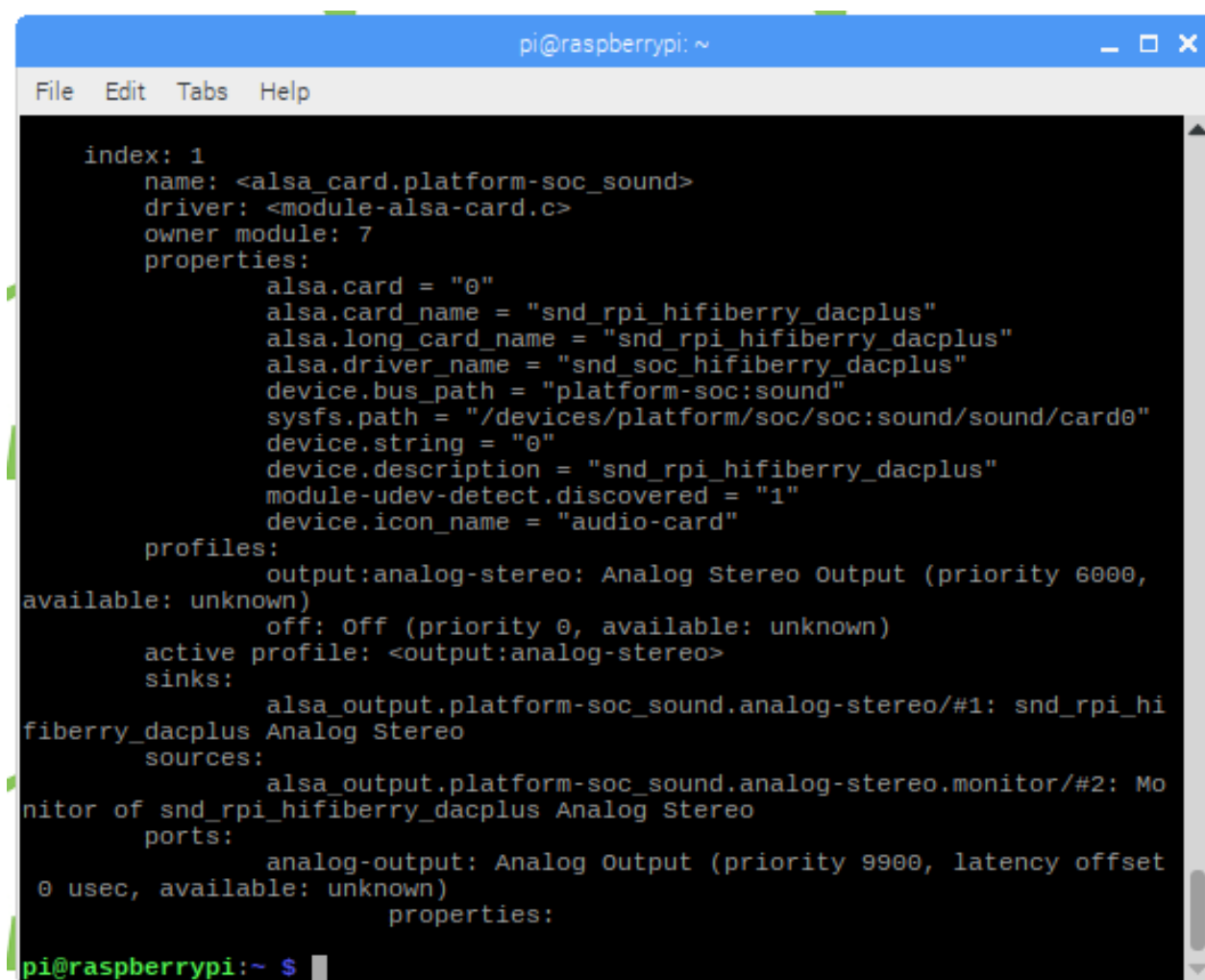
Figura 11: Comando que printa informações sobre parâmetros padrão para a sessão atual do PulseAudio.

A segunda linha do Código 14 apresenta em forma de lista um dos dispositivos de áudio (*cards*) em detalhes, onde índices, nome, perfis, saídas e entradas são as principais informações. Os termos *sink* e *source* são para *output* e *input*, respectivamente, e serão bastante utilizados. O comando `pacmd list-cards` imprime na tela extensas informações, como já mencionado

<sup>9</sup>*Little-endian* é uma maneira ordenada de armazenar *bytes* na memória do computador. Neste caso (*little*) começando a armazenar pelo *byte* menos significativo.

anteriormente, fazendo com que a Figura 12 mostre apenas parte do que é exibido. De acordo com o número do índice (1) podemos concluir que o PulseAudio lê esse dispositivo como o segundo *card*, diferentemente do ALSA. Em "properties" podemos ver listados número do *card* para o ALSA e o nome do dispositivo, que é outra diferença entre os dois. Como o HiFiBerry não possui entrada, suas características como *profile*, *sink* e *sources* são preenchidas com informações apenas sobre a saída: analógica e estéreo.

Ainda em *sources*, o "monitor" no final do nome representa um *device* (assim como uma *hardware device* e um *virtual device* [14]; apesar de estar em *sources*, na verdade o monitor está associado às *sinks*, e atua copiando, duplicando e salvando todo sinal que vai para os alto-falantes [14]. O *card* de índice 0 também foi mostrado na tela e apesar de ser o dispositivo que desejamos configurar (placa de áudio USB), utilizamos outro (HiFiBerry) apenas com intuito de ilustrar a estrutura geral das informações obtidas a partir do comando.



```

pi@raspberrypi: ~
File Edit Tabs Help

index: 1
name: <alsa_card.platform-soc_sound>
driver: <module-alsa-card.c>
owner module: 7
properties:
    alsa.card = "0"
    alsa.card_name = "snd_rpi_hifiberry_dacplus"
    alsa.long_card_name = "snd_rpi_hifiberry_dacplus"
    alsa.driver_name = "snd_soc_hifiberry_dacplus"
    device.bus_path = "platform-soc:sound"
    sysfs.path = "/devices/platform/soc/soc:sound/sound/card0"
    device.string = "0"
    device.description = "snd_rpi_hifiberry_dacplus"
    module-udev-detect.discovered = "1"
    device.icon_name = "audio-card"

profiles:
    output:analog-stereo: Analog Stereo Output (priority 6000,
available: unknown)
    off: Off (priority 0, available: unknown)
    active profile: <output:analog-stereo>
sinks:
    alsa_output.platform-soc_sound.analog-stereo/#1: snd_rpi_hi
fiberry_dacplus Analog Stereo
sources:
    alsa_output.platform-soc_sound.analog-stereo.monitor/#2: Mo
nitor of snd_rpi_hifiberry_dacplus Analog Stereo
ports:
    analog-output: Analog Output (priority 9900, latency offset
0 usec, available: unknown)
    properties:

```

Figura 12: Parte do resultado do comando `pacmd list-cards`, apresentando informações sobre o segundo *card*.

Agora que conhecemos um pouco a forma que nosso PulseAudio está configurado, podemos modificar, comparar e testar. As próximas linhas mostram como utilizar os comandos para alterar em diferentes níveis as entradas e saídas padrão do PulseAudio. A sintaxe de dois comandos importantes é descrita no Código 15.

O comando `pacmd` permite a configuração do servidor, enquanto o `pactl` controla e modifica a sessão atual do PulseAudio. Na prática isso quer dizer que, para que o `pacmd` faça

efeito, você deve reiniciar o servidor, o que leva à finalização de uma sessão e início de outra (os comandos `pulseaudio --kill` e `pulseaudio --start` fazem isso).

```
1 $ # Sintaxe
2 $ pacmd set-default-(sink|source) NAME|#N
3 $ pactl [options] set-default-(sink|source) NAME
```

Código 15: Sintaxe para alteração de *inputs* e *outputs* para `pacmd` e `pactl`.

Alguns exemplos de uso dos dois comandos podem ser vistos no Código 16. O `pacmd` possui dois modos de declaração, sendo estes utilizando o número ou o nome e `pactl` apenas o nome. Utilizando o `pactl` as modificações serão feitas na hora, porém ao reiniciar o servidor, podem ser perdidas. Recomenda-se portanto, a configuração de ambos os comandos para evitar perdas.

```
1 $ # Exemplos
2 $ pacmd set-default-sink 0
3 $ pacmd set-default-source alsa_input.usb-0d8c_USB_Sound_Device-00.analog-
   stereo
4 $ pactl set-default-sink alsa_output.usb-0d8c_USB_Sound_Device-00.analog-
   stereo
5 $ pactl set-default-source alsa_input.usb-0d8c_USB_Sound_Device-00.analog-
   stereo
```

Código 16: Exemplos de como modificar *sinks* e *sources* com `pacmd` e `pactl`.

Os nomes de *sinks* e *sources* utilizados acima no exemplo podem ser encontrados com o comando `pacmd list-cards`, já mencionado anteriormente. A Figura 13 exemplifica esses nomes e destaca também outra característica: "ports", que representa uma de cada vez as portas físicas de saída ou entrada do dispositivo [14]. Há ainda "profile", que reúne combinações de entrada e saída que podem ser bastante específicas, como habilitar um *surround* 7.1 ou deixar a entrada digital e a saída analógica, por exemplo.



```

pi@raspberrypi: ~
File Edit Tabs Help
) Output + Analog Stereo Input (priority 5560, available: unknown)
  output:iec958-stereo+input:iec958-stereo: Digital Stereo Duplex
(IEC958) (priority 5555, available: unknown)
  off: Off (priority 0, available: unknown)
  active profile: <output:analog-stereo+input:analog-stereo>
  sinks:
    alsa_output.usb-0d8c_USB_Sound_Device-00.analog-stereo/#0: CM106
    Like Sound Device Analog Stereo
  sources:
    0: Monitor of CM106 Like Sound Device Analog Stereo
    alsa_input.usb-0d8c_USB_Sound_Device-00.analog-stereo/#1: CM106
    Like Sound Device Analog Stereo
  ports:
    analog-input-mic: Microphone (priority 8700, latency offset 0 us
ec, available: unknown)
    properties:
      device.icon_name = "audio-input-microphone"
    analog-input-linein: Line In (priority 8100, latency offset 0 us
ec, available: unknown)
    properties:
      iec958-stereo-input: Digital Input (S/PDIF) (priority 0, latency
offset 0 usec, available: unknown)

```

Figura 13: Características do *card* destacadas em vermelho após o uso do comando `pacmd list-cards`.

Para configurarmos com `pacmd` ou `pactl` devemos seguir os seguintes passos:

- 1) verificar com o comando `pactl info` entradas saídas e outros parâmetros definidos como padrão pelo PulseAudio;
- 2) conhecer o nome, índice e outras características dos *cards* existentes com `pacmd list-cards`;
- 3) definir *sinks* e *sources* tanto com o `pacmd` quanto para `pactl` usando os Códigos 15 ou 16;
- 4) reiniciar o servidor com `pulseaudio --kill` e `pulseaudio --start` (opcional);
- 5) testar em diferentes aplicativos, como no Youtube, por exemplo;

## 5.2 Expondo o PulseAudio para o ALSA

Enquanto o ALSA, no primeiro momento, age escolhendo diretamente qual dispositivo de áudio (chamado pelo programa de *card*) será utilizado e controlado, o PulseAudio escolhe as saídas e entradas, que podem ser inclusive de diferentes dispositivos. Para que isso possa acontecer, o servidor deve se sobrepôr ao ALSA, podendo controlá-lo. O Código 17 mostra em sua primeira e segunda linha como acessar o ponto de interação entre PulseAudio e ALSA.

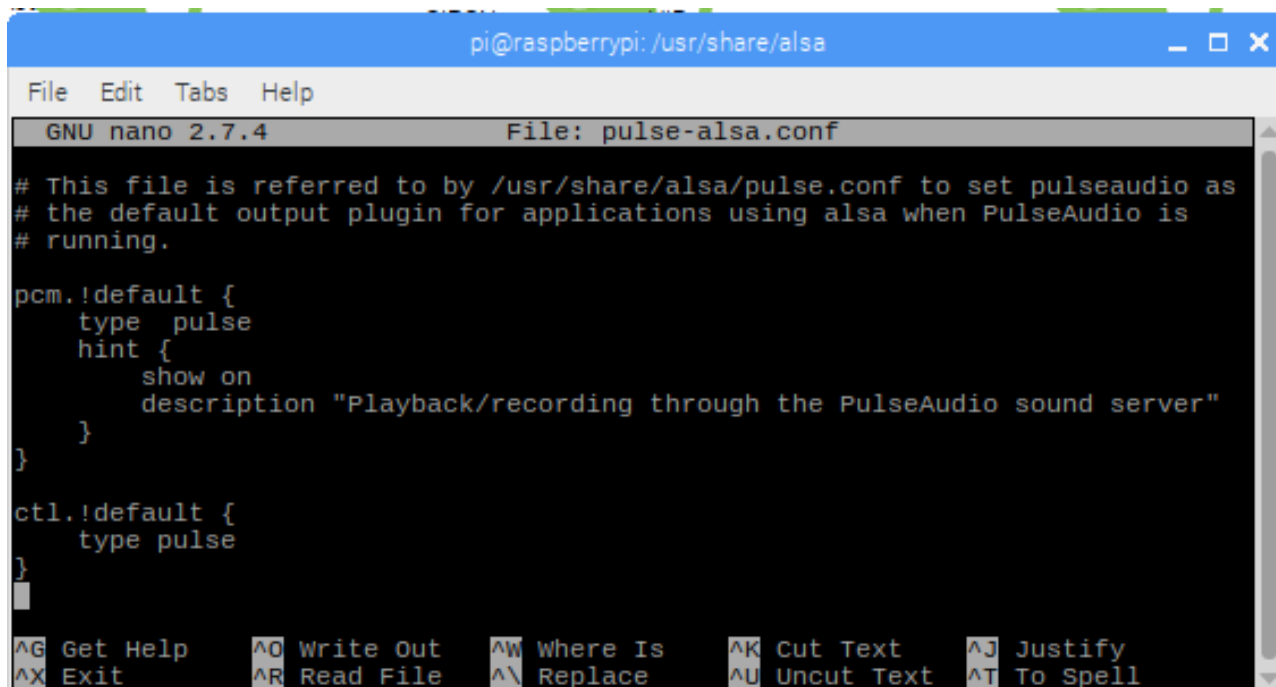
```

1 $ cd /usr/share/alsa/
2 $ sudo nano pulse-alsa.conf
3 $ cd /usr/share/alsa/alsa.conf.d/
4 $ sudo nano pulse.conf

```

Código 17: Códigos para checar a relação "pulse-alsa".

Abrindo o arquivo `pulse-alsa.conf` (Figura 14) vemos uma estrutura muito semelhante à estrutura da Figura 8, mas que nesse caso, define o PulseAudio como padrão, mesmo quando os aplicações utilizam ALSA. Para ter esse arquivo, você deve ter o pacote `pulseaudio-alsa` que faz esse trabalho, e caso não tenha, veremos mais à frente como personalizar a interação "pulse-alsa" a partir do `asound.config`.



```
pi@raspberrypi: /usr/share/alsa
File Edit Tabs Help
GNU nano 2.7.4 File: pulse-alsa.conf
# This file is referred to by /usr/share/alsa/pulse.conf to set pulseaudio as
# the default output plugin for applications using alsa when PulseAudio is
# running.
pcm.!default {
    type pulse
    hint {
        show on
        description "Playback/recording through the PulseAudio sound server"
    }
}
ctl.!default {
    type pulse
}
^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify
^X Exit      ^R Read File  ^_ Replace   ^U Uncut Text ^T To Spell
```

Figura 14: Configuração do arquivo "pulse-alsa.conf": definindo o PulseAudio como padrão para os parâmetros pcm e ctl.

O arquivo `pulse.conf` tem a capacidade de fazer com que o PulseAudio sirva como *plugin* para o ALSA ("plughw" vem disso). Na Figura 15 vemos que esse arquivo chama outro de nome `pulse-alsa.conf` (já mencionado acima), e então define os dispositivos e portas padrão do PulseAudio como sendo também o padrão do ALSA. Para manter a autonomia do ALSA, esses dois arquivos devem ser modificados, porém não recomenda-se desacoplar o servidor da arquitetura, pois se por um lado o PulseAudio tem maior capacidade de gerir os dispositivos, por outro o ALSA trabalha diretamente no núcleo do Linux, e juntos fornecem uma ferramenta mais poderosa de configuração de áudio.

```

pi@raspberrypi: /usr/share/alsa/alsa.conf.d
File Edit Tabs Help
GNU nano 2.7.4 File: pulse.conf
# PulseAudio alsa plugin configuration file to set the pulseaudio plugin as
# default output for applications using alsa when pulseaudio is running.
hook_func.pulse_load_if_running {
    lib "libasound_module_conf_pulse.so"
    func "conf_pulse_hook_load_if_running"
}

@hooks [
{
    func pulse_load_if_running
    files [
        "/usr/share/alsa/pulse-alsa.conf"
    ]
    errors false
}
]

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell

```

Figura 15: Configuração da definição do PulseAudio como *plugin* para sobrepor o ALSA.

Há vários *front-ends* para PulseAudio, sendo alguns deles *pacmixer*, *pamixer*, *pavucontrol*, *paprefs* etc. Na Referência [14] é possível encontrar informação sobre alguns deles, e *prints* detalhados com explicações (se tratando do PulseAudio, essa é uma das melhores referências encontradas). A Referência [1] também apresenta informações muito boas sobre o PulseAudio, inclusive a maneira de configuração a seguir. Vimos acima que é possível tornar as escolhas de saídas e entradas do PulseAudio padrão também para o ALSA a partir dos arquivos *pulse.conf* e *pulse-alsa.conf*. Segundo o *site* da ArchWiki, embora possamos definir o padrão ALSA como sendo o padrão PulseAudio (o que já fizemos), temos uma maneira mais versátil de configurar essa interação [1].

```

1 $ cd /etc/
2 $ sudo nano asound.config #para abrir arquivo

```

Código 18: Comandos para acessar o arquivo *asound.config*.

Para personalizar a interação "pulse-alsa", podemos criar estilos e definições editando *asound.config*: o Código 18 lembra como chegar ao arquivo, enquanto o Código 19 exemplifica essas personalizações. Podemos criar configurações exclusivas para os parâmetros *pcm* e *ctl*, utilizando "type pulse". O efeito disso é utilizar comandos do PulseAudio, como *device*, *sink* e *source* dentro das configurações do ALSA. Essa personalização deve ser usada caso você não tenha os arquivos mostrados nas Figuras 15 e 14 (que você pode conseguir instalando o pacote *pulseaudio-alsa*).

```

1
2 # Cria um input ou output ALSA usando sources ou sinks específicos do PulseAudio
3
4 pcm.pulse-exemplo1 {
5     type pulse

```

```

6     device alsa_card.usb-0d8c_USB_Sound_Device-00
7     fallback "pulse-exemplo2" #caso nao esteja disponivel
8 }
9
10 pcm.pulse-exemplo2 {
11     type pulse
12     device alsa_card.platform-soc_sound
13
14 }
15
16 # Controlando um source ou uma sink separadamente com um mixer:
17 ctl.pulse-example {
18     type pulse
19     sink "HAT"
20     source "placa-de-audio-USB"
21
22     #exemplo: saida do HAT e entrada da placa
23     sink alsa_output.platform-soc_sound.analog-stereo
24     source alsa_input.usb-0d8c_USB_Sound_Device-00.analog-stereo
25 }
26
27 # Para se sobrepor ao mixer padrao
28 ctl.!default {
29     type pulse
30     sink alsa_output.platform-soc_sound.analog-stereo
31     source alsa_input.usb-0d8c_USB_Sound_Device-00.analog-stereo
32 }

```

Código 19: Perfis de exposição do PulseAudio ao ALSA, usando o *plugin* "pulse" (adaptado de [1]).

Para automatizar a escolha do dispositivo de áudio e sua combinação de *inputs* (*source*) e *outputs* (*sink*), podemos fazer uso de um *script*. Isso não será detalhado aqui, mas pode-se resumir que o caminho é criar um arquivo como "meu-script.sh", contendo comandos que definem entradas, saídas etc e então adicionar ao *bash* e executar o *script*. Diversos exemplos podem ser encontrados *online* e a Referência [15] pode dar alguma ideia de como funciona.

Para expor o PulseAudio aos comandos do ALSA, devemos:

- 1) verificar se os arquivos `pulse-alsa.conf` e `pulse.conf` existem e se estão configurados de maneira semelhante às Figuras 14 e 15;
- 2) caso não existam, verificar com o comando `pactl info` entradas saídas e outros parâmetros definidos como padrão pelo PulseAudio;
- 3) então usar essas informações para configurar o arquivo `asound.config` conforme Código 19;
- 4) ainda, você pode criar um *script* para automatizar suas escolhas dentro do PulseAudio.

## 6 Conclusões

Este documento trouxe à luz uma série de maneiras de se configurar seu Raspberry Pi 3 utilizando o sistema operacional Raspbian Stretch, da plataforma Linux. É importante frisar que a configuração muitas vezes deve ser feita a partir de uma investigação que associa certas noções básicas, ao levar em consideração os seguintes itens:

- muitas maneiras de se manejar ou simplesmente reconhecer dispositivos podem ser feitas de forma singular ou complementar;
- a necessidade de instalação de pacotes, bibliotecas e alguns módulos será constante e a checagem deve sempre existir, atentando-se sempre aos *logs* de erro;
- devemos compreender que as aplicações podem usar *software* específicos, como trabalhar apenas através do ALSA (nível de núcleo) ou necessitando da ajuda de um *middleware* como JACK ou PulseAudio. Aqui apresentaremos diferentes maneiras que podem auxiliar a se chegar no reconhecimento e utilização do dispositivo de áudio que você deseja conectar. Alguns comandos que podem ser úteis para a investigação:

Foram apresentadas as formas mais simples para a escolha do dispositivo, entradas e saídas; espera-se que futuramente uma nova edição deste Tutorial vá abranger mais temas interessantes, como configurações e módulos de áudio para Python em Raspbian e a adição de um capítulo para o JACK. Como visto na Figura 4, ambos ocupam mesmo nível hierárquico e a diferença entre o PulseAudio e o JACK é que o último costuma ser utilizado para fins profissionais, apesar do PulseAudio cumprir bem seu papel. Somado a isso, uma comparação com modos de configuração e novidades do Raspberry Pi 4 sugerem um bom caminho de investigação, ampliando a compreensão em diversos modelos e de uma maneira mais aprofundada.

## Referências

- [1] PulseAudio - ArchWiki, Maio 2019. <https://wiki.archlinux.org/index.php/PulseAudio>. [Online; acessado em 24 de Maio de 2019].  
(Citado nas páginas 3, 18, 25 e 26)
- [2] Modelos Raspberry Pi, Jan 2017. <https://fernandolopez6.wordpress.com/2017/01/23/modelos-raspberry-pi>. [Online; acessado em 3 de Maio 2019].  
(Citado na página 4)
- [3] Raspberry Pi and realtime, low-latency audio [Linux-Sound], Set 2017. [https://wiki.linuxaudio.org/wiki/raspberrypi#audio\\_software\\_repository](https://wiki.linuxaudio.org/wiki/raspberrypi#audio_software_repository). [Online; accessed 2. May 2019].  
(Citado na página 6)
- [4] RPi Expansion Boards - eLinux.org, Jan 2019. [https://elinux.org/index.php?title=RPi\\_Expansion\\_Boards#Sound](https://elinux.org/index.php?title=RPi_Expansion_Boards#Sound). [Online; acessado em 7 de Maio 2019].  
(Citado na página 6)
- [5] RPi VerifiedPeripherals - eLinux.org, Jan 2019. [https://elinux.org/RPi\\_VerifiedPeripherals#USB\\_Sound\\_Cards](https://elinux.org/RPi_VerifiedPeripherals#USB_Sound_Cards). [Online; acessado em 2 de Maio 2019].  
(Citado na página 6)
- [6] Testing Audio | USB Audio Cards with a Raspberry Pi | Adafruit Learning System, Maio 2019. <https://learn.adafruit.com/usb-audio-cards-with-a-raspberry-pi/testing-audio>. [Online; acessado em 22 de Maio de 2019].  
(Citado na página 8)
- [7] Burk, P. PortAudio - an Open-Source Cross-Platform Audio API, Jun 2019. <http://www.portaudio.com>. [Online; acessado em 20 de Junho de 2019].  
(Citado na página 10)
- [8] Alsa Opensrc Org - Independent ALSA and linux audio support site, Jun 2019. <https://alsa.opensrc.org/ALSA>. [Online; acessado em 20 de Junho de 2019].  
(Citado na página 10)
- [9] Phillips, Stephen C. Sound configuration on Raspberry Pi with ALSA. Stephen C Phillips, Jan 2013, <http://blog.scphillips.com/posts/2013/01/sound-configuration-on-raspberry-pi-with-alsa>.  
(Citado na página 11)
- [10] Alsa Opensrc Org - Independent ALSA and linux audio support site, Maio 2019. <https://alsa.opensrc.org/Device>. [Online; acessado em 23 de Maio de 2019].  
(Citado na página 13)
- [11] Understanding ALSA device, subdevice and cards., Maio 2019. <http://delogics.blogspot.com/2014/11/understanding-alsa-device-subdevice-and.html>. [Online; acessado em 23 de Maio de 2019].  
(Citado na página 13)
- [12] Asoundrc - AlsaProject, Fevereiro 2019. <https://www.alsa-project.org/wiki/Asoundrc>. [Online; acessado em 24 de Maio de 2019].  
(Citado nas páginas 16 e 17)

- [13] PulseAudio - Debian Wiki, Maio 2019. <https://wiki.debian.org/PulseAudio>. [Online; acessado em 7 de Maio de 2019].  
(Citado na página 18)
- [14] Gaydov, V. PulseAudio under the hood, Jun 2019. <https://gavv.github.io/articles/pulseaudio-under-the-hood/#command-line-tools>. [Online; acessado em 5 de Junho de 2019].  
(Citado nas páginas 21, 22 e 25)
- [15] PulseAudio/Examples - ArchWiki, Jun 2019. [https://wiki.archlinux.org/index.php/PulseAudio/Examples#The\\_shell\\_script\\_method](https://wiki.archlinux.org/index.php/PulseAudio/Examples#The_shell_script_method). [Online; acessado em 24 de Junho de 2019].  
(Citado na página 26)