

Cirrus Logic Audio Card User Documentation

Contents

1. **An introduction to the Cirrus Logic Audio Card for Raspberry Pi**
2. **Features**
3. **How to install the card to your to Raspberry Pi**
4. **Installing software to run on Raspberry Pi**
5. **Getting started with audio from your Cirrus Logic Audio Card**
6. **Soundcard usage scenarios**
7. **Annexe**
 - a. Datasheet of WM5102 audio CODEC
 - b. Datasheet of WM8804 SPDIF receiver/transmitter
 - c. Datasheet of WM7220 Digital Microphone Module
 - d. Schematic diagram of soundcard
 - e. Feature header signal assignment
 - f. Performance
 - g. Electrical architecture of the soundcard

1. Introduction to Cirrus Logic Audio Card for Raspberry Pi

Raspberry Pi, whilst being equipped with audio capability, remains limited in a number of ways. The limitations are intended to maintain Raspberry Pi's low price point, but still represent a limitation to users interested in exploring the audio capability of Raspberry Pi.

In terms of audio, there are no ways to capture audio using Raspberry Pi alone, and audio output is limited to two paths; analogue, via its onboard 3.5 mm stereo output jack, and digital, via its onboard HDMI output. Whilst the HDMI output provides the potential for high quality rendering of audio (depending on what HDMI devices are used to finally convert audio from its digital format to an analogue signal), the audio quality from the analogue 3.5 mm stereo output jack is universally recognised as being of an 'acceptable' quality level – no more.

The most important limitation is Raspberry Pi's lack flexibility in terms of multiple types of audio input sources, and outputs.

Early in 2014, Cirrus Logic acquired Wolfson Microelectronics, and Raspberry Pi released the new 40-pin GPIO featured, for example featured on models B+ and A+. In addition to the new GPIO configuration, the 8-pin P5 connector was also dropped, allowing for an altogether easier interconnect experience for accessories. The Cirrus Logic Audio card is therefore a replacement for the Wolfson Audio Card, and addresses the need for a high quality audio solution for 40-pin versions of Raspberry Pi. It contains the same core silicon and therefore has the same high performance as its predecessor, but has been improved in a number of ways.

2. Features

This product, designed by element14 and Cirrus Logic in partnership, addresses the above by providing a rich set of high quality audio features, including the following:

Features

- Compatible with Raspberry Pi B+ and A+ onwards (with 40-pin extended GPIO, and no P5 connector)
- Analogue line-level output and input
- Digital stereo audio input and output (SPDIF)
- High quality headphone output, with microphone facility (for headphones with boom microphone)
- Onboard stereo digital microphones based on MEMS technology
- Ability to render High Definition (HD) Audio
- Arrives bundled with five High Definition (HD) audio files to demonstrate the systems capability.
- Onboard power amplifier for directly driving loudspeakers. (Requires headers to be fitted.¹)
- Expansion header to allow connection to host boards other than Raspberry Pi.
- Back power protection, allows Raspberry Pi to be powered from the Cirrus Logic Audio card, allowing for convenient set up.
- New universal software support for both this card, and the former Wolfson Audio Card.

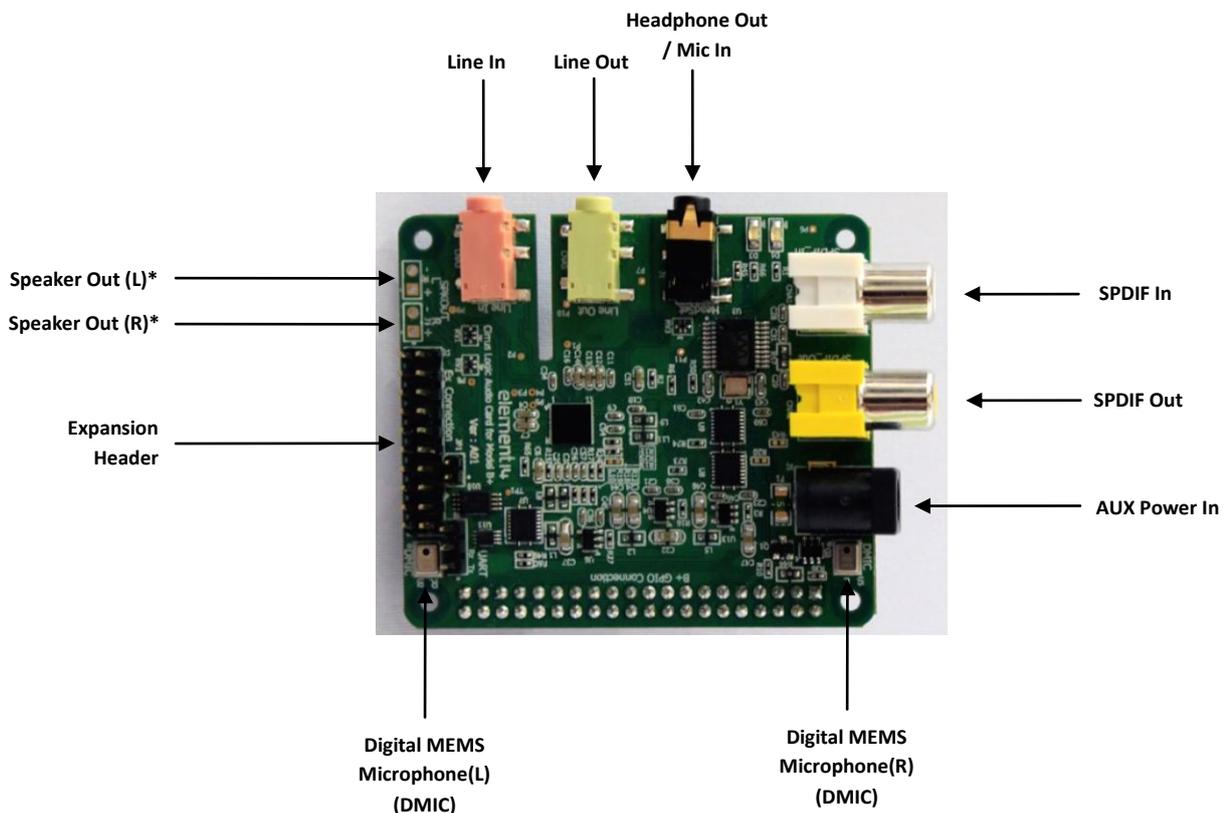


Diagram 1: Connections to and from the Cirrus Logic Audio Card

3. How to connect the Cirrus Logic Audio Card to Raspberry Pi

This Cirrus Logic soundcard has been designed to plug in to Raspberry Pi simply and easily. It is compatible with Raspberry Pi with 40-pin extended GPIO, such as models A+ and B+.

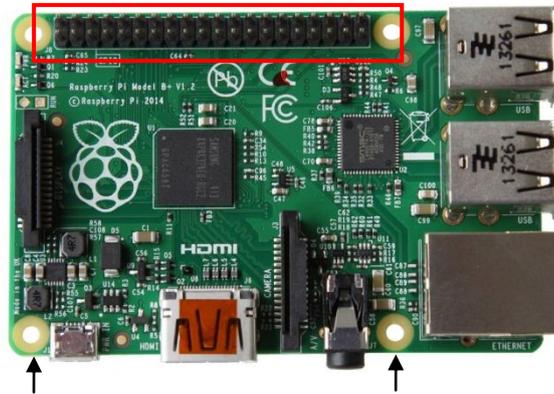


Diagram 2: Raspberry Pi with 40-pin extended GPIO connector outlined in red

The Cirrus Logic Audio Card simply pushes onto the top of Raspberry Pi. Inside the box, are two plastic pillars and four screws that facilitate this.

- a. Mount the two pillars at the two mounting holes identified in Diagram 2 above. Don't over tighten the screws.
- b. Push the Cirrus Logic Audio Card down onto Raspberry Pi's 40-pin connector, ensuring the pins are aligned.
- c. Use the remaining two screws to secure the board to the pillars underneath it.

4. Installing software to run on Raspberry Pi

Method A: Complete image install

In this approach, the software required to run the card is downloaded as a fully integrated all-in-one package that contains all the drivers and settings you need to work straight out of the box. **Please note that you will require a minimum 8 GB microSD card.**

The advantage is that installation is easy, and everything will work straight away. However, it means you may have a separate microSD Card for your Raspberry Pi audio solution, and that will need setting up.

1. Download the most recent image file from www.element14.com/cirruslogic_ac, and save it in a known location on your Windows computer. The file can be quite large (around 2GB), so be patient as it downloads.
2. The file **.img** has been compressed into a **.zip** format in order to minimise download time. That means, before you can do anything else, that file needs to be restored to its original form again. You may already have a zip/unzip tool installed on your computer (check by right-clicking on the .zip file – if you see the **Extract all...** option, then you have it and should use it), but if not, try **PeaZip** available at <http://peazip.com>. In the case of PeaZip, simply right click on the .zip file, and select **Peazip > Extract here**.
3. When you have done this, you get a single file called an Image File, or **.img**. This is a snapshot of what needs to be written to the microSD Card.
4. Next, you need a tool to install the **.img** file to the microSD Card. A popular tool is called **Win32DiskImager**, and it is available at this address <https://launchpad.net/win32-image-writer>. Simply download this program file and follow the instructions to install it on your Windows computer.
5. Follow the instructions provided with **Win32DiskImager** for writing the Operating System to your SD Card. This process will require the microSD Card to be inserted into a SD Card slot on your computer (probably with an SD Card adaptor), and for you to know where you saved the **.img** file on your computer. The write process itself can take some time.

WARNING: Make sure you select the correct device to write the Operating System to. Selecting the wrong destination could result in unrecoverable data loss. Be careful and DOUBLE CHECK.

6. **Win32DiskImager** will tell you when this process has been completed and when it has, remove the SD Card. Now insert it into the SD card slot of your Raspberry Pi. It is worth keeping the Operating System file on your computer for future rewrites to your SD Card, just in case.

7. Now, power up your Raspberry Pi and Cirrus Logic Audio Card.
8. The **red LED** will light up on the Raspberry Pi, indicating the main chip has started up, and then the **green LED** will begin to flash, indicating data is being read from/written to the microSD Card. The TV/monitor (provided it's switched on of course) will begin to show the Linux boot sequence.
9. This image has been designed to boot into the Graphical User Interface, so that when the boot process has been completed, the Raspbian desktop will be presented. Once this appears, you are ready to go.
10. In order to maximise the space available on the microSD card, it is recommended that you perform a file system expand operation. This can be achieved as follows:
 - a. Either start LXterminal on the desktop, or use the command line screen, and type the following command:

```
sudo raspi-config <Enter>
```
 - b. Select the first option, Expand Filesystem, and allow the process to complete.
 - c. At the next reboot, the file system size will have been resized to fit the microSD card.

Method B: Kernel patch set install

In this method, users will take a kernel patch set written especially for the audio card. The kernel patch set will be installed into your existing OS using a documented method. This approach is more of an upgrade path, and will suit users who are confident and willing to try it, or advanced users who know what they are doing.

The advantage of this method is that you are not creating a separate new system, but upgrading your current one. Therefore, changes you might have made to your OS or applications will be preserved using this method.

Full documentation for this approach is contained at a dedicated **Cirrus Logic GitHub** page, address <https://github.com/cirruslogic/rpi-linux/wiki>, where it is maintained.

Notes on this section:

- Please note that this new version of the software supports both this card, and the former Wolfson Audio Card.
- The kernel patch is being up-streamed to the Linux repositories, such that in future, they will already be available when an OS for Raspberry Pi is downloaded.
- A selection of bonus HD Audio tracks are preinstalled for you to play as soon as you get up and running. They are also available at www.element14.com/cirruslogic_ac.
- Once the system has booted up, it is suggested that one complete reboot cycle is performed before attempting any of the next steps.
- **NOTE:** A `rpi -update / upgrade` will currently result in the replacement of a kernel which does NOT support the audio card, hence audio card support will be lost if this action is completed. Please check back at www.element14.com/cirruslogic_ac for the status of Raspberry Pi kernel upstreaming integration.

5. Getting started with audio on your Cirrus Logic Audio Card

The Cirrus Logic Audio Soundcard renders and captures high quality audio from a variety of sources. It is designed to work with the Linux operating system, and as such takes advantage of ALSA, the Advanced Linux Sound Architecture. You can read more about this at www.alsa-project.org.

In the OS download referred to in section 4. above, the supported audio player is **LXmusic**, and labelled **Music Player** on the desktop. In general, there are two main stages to performing an audio task, whether that is to play a file, or record one:

- a. The **Use Case** has to be defined first, so that the sound device is configured properly. A Use Case is simply a set of commands that are run in series, which set the audio card up. They define the signal paths for any audio that is played or recorded. So, for example, if you want to play audio to speakers, you'd run the script for that case. A script is a collection of commands that can be run in a single instruction.
- b. The action to **play/record** can then be executed.

The Use Case definition is set up by running one of a set of scripts that are supplied in the software package. It is done this way:

- a. To run any kind of script, it is necessary to start a **Terminal** session. Double click on **LXTerminal** on the desktop (or indeed, any other terminal programme you prefer).
- b. The terminal allows you type commands directly instructing the OS to form operations you want to perform. Start by typing **dir**. That will give you a list of files and folders that are contained in the home directory that the Terminal session opens in.
- c. A set of use cases have been put placed into this directory for you, as follows:

#	Audio Function	Script to run
1.	Play an audio file to your headphones	Playback_to_Headset.sh
2.	Play an audio file to the line out jack	Playback_to_Lineout.sh
3.	Play an audio file to loudspeakers	Playback_to_Speakers.sh
4.	Play an audio file to the SPDIF output of your card	SPDIF_playback.sh
5.	Record from onboard digital microphones	Record_from_DMIC.sh
6.	Record from the boom microphone of your gamer/VoIP headphones	Record_from_Headset.sh
7.	Record from the analogue Line In socket	Record_from_lineIn.sh
9.	Record from the SPDIF input of your card	SPDIF_record.sh
10.	Reset all audio paths	Reset_paths.sh
11.	Record from line in socket at microphone sensitivity level, and with mic bias voltage switched ON	Record_from_lineIn_Micbias.sh

Table 1: A list of the Use Case Scripts for setting up the audio card.

As an example, let us say that you want to play one of the bundled High Definition audio tracks that are included in the software package, and you want to play it to your headset. In this case, and at the command line in **LXTerminal**, type the following command:

```
./Playback_to_Headset.sh <Enter>
```

The script will run, and you will be returned to the command prompt again. If at any time during the time the script runs, you see the following message displayed on the screen:

amixer: Cannot find the given element from control hw:0

This means the card was not detected and registered properly during start up. It is recommended in this case that you check the card is properly secured to your Raspberry Pi, and perform a hard reboot (issue a **sudo halt**, switch of the power, re-apply power and allow to boot up).

Assuming all was well, your system has now been configured to play audio.

Next, start **Music Player** by double-clicking on it on the desktop. The programme will launch. To select a file for playback, it must first be added to the playlist. Select **Playlist > Add To Playlist > Add Files or Folders**, and you will be presented with a dialogue box where you can navigate to the location of your music files. In the case of the bundled HD Audio tracks, these are located in the **Home** directory.

Select the file(s) you want to play, and then click on the **Add** button.

The files you added will now be available for playback in the Playlist.

Finally, plug in your headphones into the Headphone socket.

Now, double-click on the item you wish to play, and **Music Player** will play the audio track to your headphones.

The scripts for playback are cumulative in nature. So, taking the above example into account, where we have enabled playback to a headset, if we then run the script to play audio to the line out jack as well with the following command:

```
Playback_to_Lineout.sh <Enter>
```

What happens is that audio playback is enabled for both headset *and* line out.

If that is not what you want, and simply want to reset all the paths back to NO playback paths enabled, then just run the following command first:

```
Reset_paths.sh <Enter>
```

Now, select the single output that you want to play to.

Notes on this section:

- **If you are recording to, or playing back from your SD Card, it is important that you use an SD Card that is Class 6 and above. The class directly correlates with speed of reading and writing to the card. When playing HD Audio, the rate of data being read from the**

card is significant, and a low card will leave occasional gaps in the audio experience. The situation is similar when recording.

- Whilst the onboard hardware CODEC is capable of handling multichannel audio, Raspberry Pi is not. Therefore, the card will not allow the decoding of compressed multichannel audio via its SPDIF in connector.
- The system will remember what the last used use case was. After a reboot, it will return to that one unless you choose another.

6. Soundcard usage scenarios

The Cirrus Logic Audio Card is flexible with regards to how it can be connected to various types of input devices (sources), and output devices (sinks). If you'd prefer to keep your audio in the digital domain, the SPDIF digital audio connections allow this. High quality analogue audio capture and playback is also made available with this card.

▪ **Connecting speakers to your Audio Card**

The Cirrus Logic Pi Audio Card contains an on board Class D power amplifier, capable of delivering up to 1.4W per channel, directly to external loudspeakers. The connectors required to allow this are in locations J6 (left) and J3 (right), but they are not populated as standard.

¹ You can solder your own pair of male 0.1" headers, by purchasing from Farnell/Newark/element14. Suggested models are as follows:

Region	Distributor	Part Number
North America	Newark	60H4177
Europe	Farnell	9733302
Asia Pacific	element14	9733302

Table 2: Suppliers of 0.1" headers.

Warning:

- **Once fitted, be extremely careful not to accidentally short the speaker pins. Doing so could damage the Audio Card.**
- **It is necessary to supply auxiliary power to the Audio Card in order to drive external loudspeakers. An external power supply with output voltage of 5V, current delivery capability of 2,000mA, a centre positive power plug of internal diameter 2.1mm, and external diameter 5.5mm.**

With your Raspberry Pi powered down, connect speakers to the newly fitted headers. Make sure your speakers are connected in phase. To do this, observe on the header pins that one pin is marked with a white dot. Your speaker cable should be polarised as well, with perhaps one core being coloured red, and the other black. Be consistent with how you connect, for example, the red cable goes to the connector with the dot on both pins.

Note that in this configuration, and with a substantial enough power supply, the Cirrus Logic Audio Card will power Raspberry Pi with a single power input.

Before playing a file to your speakers, you must run the appropriate Use Case Script to play to loudspeakers. Then play a file using the **LXmusic player**.

- **Connecting your audio card to a SPDIF sink**

You can connect your audio card to digital amplifiers and receivers (also known collectively as 'sinks') using the phono/RCA/Cinch SPDIF output jack. The Cirrus Logic Audio Card can deliver audio in bit depths of up to 24-bit. Make sure your sink can support resolutions this high.

Warnings:

- **Be very careful when setting the output gain and volume control on your amplifier, especially if it is a powerful type. Loudspeaker damage can easily occur if the volume is too high.**

- **Connecting your audio card to a Line level sink**

You can connect your audio card to analogue amplifiers and receivers by using the green 3.5mm line output jack. The use case scripts have been set up to provide a line out level that is not too high, and should be suitable for most amplifiers and receivers.

- **Connecting your audio card to a Line level source**

Capturing audio using your Cirrus Logic Audio Card is straightforward as well. In the case of an analogue line level source, simply connect your analogue source to the Audio Card by connecting to the red 3.5 mm input jack. Then, run the appropriate use case to set up the audio card routing and gains properly.

You can capture audio to a file by using the **arecord** command at a command prompt on Raspberry Pi. The **arecord** command will allow you to flexibly capture live audio, and saves it directly to Raspberry Pi's mass storage memory as an uncompressed audio file.

For example, if you would like to capture audio from the line in jack, first of all, it is necessary to tell the audio card to switch audio into the card from that source. This is done by running the following command:

```
./Record_from_lineIn.sh <Enter>
```

Now, we need to use the arecord command to capture the audio to a specified file as follows:

```
arecord -c 2 -f S16_LE -r 44100 record_from_line_in.wav  
<Enter>
```

[The `.asoundrc` file (in the Pi home directory) sets the Cirrus Logic Audio card as its default audio device. If the `.asoundrc` file is not present, the following more absolute command will be required instead:

```
arecord -Dhw:0 -c 2 -f S16_LE -r 44100  
record_from_line_in.wav <Enter>
```

If you want to ensure the sound card is the selected device the full command would be

```
arecord -Dhw:sndrpiwsp -c 2 -f S16_LE -r 44100  
record_from_line_in.wav <Enter>
```

As a useful exercise, it is worth having a look at some of the example user scripts in the Pi directory. You could use these as the basis of new scripts that you want to create. You'll see that will also see that explicit reference is made to `sndrpiwsp` in these setup scripts.]

(Note: the above should be entered as a single line, and not two lines as shown here.)

Recording will start. Press CTRL-C to stop recording.

A brief explanation of the options above is as follows:

- c = Number of channels. Here, we specify two, for stereo.
- f = Recording format. In this case, we are asking for 16-bit, Little Endian.
- r = Sample rate. We ask here for 44,100 audio samples per second.

To find out more about this command, type:

```
arecord --help <Enter>
```

If you wish to compress the audio file into other formats, there are other programmes that can be installed on Raspberry Pi to do this. **ffmpeg** is such a programme, available at www.ffmpeg.org. Its use and implementation is beyond the scope of this document.

A General Warning If Editing Use Case Scripts

- **Use case scripts can be adjusted to accommodate various types of sources.**
- **In particular, the gain of those inputs can be adjusted to make them very sensitive.**
- **The result can be an extremely LOUD output signal that, if connected to power amplifiers, could create permanent equipment damage. Turn the volume down!**
- **It is always advisable to reduce the volume of any amplifier, or active speaker connected to this card, before playing back any audio tracks.**

7. Annexe

a. **Datasheet of WM5102 audio CODEC**

The key technology that is used at the heart of the Cirrus Logic Audio Card, is an Audio Hub (CODEC) device. The WM5102 is a highly-integrated low-power audio system that has been designed for use in smartphones, tablets and other portable audio devices. It combines wideband telephony voice processing with a flexible, high-performance audio hub CODEC, and is capable of delivering HD audio. If you'd like to know more about this device, take a look at it here:

www.element14.com/community/docs/DOC-65664?ICID=Pi-Accessories-Cirrus-Logic-audio-space

b. **Datasheet of WM8804 SPDIF receiver/transmitter**

The WM8804 allows digital audio data to be transmitted and received to the audio card via the phono/RCA jacks, and works in partnership with the WM5102. You can read more about this device by checking out the following datasheet:

www.element14.com/community/docs/DOC-65664?ICID=Pi-Accessories-Cirrus-Logic-audio-space

c. **Datasheet of WM7220 Digital Microphone Module (MEMS)**

The WM7220 is a Digital Microphone Module (DMIC), that is based on MEMS technology. MEMS, or Micro-Electrical-Mechanical Systems, are devices that combine small scale electronics on a silicon die, together with a mechanical component that can interact with the physical environment. In this case, the mechanical element responds to acoustic pressure waves and converts them in to a digital audio data stream. Read more here:

www.element14.com/community/docs/DOC-65664?ICID=Pi-Accessories-Cirrus-Logic-audio-space

d. **Schematic diagram**

If you are interested in knowing how the Cirrus Logic Audio Card has been designed, the schematic diagram is available to view as a .pdf document, on the element14.com website at the following address:

www.element14.com/community/docs/DOC-65689?ICID=Pi-Accessories-Cirrus-Logic-audio-space

e. 26-pin versus 40-pin GPIO header signal assignment

The table below gives a detailed overview of the signals available on the GPIO header. The blue column calls out details for Model B, and the green column Models A+/B+. CLAC is an acronym for Cirrus Logic Audio Card.

P1 Pin	Name	Model B		Model A+/B+	
		GPIO (BCM)	WAC Function	GPIO (BCM)	CLAC Function
1	3V3 Power				
2	5V0 Power				
3	SDA1	GPIO2	WM8804 I2C - SDA	GPIO2	WM8804 I2C - SDA EXP/11
4	5V0 Power				
5	SCL1	GPIO3	WM8804 I2C - SCLK	GPIO3	WM8804 I2C - SCLK EXP/9
6	Ground				
7	GPIO_GCLK	GPIO4	WM5102 MCLK2 (unused) WM8804 MCLK (unused)	GPIO4	WM5102 MCLK2 (unused) WM8804 MCLK (unused)
8	TXD0 (UART)	GPIO14	J8/1	GPIO14	J8/1
9	Ground				
10	RXD0 (UART)	GPIO15	J8/2	GPIO15	J8/2
11	GPIO_GEN0	GPIO17	WM5102 RST	GPIO17	WM5102 RST
12	GPIO_GEN1 (PCM_CLK)	GPIO18	WM8804 - I2C Address Config	GPIO18	WM5102 AIF PCM - BCLK
13	GPIO_GEN2	GPIO27	WM5102 IRQ (unused)	GPIO27	WM5102 IRQ (unused)
14	Ground				
15	GPIO_GEN3	GPIO22	WM5102 LDO Enable	GPIO22	WM5102 LDO Enable
16	GPIO_GEN4	GPIO23	WM8804 - Control I/F Config	GPIO23	WM8804 - Control I/F Config
17	3V3 Power				
18	GPIO_GEN5	GPIO24	WM5102 GPIO5 (unused)	GPIO24	WM5102 GPIO5 (unused)
19	SPI_MOSI	GPIO10	WM5102 SPI - MOSI	GPIO10	WM5102 SPI - MOSI
20	Ground				
21	SPI_MISO	GPIO9	WM5102 SPI - MISO	GPIO9	WM5102 SPI - MISO
22	GPIO_GEN6	GPIO25	WM5102 GPIO3 (unused) EXP/16	GPIO25	WM5102 GPIO3 (unused) EXP/16
23	SPI_SCLK	GPIO11	WM5102 SPI - SCLK1	GPIO11	WM5102 SPI - SCLK1
24	SPI_CE0_N	GPIO8	WM8804 RST	GPIO8	WM8804 RST
25	Ground				
26	SPI_CE1_N	GPIO7	WM5102 SPI - CE	GPIO7	WM5102 SPI - CE
27	ID_SD			GPIO0	EEPROM
28	ID_SC			GPIO1	EEPROM
29	GPIO5			GPIO5	EXP/14

30	Ground				
31	GPIO6			GPIO6	EXP/17
32	GPIO12			GPIO12	EXP/18
33	GPIO13			GPIO13	WM8804 - I2C Address Config
34	Ground				
35	GPIO19 (PCM_FS)			GPIO19	WM5102 AIF PCM - FS
36	GPIO16			GPIO16	EXP/19
37	GPIO26			GPIO26	EXP/20
38	GPIO20 (PCM_DIN)			GPIO20	WM5102 AIF PCM - DIN
39	3V3 Power				
40	GPIO21 (PCM_DOUT)			GPIO21	WM5102 AIF PCM - DOUT

Note: Unused functions need to be treated carefully. Although they are not enabled in the current driver they provide scope for increased functionality going forward.

Table 3: GPIO signal definitions

f. Feature header signal assignments

The table below shows the signals available on the feature header. Please note that I2C and SPI are functions utilised on the card so these cannot be changed for other functions.

	Cirrus Logic Audio Board Pin	RPi GPIO Description	RPI Alternate Description	RPI pin
1	EX 9	GPIO 3	SCL I2C	5
2	EX 11	GPIO 2	SDA1 I2C	3
3	J9-19	GPIO 10	SPI_MOSI	19
4	J9-21	GPIO 9	SPI_MISO	21
5	J9-23	GPIO 11	SPI_SCLK	23
6	EX 14	GPIO 5		29
7	EX 16	GPIO 25		22
8	EX 17	GPIO 6		31
9	EX 18	GPIO 12		32
10	EX 19	GPIO 16		36
11	EX 20	GPIO 26		37
12	J8-1	GPIO 14	UART TX	8
13	J8-2	GPIO 15	UART RX	10

Table 4: Feature header signal assignments.

g. Audio interface (AIF1) specification

Audio Interface (AIF1)			
Parameter	Description	Typical Value	Units
Supported Sample Rates	8, 12, 16, 24, 32, 48, 96, 192, 11.025, 22.05, 44.1, 88.2, 176.4		kHz
Sample depth	Min bit depth of samples	16	bits
Sample depth	Max bit depth of samples	32	bits
Notes	The minimum supported sample rate is dependent on the SPDIF configuration. If the SPDIF Input or Output signal paths are enabled then the minimum sample rate will be 32kHz.		

h. Performance and Specification

During the design process, the board layout was optimised to ensure each of the Cirrus Logic Microelectronics devices performed optimally, providing users with an exceptional audio experience. Measured performance of the board is as follows:

SPDIF Output			
Parameter	Description	Typical Value	Units
Connector	Electrical output via phono connector		
Rout	Output impedance	75	Ohms
Supported Sample Rates	32, 48, 96, 192, 44.1, 88.2, 176.4, 192		kHz
Sample depth	Max bit depth of samples	24	bits
Notes	Sample rate must be the same as TX if both are enabled concurrently		

SPDIF Input			
Parameter	Description	Typical Value	Units
Connector	Electrical input via phono connector		
Rin	Input impedance	75	Ohms
Supported Sample Rates	32, 48, 96, 192, 44.1, 88.2, 176.4, 192		kHz
Sample depth	Max bit depth of samples	24	bits
Notes	Sample rate must be the same as RX if both are enabled concurrently		

Line Output			
Parameter	Description/Conditions	Typical Value	Units
Connector	Electrical output via 3 pole 3.5mm Socket		
Rout	Output impedance	16	Ohms
Cload	Max capacitive load on output	2	nF
Rload		10	kOhms
Vout	Full scale output signal level	1	Vrms
Vnoise	Noise Floor, no signal applied, A-weighting filter	4.5	uVrms
THD	Total Harmonic Distortion	0.005	%
Notes	Test conditions as follows unless otherwise noted: 48kHz sample rate, test signal 1kHz sine wave, bandwidth measured 20Hz to 20kHz, Rload = 10kOhms		

Line Input			
Parameter	Description	Typical Value	Units
Connector	Electrical input via 3 pole 3.5mm Socket		
Rin	Input Impedance	16	kOhms
Fcut	-3dB down from 1kHz signal level at this frequency	13	Hz
Vin	Full scale input signal	1	Vrms
Vnoise	Equivalent input noise level, no signal applied	20	uVrms
THD	Total Harmonic Distortion @ 0.9Vrms Input	0.035	%
Notes	Test conditions as follows unless otherwise noted: 48kHz sample rate, test signal 1kHz sine wave and bandwidth measured is 20Hz to 20kHz		

Headphone Output			
Parameter	Description/Conditions	Typical Value	Units
Connector	Electrical output via 4 pole 3.5mm Socket		
Rout	Output impedance	0.2	Ohms
Cload	Max capacitive load on output	130	pF
Vout	Full scale output signal level (note that the headphone use case script limits this to 0.5Vrms for safety)	1	Vrms
Vnoise	Noise Floor, no signal applied, A-weighting filter	2.3	uVrms
THD	Total Harmonic Distortion @ 400mVrms output	0.0035	%
Notes	Test conditions as follows unless otherwise noted: 48kHz sample rate, test signal 1kHz sine wave, bandwidth measured 20Hz to 20kHz, Rload = 32Ohms		

Headset Input			
Parameter	Description	Typical Value	Units
Connector	Electrical input via 4 pole 3.5mm Socket, MIC on Sleeve		
Vbias	Microphone voltage bias level	2.8	V
Rbias	Microphone bias resistor	2.2	kOhms
Fcut	-3dB down from 1kHz signal level at this frequency	13	Hz
Vin	Full scale input signal (500mVrms to 14mVrms depending on input gain setting)	63	mVrms
Vnoise	Equivalent input noise level, no signal applied, (settings for 63mVrms signal)	3.2	uVrms
THD	Total Harmonic Distortion @ 57mVrms Input	0.01	%
Notes	Test conditions as follows unless otherwise noted: 48kHz sample rate, test signal 1kHz sine wave and bandwidth measured is 20Hz to 20kHz		

Speaker Output			
Parameter	Description/Conditions	Typical Value	Units
Connector	Unpopulated 2.54mm pitch headers		
Rout	Output impedance	0.4	Ohms
Rload	Minimum Rload	3	Ohms
Power Out	Max power output	1.4	W
Vnoise	Noise Floor, no signal applied, A-weighting filter	30	uVrms
THD	Total Harmonic Distortion @ 0.5W output	0.06	%
Notes	Test conditions as follows unless otherwise noted: 48kHz sample rate, test signal 1kHz sine wave, bandwidth measured 20Hz to 20kHz, Rload = 8 Ohms, SPKVDD=5V		

d. Electrical architecture of the audio card

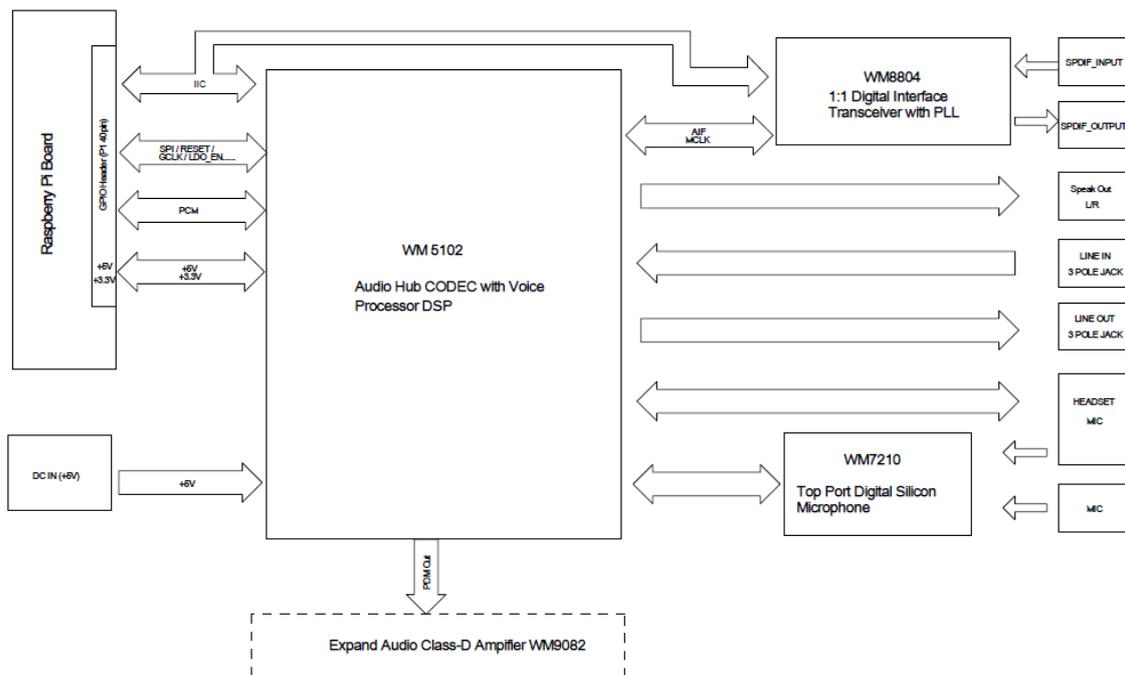


Diagram 3: Audio architecture of the Cirrus Logic audio card

Revisions

Revision	Created	Revision
1.00	09-Dec-2014	<ul style="list-style-type: none">▪ Re-written for Cirrus Logic Audio Card for B+ and A+ onwards.
1.01	12-Dec-2014	<ul style="list-style-type: none">▪ Includes advice that new software is backwards compatible with Wolfson Audio Card.
1.02	29-Jan-2015	<ul style="list-style-type: none">▪ Clarifies and corrects errors relating to the expansion header.▪ Adds comparison between model B GPIO and models A+/B+.▪ Addition of audio interface (AIF1) specification table.