Soundcard configuration

toc

bs-flaggedrevsconnector-addstabledatetochapterheadlinesmodifier-laststable-tag-text: 03.03.2021 - 17:27 / bs-flaggedrevsconnector-addstabledatetochapterheadlinesmodifier-stablerevisiondate-tag-text: 24.02.2021 - 13:29

toc

1 Overview

This article explains how to configure ST audio peripherals, as well as STM32MP1 boards external audio components, when they are assigned to the **Linux® OS**. In such cases, they are controlled by the ALSA framework.

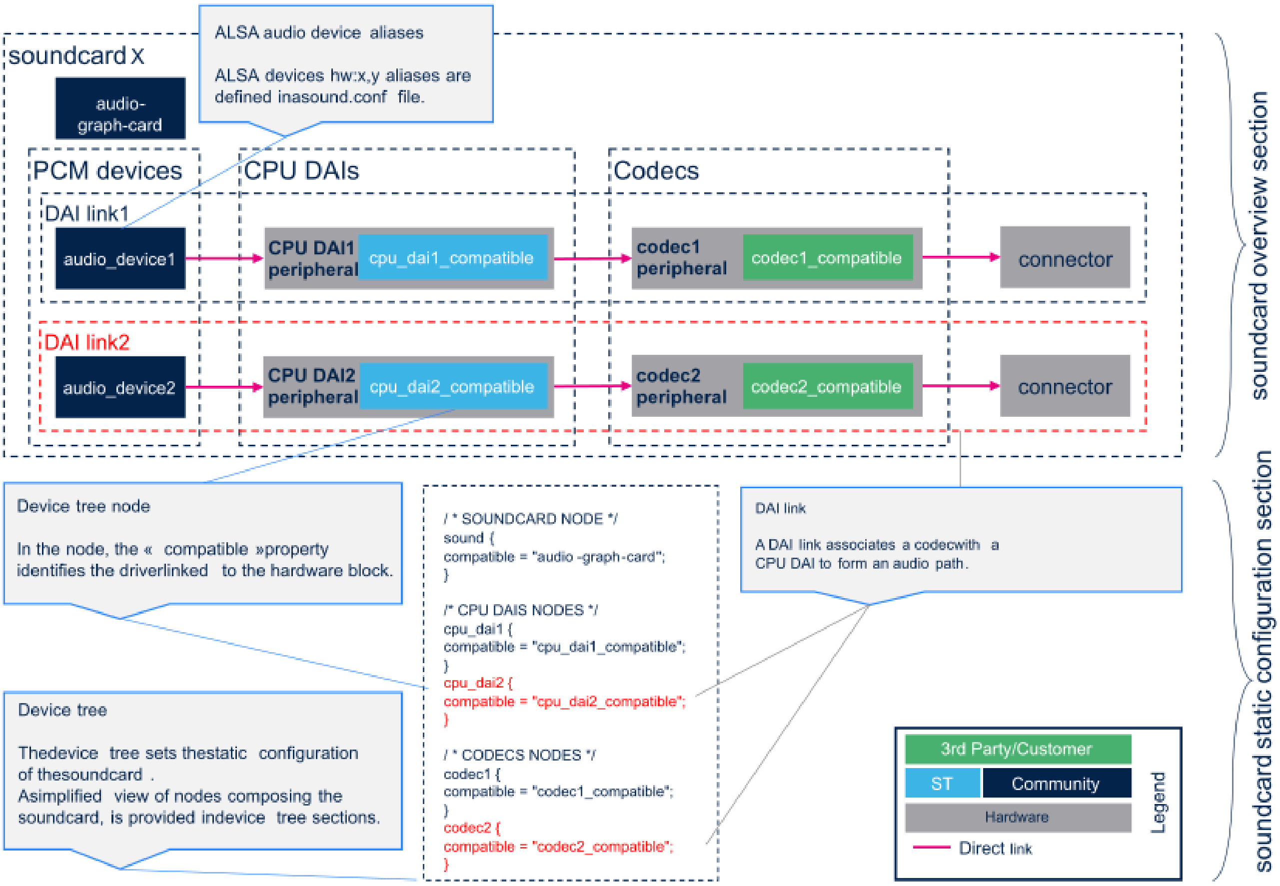
In the ASoC layer of the ALSA framework, audio hardware components are described as CPU DAIs and codec, which are linked together to create DAI links. A sound card is a software component gathering a set of DAI links.

Each of the following STM32 MPU board sections describes one or more sound cards. A schematic for each sound card is provided, as well as its means of static and dynamic configuration.

1.1 Sound card schematic

The sound card schematic gives an overview of the hardware and software components forming the sound card, and their relationships.

The example sound card schematic given below emphasizes the links between the sound card and the device-tree section.



1.2 Static configuration

- Device tree

The device tree allows the description, configuration and connection of the audio hardware components to define the sound card. The user has to follow the audio graph card bindingscite_reference_link to configure the sound card and device graph bindingscite_reference_link to connect audio components. The user must also refer to the audio component (codec and CPU DAI) bindings to configure these components properly. The bindings of the audio components can be found in device-tree samples in following sections and in the References chapter.

The STMicroelectronics configuration tool STM32CubeMX, allows the generation of CPU DAI device tree nodes.

STM32CubeMX does not allow configuration of sound card and codec nodes, which are board dependent. The sound card node and the codec nodes have to be filled manually through user sections.

- asound.conf cite_reference_link

The optional asound.confcite_reference_link system-global custom settings file, provides extra functionalities, such as routing and audio sample conversion. It can be found in the /etc directory.

- Sound card configuration files cite_reference_link

The alsa-lib layer provides card configuration files in /usr/share/alsa/cards directory. These files allow to map ALSA hardware devices on standard devices, such as "front", "hdmi" or "iec958" devices. The label defined in sound card device tree node defines the name of the card. The card configuration is retrieved from this card name, according to /usr/share/alsa/cards/aliases.conf mapping.

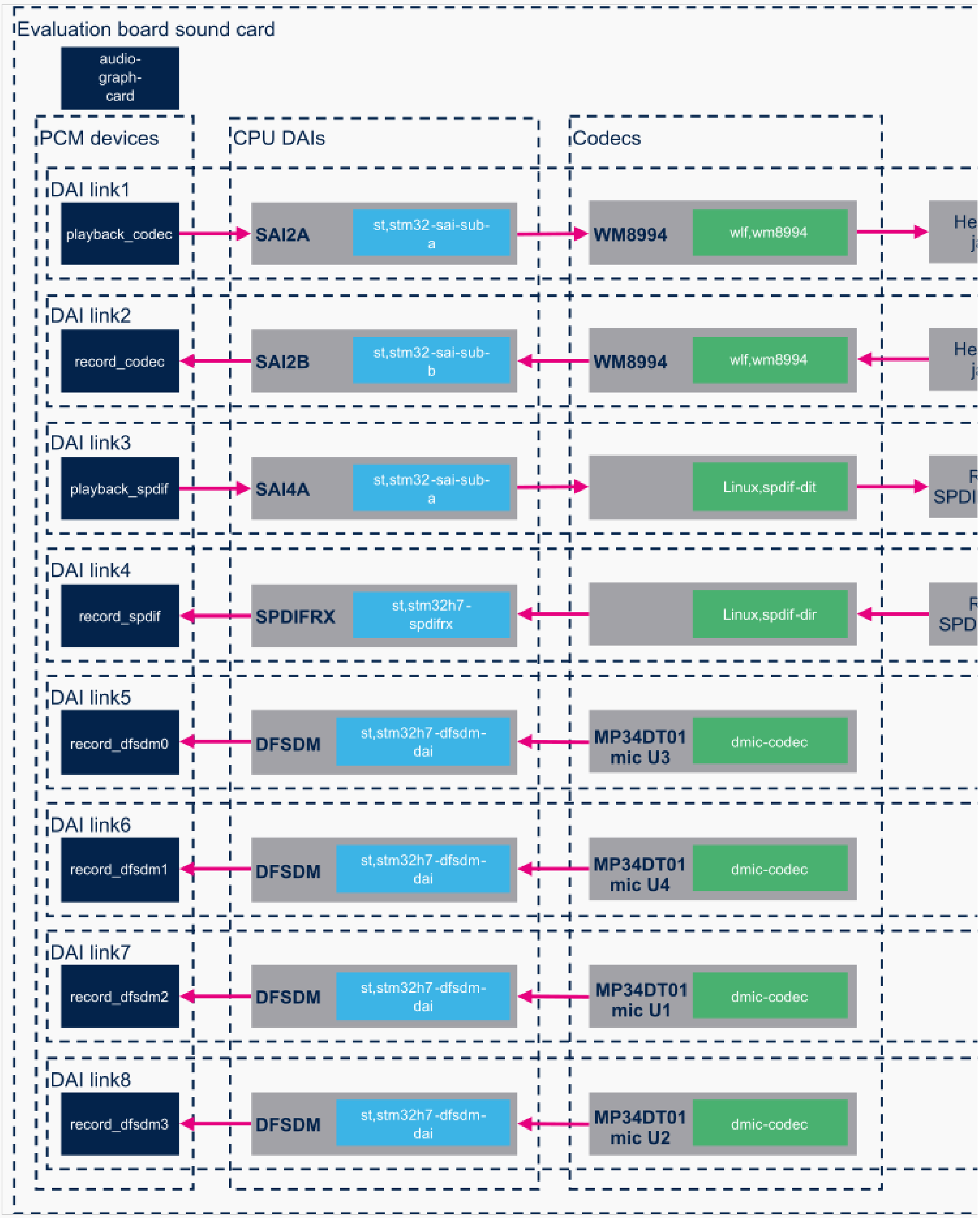
1.3 Dynamic configuration

The codecs and CPU DAI drivers also provide ALSA controls, allowing dynamic configuration of the sound card. The controls can be changed at runtime through the amixer utility to modify certain settings in the audio path. For instance, such controls can be used to modify the audio volume or the mute state of a block in the codec.

A customized configuration of these controls can be saved in the asound.state configuration file using the alsactl utility. This configuration can be restored at boot time through alsactl. STM32MPU sound cards come with a dedicated asound.state configuration file providing relevant control settings.

2 STM32MP15 evaluation board sound card configuration

2.1 Sound card overview



STM32MP15 Evaluation board sound card

2.2 Static configuration

The extract below is from the STM32MP15 evaluation board device tree. Only the nodes associated to the sound card, and the most relevant properties are shown here. For example, the properties linking nodes to form the first DAI link are emphasized with green font.

```
/ * SOUNDCARD */
        sound {
                compatible = "audio-graph-cardcite_reference_link";
                label = "STM32MP15-EV";                         /* Sound card identified
as STM32MP15EV in ALSA */
                routing =
                        "AIF1CLK" , "MCLK1",
                        "AIF2CLK" , "MCLK1",
                        "IN1LN" , "MICBIAS2",
                        "DMIC2DAT" , "MICBIAS1",
                        "DMIC1DAT" , "MICBIAS1";
                dais = <&sai2a_port &sai2b_port &sai4a_port &spdifrx_port
                        &dfsdm0_port &dfsdm1_port &dfsdm2_port &dfsdm3_port>;
        };

/ * CODECS */
        spdif_out: spdif-out {
                compatible = "linux,spdif-ditcite_reference_link";

                spdif_out_port: port@0 {
                        spdif_out_endpoint: endpoint {
                                remote-endpoint = <&sai4a_endpoint>;
                        };
                };
        };

        spdif_in: spdif-in {
                compatible = "linux,spdif-dircite_reference_link";
                spdif_in_port: port@0 {
                        spdif_in_endpoint: endpoint {
                                remote-endpoint = <&spdifrx_endpoint>;
                        };
                };
        };

        dmic0: dmic@0 {
                compatible = "dmic-codec";
                port {
                        dmic0_endpoint: endpoint {
                                remote-endpoint = <&dfsdm_endpoint0>;
                        };
                };
        };

        dmic1: dmic@1 {
                compatible = "dmic-codec";
                port {
                        dmic1_endpoint: endpoint {
                                remote-endpoint = <&dfsdm_endpoint1>;
                        };
                };
        };

        dmic2: dmic@2 {
                compatible = "dmic-codec";
                port {
                        dmic2_endpoint: endpoint {
                                remote-endpoint = <&dfsdm_endpoint2>;
```

```
                        };
                };
        };

        dmic3: dmic@3 {
                compatible = "dmic-codec";
                port {
                        dmic3_endpoint: endpoint {
                                remote-endpoint = <&dfsdm_endpoint3>;
                        };
                };
        };
};

&i2c2 {
        wm8994: wm8994@1b {
                compatible = "wlf,wm8994";
                ...
                clocks = <&sai2a>;
                clock-names = "MCLK1";

                ports {
                        #address-cells = <1>;
                        #size-cells = <0>;

                        wm8994_tx_port: port@0 {
                                wm8994_tx_endpoint: endpoint {
                                        remote-endpoint = <&sai2a_endpoint>;
                                };
                        };

                        wm8994_rx_port: port@1 {
                                wm8994_rx_endpoint: endpoint {
                                        remote-endpoint = <&sai2b_endpoint>;
                                };
                        };
                };
        };
};

/* CPU DAIS */
&sai2 {
        clocks = <&rcc SAI2>, <&rcc PLL3_Q>, <&rcc PLL4_Q>;
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&sai2a_pins_a>, <&sai2b_pins_a>;
        pinctrl-1 = <&sai2a_sleep_pins_a>, <&sai2b_sleep_pins_a>;
        clock-names = "pclk", "x8k", "x11k";

        sai2a: audio-controller@4400b004 {
                compatible = "st,stm32-sai-sub-a";
                dma-names = "tx";                                       /* SAI set as
transmitter */
                clocks = <&rcc SAI2_K>;
                clock-names = "sai_ck";

                sai2a_port: port@0 {
                        sai2a_endpoint: endpoint {
                                remote-endpoint = <&wm8994_tx_endpoint>;
                                format = "i2s";
                                mclk-fs = <256>;                /* SAI is master clock
provider */
                        };
                };
        };

        sai2b: audio-controller@4400b024 {
                compatible = "st,stm32-sai-sub-b";
```

```
                dma-names = "rx";                                /* SAI set as receiver */
                clocks = <&rcc SAI2_K>, <&sai2a>;
                clock-names = "sai_ck", "MCLK";

                sai2b_port: port@0 {
                        sai2b_endpoint: endpoint {
                                remote-endpoint = <&wm8994_rx_endpoint>;
                                format = "i2s";
                                mclk-fs = <256>;                /* SAI is master clock
provider */
                        };
                };
        };
};

&sai4 {
        clocks = <&rcc SAI4>, <&rcc PLL3_Q>, <&rcc PLL4_Q>;
        clock-names = "pclk", "x8k", "x11k";

        sai4a: audio-controller@50027004 {
                compatible = "st,stm32-sai-sub-a";
                dma-names = "tx";
                st,iec60958;                                        /* SAI configured for
S/PDIF protocol*/
                pinctrl-names = "default", "sleep";
                pinctrl-0 = <&sai4a_pins_a>;
                pinctrl-1 = <&sai4a_sleep_pins_a>;
                 clocks = <&rcc SAI4_K>;
                clock-names = "sai_ck";

                sai4a_port: port@0 {
                        sai4a_endpoint: endpoint {
                                remote-endpoint = <&spdif_out_endpoint>;
                        };
                };
        };
};

&spdifrx {
         compatible = "st,stm32h7-spdifrx";
         pinctrl-names = "default", "sleep";
        pinctrl-0 = <&spdifrx_pins_a>;
        pinctrl-1 = <&spdifrx_sleep_pins_a>;

        spdifrx_port: port@0 {
                spdifrx_endpoint: endpoint {
                        remote-endpoint = <&spdif_in_endpoint>;
                };
        };

&dfsdm {
        compatible = "st,stm32mp1-dfsdm";
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&dfsdm_clkout_pins_a
                     &dfsdm_data1_pins_a &dfsdm_data3_pins_a>;
        pinctrl-1 = <&dfsdm_clkout_sleep_pins_a
                     &dfsdm_data1_sleep_pins_a &dfsdm_data3_sleep_pins_a>;
        spi-max-frequency = <2048000>;

        clocks = <&rcc DFSDM_K>, <&rcc ADFSDM_K>;
        clock-names = "dfsdm", "audio";

        dfsdm0: filter@0 {
                compatible = "st,stm32-dfsdm-dmic";
                st,adc-channels = <3>;                           /* Use channel 3 fed by
mic U1 signal wired to input 3 */
                st,adc-channel-names = "dmic_u1";          /* Free name used to reference
```

```
associated mic U1 */
                st,adc-channel-types = "SPI_R";         /* mic U1 signal available on
input 3 Rising edge */
                st,adc-channel-clk-src = "CLKOUT";         /* CKOUT clocks the
microphones */
                st,filter-order = <3>;

                asoc_pdm0: dfsdm-dai {
                        compatible = "st,stm32h7-dfsdm-dai";
                        #sound-dai-cells = <0>;
                        io-channels = <&dfsdm0 0>;
                        cpu_port0: port {
                                dfsdm_endpoint0: endpoint {
                                        remote-endpoint = <&dmic0_endpoint>;
                                };
                        };
                };
        };

        dfsdm1: filter@1 {
                compatible = "st,stm32-dfsdm-dmic";
                st,adc-channels = <0>;                              /* Use channel 0 fed by
mic U2 signal wired to input 1 */
                st,adc-alt-channel = <1>;                     /* Connect channel 0 to next
input (input 1) */
                st,adc-channel-names = "dmic_u2";          /* Free name used to reference
associated mic U2 */
                st,adc-channel-types = "SPI_F";         /* mic U2 signal available on
input 1 Falling edge */
                st,adc-channel-clk-src = "CLKOUT";
                st,filter-order = <3>;

                asoc_pdm1: dfsdm-dai {
                        compatible = "st,stm32h7-dfsdm-dai";
                        #sound-dai-cells = <0>;
                        io-channels = <&dfsdm1 0>;
                        cpu_port1: port {
                                dfsdm_endpoint1: endpoint {
                                        remote-endpoint = <&dmic1_endpoint>;
                                };
                        };
                };
        };

        dfsdm2: filter@2 {
                compatible = "st,stm32-dfsdm-dmic";
                st,adc-channels = <2>;                              /* Use channel 2 fed by
mic U3 signal wired to input 3 */
                st,adc-alt-channel = <1>;                     /* Connect channel 2 to next
input (input 3) */
                st,adc-channel-names = "dmic_u3";          /* Free name used to reference
associated Dmic U3 */
                st,adc-channel-types = "SPI_F";         /* mic U3 signal available on
input 3 Falling edge */
                st,adc-channel-clk-src = "CLKOUT";
                st,filter-order = <3>;

                asoc_pdm2: dfsdm-dai {
                        compatible = "st,stm32h7-dfsdm-dai";
                        #sound-dai-cells = <0>;
                        io-channels = <&dfsdm2 0>;
                        cpu_port2: port {
                                dfsdm_endpoint2: endpoint {
                                        remote-endpoint = <&dmic2_endpoint>;
                                };
                        };
                };
```

```
        };

        dfsdm3: filter@3 {
                compatible = "st,stm32-dfsdm-dmic";
                st,adc-channels = <1>;                           /* Use channel 1 fed by
mic U4 signal wired to input 1 */
                st,adc-channel-names = "dmic_u4";          /* Free name used to reference
associated mic U4 */
                st,adc-channel-types = "SPI_R";          /* mic U4 signal available on
input 1 Rising edge */
                st,adc-channel-clk-src = "CLKOUT";
                st,filter-order = <3>;

                asoc_pdm3: dfsdm-dai {
                        compatible = "st,stm32h7-dfsdm-dai";
                        #sound-dai-cells = <0>;
                        io-channels = <&dfsdm3 0>;
                        cpu_port3: port {
                                dfsdm_endpoint3: endpoint {
                                        remote-endpoint = <&dmic3_endpoint>;
                                };
                        };
                };
        };
};
```

The card-specific alsa-lib configuration file for STMP32MP15 Evaluation board is /usr/share/alsa/cards/STM32MP15EV.conf.

2.3 Dynamic configuration

The table below gives an overview of the controls allowing configuration of the STM32MPU evaluation board "sound" sound card.

audio device	CPU DAI	codec
playback_codec	no controls available	configure codec output path
record_codec	no controls available	configure codec input path
playback_spdif	configure iec958	no controls available
record_spdif	configure SPDFIRX input path	no controls available

2.3.1 Wolfson wm8994 output configuration

Control commands to configure aif1 interface to headset output (HPOUT1L/R) path, on wm8994 codec:

```
amixer -c STM32MP15EV cset name='AIF1DAC1 Volume' '96' '96'
amixer -c STM32MP15EV cset name='Headphone Volume' '63' '63'
amixer -c STM32MP15EV cset name='DAC1 Volume' '50' '50'
amixer -c STM32MP15EV cset name='DAC1L Mixer AIF1.1 Switch' 'on'
amixer -c STM32MP15EV cset name='DAC1R Mixer AIF1.1 Switch' 'on'
amixer -c STM32MP15EV cset name='DAC1 Switch' 'on' 'on'
amixer -c STM32MP15EV cset name='Left Output Mixer DAC Switch' 'on'
amixer -c STM32MP15EV cset name='Right Output Mixer DAC Switch' 'on'
amixer -c STM32MP15EV cset name='Headphone Switch' 'on' 'on'
```

Control commands to configure aif1 interface to speaker output (SPKOUTL/RP) path, on wm8994 codec:

```
amixer -c STM32MP15EV cset name='AIF1DAC1 Volume' '96' '96'
amixer -c STM32MP15EV cset name='DAC1L Mixer AIF1.1 Switch' 'on'
amixer -c STM32MP15EV cset name='DAC1R Mixer AIF1.1 Switch' 'on'
amixer -c STM32MP15EV cset name='DAC1 Switch' 'on','on'
amixer -c STM32MP15EV cset name='DAC1 Volume' '96','96'
amixer -c STM32MP15EV cset name='SPKL DAC1 Volume' '50' '50'
amixer -c STM32MP15EV cset name='SPKR DAC1 Volume' '50' '50'
amixer -c STM32MP15EV cset name='SPKL DAC1 Switch' 'on'
amixer -c STM32MP15EV cset name='SPKR DAC1 Switch' 'on'
amixer -c STM32MP15EV cset name='SPKL Output Switch' 'on'
amixer -c STM32MP15EV cset name='SPKR Output Switch' 'on'
amixer -c STM32MP15EV cset name='Speaker Mode' 'Class AB'
amixer -c STM32MP15EV cset name='Speaker Volume' '50' '50'
amixer -c STM32MP15EV cset name='Speaker Mixer Volume' 3
amixer -c STM32MP15EV cset name='Speaker Reference' 0
amixer -c STM32MP15EV cset name='Speaker Switch' 'on'
```

2.3.2 Wolfson wm8994 input configuration

Control commands to configure headset microphone input (IN1LN) to aif2 interface, on wm8994 codec:

```
amixer -c STM32MP15EV cset name='IN1L PGA IN1LN Switch' 'on'
amixer -c STM32MP15EV cset name='IN1L PGA IN1LP Switch' 'off'
amixer -c STM32MP15EV cset name='IN1L Volume' '25'
amixer -c STM32MP15EV cset name='IN1L Switch' 'on'
amixer -c STM32MP15EV cset name='MIXINL IN1L Switch' 'on'
amixer -c STM32MP15EV cset name='MIXINL IN1L Volume' '1'
amixer -c STM32MP15EV cset name='MIXINL IN1LP Volume' '0'
amixer -c STM32MP15EV cset name='AIF1ADCL Source' 'Left'
amixer -c STM32MP15EV cset name='ADCL Mux' 'ADC'
amixer -c STM32MP15EV cset name='DAC2 Left Sidetone Volume' '12'
amixer -c STM32MP15EV cset name='DAC2 Right Sidetone Volume' '12'
amixer -c STM32MP15EV cset name='AIF2DAC2L Mixer Left Sidetone Switch' 'on'
amixer -c STM32MP15EV cset name='AIF2DAC2R Mixer Right Sidetone Switch' 'on'
amixer -c STM32MP15EV cset name='DAC2 Volume' '96' '96'
amixer -c STM32MP15EV cset name='DAC2 Switch' 'on' 'on'
amixer -c STM32MP15EV cset name='AIF2ADC Volume' '96' '96'
amixer -c STM32MP15EV cset name='AIF2ADC Mux' 'AIF2ADCDAT'
amixer -c STM32MP15EV cset name='AIF2 Boost Volume' '1'
amixer -c STM32MP15EV cset name='ADC OSR' 'Low Power'
```

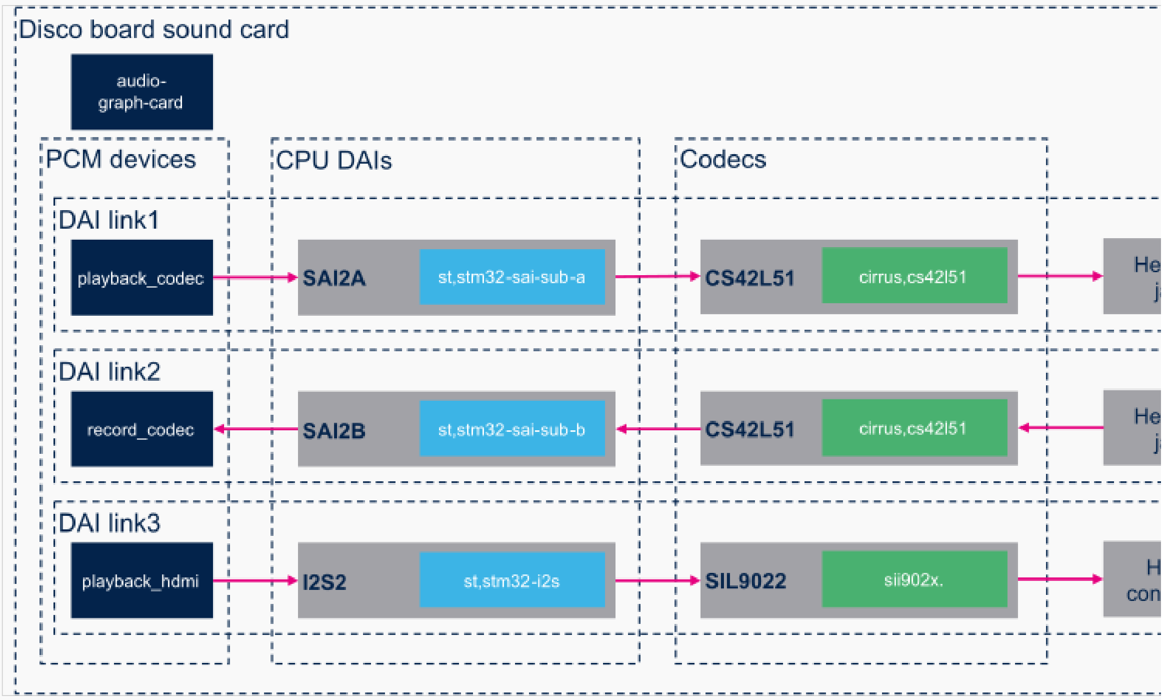
2.3.3 SPDFIRX input configuration

Control commands to configure rx1 input path on SPDFIRX:

```
amixer -c STM32MP15EV cset name='SPDIFRX input' 1
```

3 STM32MP15 disco board sound card configuration

3.1 Sound card overview



STM32MP15 Disco board sound card

3.2 Static configuration

The extract below is from the STM32MP15 disco board device tree. Only the nodes associated to the sound card, and the most relevant properties are shown here. As an example, the properties linking nodes to form the first DAI link are emphasized with green font.

```
/ {
/ * SOUNDCARD */
        sound {
                compatible = "audio-graph-card";
                label = "STM32MP15-DK";                      /* Sound card identified
as STM32MP15DK in ALSA */
                routing =
                        "Playback" , "MCLK",
                        "Capture" , "MCLK",
                        "MICL" , "Mic Bias";
                dais = <&sai2a_port &sai2b_port &i2s2_port>;
                status = "okay";
        };
};

 / * CODECS */
&i2c1 {
        cs42l51: cs42l51@4a {
                compatible = "cirrus,cs42l51";
```

```
                ...

                clocks = <&sai2a>;
                clock-names = "MCLK";

                cs42l51_port: port {
                        #address-cells = <1>;
                        #size-cells = <0>;

                        cs42l51_tx_endpoint: endpoint@0 {
                                reg = <0>;
                                remote-endpoint = <&sai2a_endpoint;
                                frame-master;                           /* codec is
master */
                                bitclock-master;
                        };

                        cs42l51_rx_endpoint: endpoint@1 {
                                reg = <1>;
                                remote-endpoint = <&sai2b_endpoint>;
                                frame-master;                           /* codec is
master */
                                bitclock-master;
                        };
                };
        };

        hdmi-transmitter@39 {
                compatible = "sil,sii9022";
                ...

                ports {
                        #address-cells = <1>;
                        #size-cells = <0>;

                        port@0 {
                                reg = <0>;
                                sii9022_in: endpoint {
                                        remote-endpoint = <&ltdc_ep0_out>;
                                };
                        };

                        port@1 {
                                reg = <1>;
                                sii9022_tx_endpoint: endpoint {
                                        remote-endpoint = <&i2s2_endpoint>;
                                };
                        };
                };
        };
};


/* CPU DAIS */
&sai2 {
        clocks = <&rcc SAI2>, <&rcc PLL3_Q>, <&rcc PLL3_Q>;
        clock-names = "pclk", "x8k", "x11k";
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&sai2a_pins_a>, <&sai2b_pins_b>;
        pinctrl-1 = <&sai2a_sleep_pins_a>, <&sai2b_sleep_pins_b>;
        status = "okay";

        sai2a: audio-controller@4400b004 {
                compatible = "st,stm32-sai-sub-a";
                #clock-cells = <0>;
                dma-names = "tx";                               /* SAI set as
```

```
transmitter */
                clocks = <&rcc SAI2_K>;
                clock-names = "sai_ck";

                sai2a_port: port {
                        sai2a_endpoint: endpoint {
                                remote-endpoint = <&cs42l51_tx_endpoint>;
                                format = "i2s";
                                mclk-fs = <256>;
                                dai-tdm-slot-num = <2>;
                                dai-tdm-slot-width = <32>;
                        };
                };
        };

        sai2b: audio-controller@4400b024 {
                dma-names = "rx";                                   /* SAI set as receiver */
                st,sync = <&sai2a 2>;                           /* SAI2B is slave of SAI2A */
                clocks = <&rcc SAI2_K>, <&sai2a>;
                clock-names = "sai_ck", "MCLK";

                sai2b_port: port {
                        sai2b_endpoint: endpoint {
                                remote-endpoint = <&cs42l51_rx_endpoint>;
                                format = "i2s";
                                mclk-fs = <256>;
                                dai-tdm-slot-num = <2>;
                                dai-tdm-slot-width = <32>;
                        };
                };
        };
};

&i2s2 {
        clocks = <&rcc SPI2>, <&rcc SPI2_K>, <&rcc PLL3_Q>, <&rcc PLL4_Q>;
        clock-names = "pclk", "i2sclk", "x8k", "x11k";
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&i2s2_pins_a>;
        pinctrl-1 = <&i2s2_pins_sleep_a>;
        status = "okay";

        i2s2_port: port {
                i2s2_endpoint: endpoint {
                        remote-endpoint = <&sii9022_tx_endpoint>;
                        format = "i2s";
                        mclk-fs = <256>;
                };
        };
};
```

The card-specific alsa-lib configuration file for STMP32MP15 Disco board is /usr/share/alsa/cards/STM32MP15DK.conf.

3.3 Dynamic configuration

The table below gives an overview of the controls allowing the configuration of the STM32MPU disco board sound card.

audio device	CPU DAI	codec
playback_codec	no controls available	configure codec output path
	no controls	

audio device	CPU DAI	codec
record_codec	available	configure codec input path
playback_hdmi	no controls available	no controls available

3.3.1 Cirrus cs42l51 output configuration

Control commands to configure the aif interface to headset output (AOUTA/B) path, on the cs42l51 codec:

```
amixer -c STM32MP15DK cset name='PCM Playback Switch' 'on','on'
amixer -c STM32MP15DK cset name='PCM Playback Volume' '63','63'
amixer -c STM32MP15DK cset name='Analog Playback Volume' '204','204'
amixer -c STM32MP15DK cset name='PCM channel mixer' 'L R'
```

3.3.2 Cirrus cs42l51 input configuration

Control commands to configure headset microphone input (MICIN1/AIN3A) to the aif interface, on the cs42l51 codec:

```
amixer -c STM32MP15DK cset name='PGA-ADC Mux Left' '3'
amixer -c STM32MP15DK cset name='Mic Boost Volume' '1','1'
```

4 References

cite_references_link_many cite_references_link_one cite_references_link_many cite_references_link_one cite_references_link_one

Linux® is a registered trademark of Linus Torvalds.

Operating System

Advanced Linux sound architecture

ALSA System on Chip

Digital Audio Interface

Microprocessor Unit

Central processing unit

Serial Audio Interface (Mechanism used to transfer non-buffered audio data between processors and/or audio converters.)

Sony/Philips Digital Interface Format (Protocol (IEC-60958))

Digital Filter for Sigma-Delta Modulator

bs-flaggedrevsconnector-addstabledatetochapterheadlinesmodifier-laststable-tag-text: 05.01.2021 - 15:38 / bs-flaggedrevsconnector-addstabledatetochapterheadlinesmodifier-stablerevisiondate-tag-text: 04.01.2021 - 17:18

Serial Peripheral Interface

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

bs-flaggedrevsconnector-addstabledatetochapterheadlinesmodifier-laststable-tag-text: 02.06.2020 - 14:38 / bs-flaggedrevsconnector-addstabledatetochapterheadlinesmodifier-stablerevisiondate-tag-text: 27.05.2020 - 15:56

revreview-invalid

returnto

bs-flaggedrevsconnector-addstabledatetochapterheadlinesmodifier-laststable-tag-text: 17.06.2020 - 15:26 / bs-flaggedrevsconnector-addstabledatetochapterheadlinesmodifier-stablerevisiondate-tag-text: 16.01.2020 - 09:30

revreview-invalid

returnto

revreview-invalid

returnto

subcategories

category-subcat-count

S

- categorytree-expand-bullet STM32MP15 Discovery kits categorytree-member-num
- categorytree-expand-bullet STM32MP15 Evaluation boards categorytree-member-num

category_header

category-article-count

L

- LEDs and buttons on STM32 MPU boards

S

- STM32MP1 Developer Package
- STM32MP1 Developer Package for Android
- STM32MP1 Distribution Package
- STM32MP1 Distribution Package for Android

bs-flaggedrevsconnector-addstabledatetochapterheadlinesmodifier-laststable-tag-text: 26.03.2021 - 15:13 / bs-flaggedrevsconnector-addstabledatetochapterheadlinesmodifier-stablerevisiondate-tag-text: 09.03.2021 - 13:42

bs-flaggedrevsconnector-addstabledatetochapterheadlinesmodifier-laststable-tag-text: 03.03.2021 - 17:23 / bs-flaggedrevsconnector-addstabledatetochapterheadlinesmodifier-stablerevisiondate-tag-text: 24.02.2021 - 14:14

revreview-invalid

returnto

revreview-invalid

bs-flaggedrevsconnector-addstabledatetochapterheadlinesmodifier-laststable-tag-text: 03.03.2021 - 17:28 / bs-flaggedrevsconnector-addstabledatetochapterheadlinesmodifier-stablerevisiondate-tag-text: 25.01.2021 - 08:45

returnto

revreview-invalid

bs-flaggedrevsconnector-addstabledatetochapterheadlinesmodifier-laststable-tag-text: 23.09.2020 - 13:22 / bs-flaggedrevsconnector-addstabledatetochapterheadlinesmodifier-stablerevisiondate-tag-text: 12.06.2020 - 13:25

returnto

revreview-invalid

returnto

bs-flaggedrevsconnector-addstabledatetochapterheadlinesmodifier-laststable-tag-text: 03.03.2021 - 17:27 / bs-flaggedrevsconnector-addstabledatetochapterheadlinesmodifier-stablerevisiondate-tag-text: 24.02.2021 - 13:29

toc

1 Overview

This article explains how to configure ST audio peripherals, as well as STM32MP1 boards external audio components, when they are assigned to the **Linux® OS**. In such cases, they are controlled by the ALSA framework.

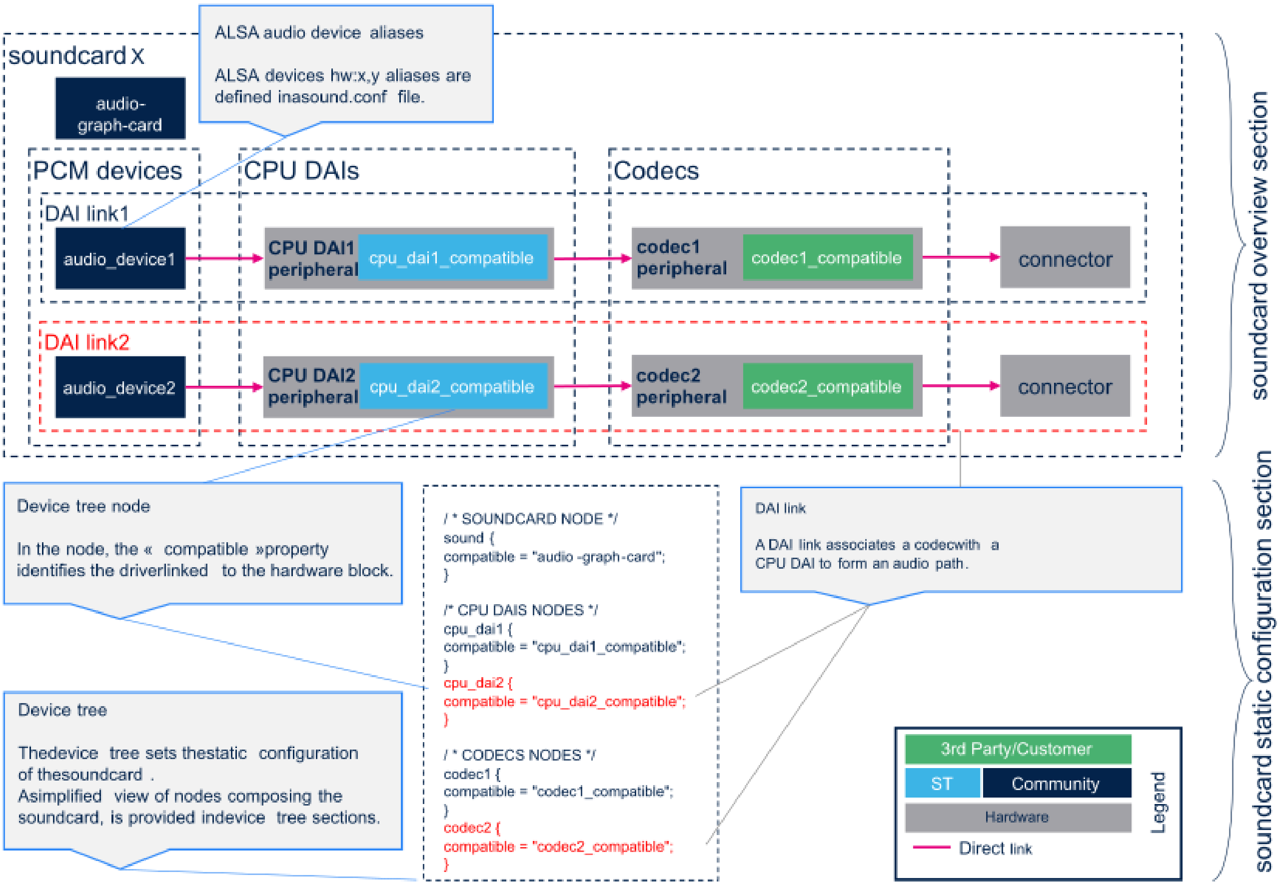
In the ASoC layer of the ALSA framework, audio hardware components are described as CPU DAIs and codec, which are linked together to create DAI links. A sound card is a software component gathering a set of DAI links.

Each of the following STM32 MPU board sections describes one or more sound cards. A schematic for each sound card is provided, as well as its means of static and dynamic configuration.

1.1 Sound card schematic

The sound card schematic gives an overview of the hardware and software components forming the sound card, and their relationships.

The example sound card schematic given below emphasizes the links between the sound card and the device-tree section.



1.2 Static configuration

- Device tree

The device tree allows the description, configuration and connection of the audio hardware components to define the sound card. The user has to follow the audio graph card bindingscite_reference_link to configure the sound card and device graph bindingscite_reference_link to connect audio components. The user must also refer to the audio component (codec and CPU DAI) bindings to configure these components properly. The bindings of the audio components can be found in device-tree samples in following sections and in the References chapter.

The STMicroelectronics configuration tool STM32CubeMX, allows the generation of CPU DAI device tree nodes.

STM32CubeMX does not allow configuration of sound card and codec nodes, which are board dependent. The sound card node and the codec nodes have to be filled manually through user sections.

- asound.conf cite_reference_link

The optional asound.confcite_reference_link system-global custom settings file, provides extra functionalities, such as routing and audio sample conversion. It can be found in the /etc directory.

- Sound card configuration files cite_reference_link

The alsa-lib layer provides card configuration files in /usr/share/alsa/cards directory. These files allow to map ALSA hardware devices on standard devices, such as "front", "hdmi" or "iec958" devices. The label defined in sound card device tree node defines the name of the card. The card configuration is retrieved from this card name, according to /usr/share/alsa/cards/aliases.conf mapping.

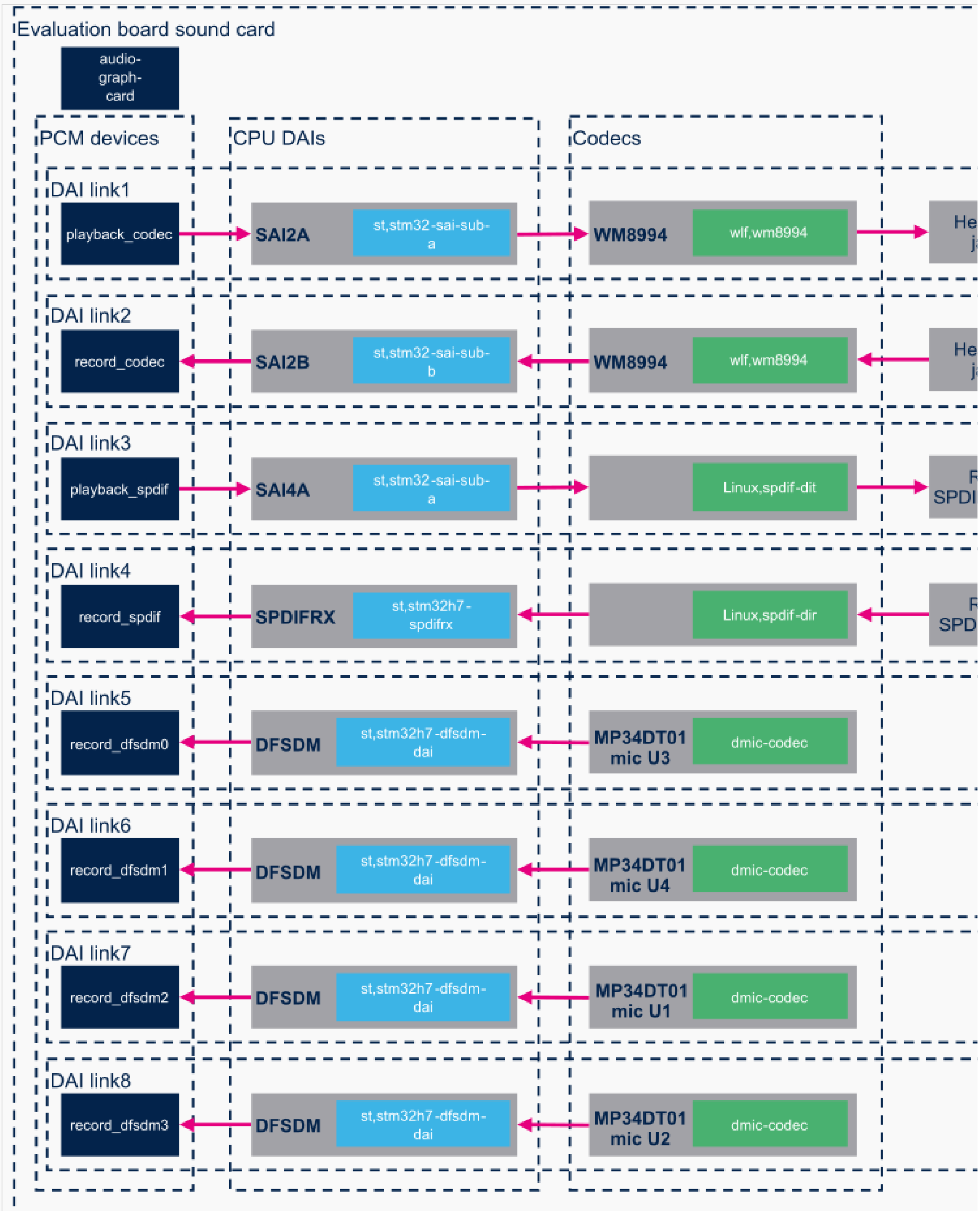
1.3 Dynamic configuration

The codecs and CPU DAI drivers also provide ALSA controls, allowing dynamic configuration of the sound card. The controls can be changed at runtime through the amixer utility to modify certain settings in the audio path. For instance, such controls can be used to modify the audio volume or the mute state of a block in the codec.

A customized configuration of these controls can be saved in the asound.state configuration file using the alsactl utility. This configuration can be restored at boot time through alsactl. STM32MPU sound cards come with a dedicated asound.state configuration file providing relevant control settings.

2 STM32MP15 evaluation board sound card configuration

2.1 Sound card overview



STM32MP15 Evaluation board sound card

2.2 Static configuration

The extract below is from the STM32MP15 evaluation board device tree. Only the nodes associated to the sound card, and the most relevant properties are shown here. For example, the properties linking nodes to form the first DAI link are emphasized with green font.

```
/ * SOUNDCARD */
        sound {
                compatible = "audio-graph-cardcite_reference_link";
                label = "STM32MP15-EV";                        /* Sound card identified
as STM32MP15EV in ALSA */
                routing =
                        "AIF1CLK" , "MCLK1",
                        "AIF2CLK" , "MCLK1",
                        "IN1LN" , "MICBIAS2",
                        "DMIC2DAT" , "MICBIAS1",
                        "DMIC1DAT" , "MICBIAS1";
                dais = <&sai2a_port &sai2b_port &sai4a_port &spdifrx_port
                        &dfsdm0_port &dfsdm1_port &dfsdm2_port &dfsdm3_port>;
        };

/ * CODECS */
        spdif_out: spdif-out {
                compatible = "linux,spdif-ditcite_reference_link";

                spdif_out_port: port@0 {
                        spdif_out_endpoint: endpoint {
                                remote-endpoint = <&sai4a_endpoint>;
                        };
                };
        };

        spdif_in: spdif-in {
                compatible = "linux,spdif-dircite_reference_link";
                spdif_in_port: port@0 {
                        spdif_in_endpoint: endpoint {
                                remote-endpoint = <&spdifrx_endpoint>;
                        };
                };
        };

        dmic0: dmic@0 {
                compatible = "dmic-codec";
                port {
                        dmic0_endpoint: endpoint {
                                remote-endpoint = <&dfsdm_endpoint0>;
                        };
                };
        };

        dmic1: dmic@1 {
                compatible = "dmic-codec";
                port {
                        dmic1_endpoint: endpoint {
                                remote-endpoint = <&dfsdm_endpoint1>;
                        };
                };
        };

        dmic2: dmic@2 {
                compatible = "dmic-codec";
                port {
                        dmic2_endpoint: endpoint {
                                remote-endpoint = <&dfsdm_endpoint2>;
```

```
                        };
                };
        };

        dmic3: dmic@3 {
                compatible = "dmic-codec";
                port {
                        dmic3_endpoint: endpoint {
                                remote-endpoint = <&dfsdm_endpoint3>;
                        };
                };
        };
};

&i2c2 {
        wm8994: wm8994@1b {
                compatible = "wlf,wm8994";
                ...
                clocks = <&sai2a>;
                clock-names = "MCLK1";

                ports {
                        #address-cells = <1>;
                        #size-cells = <0>;

                        wm8994_tx_port: port@0 {
                                wm8994_tx_endpoint: endpoint {
                                        remote-endpoint = <&sai2a_endpoint>;
                                };
                        };

                        wm8994_rx_port: port@1 {
                                wm8994_rx_endpoint: endpoint {
                                        remote-endpoint = <&sai2b_endpoint>;
                                };
                        };
                };
        };
};

/* CPU DAIS */
&sai2 {
        clocks = <&rcc SAI2>, <&rcc PLL3_Q>, <&rcc PLL4_Q>;
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&sai2a_pins_a>, <&sai2b_pins_a>;
        pinctrl-1 = <&sai2a_sleep_pins_a>, <&sai2b_sleep_pins_a>;
        clock-names = "pclk", "x8k", "x11k";

        sai2a: audio-controller@4400b004 {
                compatible = "st,stm32-sai-sub-a";
                dma-names = "tx";                               /* SAI set as
transmitter */
                clocks = <&rcc SAI2_K>;
                clock-names = "sai_ck";

                sai2a_port: port@0 {
                        sai2a_endpoint: endpoint {
                                remote-endpoint = <&wm8994_tx_endpoint>;
                                format = "i2s";
                                mclk-fs = <256>;                /* SAI is master clock
provider */
                        };
                };
        };

        sai2b: audio-controller@4400b024 {
                compatible = "st,stm32-sai-sub-b";
```

```
                dma-names = "rx";                                   /* SAI set as receiver */
                clocks = <&rcc SAI2_K>, <&sai2a>;
                clock-names = "sai_ck", "MCLK";

                sai2b_port: port@0 {
                        sai2b_endpoint: endpoint {
                                remote-endpoint = <&wm8994_rx_endpoint>;
                                format = "i2s";
                                mclk-fs = <256>;                /* SAI is master clock
provider */
                        };
                };
        };
};

&sai4 {
        clocks = <&rcc SAI4>, <&rcc PLL3_Q>, <&rcc PLL4_Q>;
        clock-names = "pclk", "x8k", "x11k";

        sai4a: audio-controller@50027004 {
                compatible = "st,stm32-sai-sub-a";
                dma-names = "tx";
                st,iec60958;                                        /* SAI configured for
S/PDIF protocol*/
                pinctrl-names = "default", "sleep";
                pinctrl-0 = <&sai4a_pins_a>;
                pinctrl-1 = <&sai4a_sleep_pins_a>;
                 clocks = <&rcc SAI4_K>;
                clock-names = "sai_ck";

                sai4a_port: port@0 {
                        sai4a_endpoint: endpoint {
                                remote-endpoint = <&spdif_out_endpoint>;
                        };
                };
        };
};

&spdifrx {
         compatible = "st,stm32h7-spdifrx";
         pinctrl-names = "default", "sleep";
        pinctrl-0 = <&spdifrx_pins_a>;
        pinctrl-1 = <&spdifrx_sleep_pins_a>;

        spdifrx_port: port@0 {
                spdifrx_endpoint: endpoint {
                        remote-endpoint = <&spdif_in_endpoint>;
                };
        };

&dfsdm {
        compatible = "st,stm32mp1-dfsdm";
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&dfsdm_clkout_pins_a
                     &dfsdm_data1_pins_a &dfsdm_data3_pins_a>;
        pinctrl-1 = <&dfsdm_clkout_sleep_pins_a
                     &dfsdm_data1_sleep_pins_a &dfsdm_data3_sleep_pins_a>;
        spi-max-frequency = <2048000>;

        clocks = <&rcc DFSDM_K>, <&rcc ADFSDM_K>;
        clock-names = "dfsdm", "audio";

        dfsdm0: filter@0 {
                compatible = "st,stm32-dfsdm-dmic";
                st,adc-channels = <3>;                              /* Use channel 3 fed by
mic U1 signal wired to input 3 */
                st,adc-channel-names = "dmic_u1";          /* Free name used to reference
```

```
associated mic U1 */
                st,adc-channel-types = "SPI_R";         /* mic U1 signal available on
input 3 Rising edge */
                st,adc-channel-clk-src = "CLKOUT";         /* CKOUT clocks the
microphones */
                st,filter-order = <3>;

                asoc_pdm0: dfsdm-dai {
                        compatible = "st,stm32h7-dfsdm-dai";
                        #sound-dai-cells = <0>;
                        io-channels = <&dfsdm0 0>;
                        cpu_port0: port {
                                dfsdm_endpoint0: endpoint {
                                        remote-endpoint = <&dmic0_endpoint>;
                                };
                        };
                };
        };

        dfsdm1: filter@1 {
                compatible = "st,stm32-dfsdm-dmic";
                st,adc-channels = <0>;                              /* Use channel 0 fed by
mic U2 signal wired to input 1 */
                st,adc-alt-channel = <1>;                  /* Connect channel 0 to next
input (input 1) */
                st,adc-channel-names = "dmic_u2";          /* Free name used to reference
associated mic U2 */
                st,adc-channel-types = "SPI_F";         /* mic U2 signal available on
input 1 Falling edge */
                st,adc-channel-clk-src = "CLKOUT";
                st,filter-order = <3>;

                asoc_pdm1: dfsdm-dai {
                        compatible = "st,stm32h7-dfsdm-dai";
                        #sound-dai-cells = <0>;
                        io-channels = <&dfsdm1 0>;
                        cpu_port1: port {
                                dfsdm_endpoint1: endpoint {
                                        remote-endpoint = <&dmic1_endpoint>;
                                };
                        };
                };
        };

        dfsdm2: filter@2 {
                compatible = "st,stm32-dfsdm-dmic";
                st,adc-channels = <2>;                              /* Use channel 2 fed by
mic U3 signal wired to input 3 */
                st,adc-alt-channel = <1>;                  /* Connect channel 2 to next
input (input 3) */
                st,adc-channel-names = "dmic_u3";          /* Free name used to reference
associated Dmic U3 */
                st,adc-channel-types = "SPI_F";         /* mic U3 signal available on
input 3 Falling edge */
                st,adc-channel-clk-src = "CLKOUT";
                st,filter-order = <3>;

                asoc_pdm2: dfsdm-dai {
                        compatible = "st,stm32h7-dfsdm-dai";
                        #sound-dai-cells = <0>;
                        io-channels = <&dfsdm2 0>;
                        cpu_port2: port {
                                dfsdm_endpoint2: endpoint {
                                        remote-endpoint = <&dmic2_endpoint>;
                                };
                        };
                };
```

```
        };

        dfsdm3: filter@3 {
                compatible = "st,stm32-dfsdm-dmic";
                st,adc-channels = <1>;                           /* Use channel 1 fed by
mic U4 signal wired to input 1 */
                st,adc-channel-names = "dmic_u4";         /* Free name used to reference
associated mic U4 */
                st,adc-channel-types = "SPI_R";         /* mic U4 signal available on
input 1 Rising edge */
                st,adc-channel-clk-src = "CLKOUT";
                st,filter-order = <3>;

                asoc_pdm3: dfsdm-dai {
                        compatible = "st,stm32h7-dfsdm-dai";
                        #sound-dai-cells = <0>;
                        io-channels = <&dfsdm3 0>;
                        cpu_port3: port {
                                dfsdm_endpoint3: endpoint {
                                        remote-endpoint = <&dmic3_endpoint>;
                                };
                        };
                };
        };
};
```

The card-specific alsa-lib configuration file for STMP32MP15 Evaluation board is /usr/share/alsa/cards/STM32MP15EV.conf.

2.3 Dynamic configuration

The table below gives an overview of the controls allowing configuration of the STM32MPU evaluation board "sound" sound card.

audio device	CPU DAI	codec
playback_codec	no controls available	configure codec output path
record_codec	no controls available	configure codec input path
playback_spdif	configure iec958	no controls available
record_spdif	configure SPDFIRX input path	no controls available

2.3.1 Wolfson wm8994 output configuration

Control commands to configure aif1 interface to headset output (HPOUT1L/R) path, on wm8994 codec:

```
amixer -c STM32MP15EV cset name='AIF1DAC1 Volume' '96' '96'
amixer -c STM32MP15EV cset name='Headphone Volume' '63' '63'
amixer -c STM32MP15EV cset name='DAC1 Volume' '50' '50'
amixer -c STM32MP15EV cset name='DAC1L Mixer AIF1.1 Switch' 'on'
amixer -c STM32MP15EV cset name='DAC1R Mixer AIF1.1 Switch' 'on'
amixer -c STM32MP15EV cset name='DAC1 Switch' 'on' 'on'
amixer -c STM32MP15EV cset name='Left Output Mixer DAC Switch' 'on'
amixer -c STM32MP15EV cset name='Right Output Mixer DAC Switch' 'on'
amixer -c STM32MP15EV cset name='Headphone Switch' 'on' 'on'
```

Control commands to configure aif1 interface to speaker output (SPKOUTL/RP) path, on wm8994 codec:

```
amixer -c STM32MP15EV cset name='AIF1DAC1 Volume' '96' '96'
amixer -c STM32MP15EV cset name='DAC1L Mixer AIF1.1 Switch' 'on'
amixer -c STM32MP15EV cset name='DAC1R Mixer AIF1.1 Switch' 'on'
amixer -c STM32MP15EV cset name='DAC1 Switch' 'on','on'
amixer -c STM32MP15EV cset name='DAC1 Volume' '96','96'
amixer -c STM32MP15EV cset name='SPKL DAC1 Volume' '50' '50'
amixer -c STM32MP15EV cset name='SPKR DAC1 Volume' '50' '50'
amixer -c STM32MP15EV cset name='SPKL DAC1 Switch' 'on'
amixer -c STM32MP15EV cset name='SPKR DAC1 Switch' 'on'
amixer -c STM32MP15EV cset name='SPKL Output Switch' 'on'
amixer -c STM32MP15EV cset name='SPKR Output Switch' 'on'
amixer -c STM32MP15EV cset name='Speaker Mode' 'Class AB'
amixer -c STM32MP15EV cset name='Speaker Volume' '50' '50'
amixer -c STM32MP15EV cset name='Speaker Mixer Volume' 3
amixer -c STM32MP15EV cset name='Speaker Reference' 0
amixer -c STM32MP15EV cset name='Speaker Switch' 'on'
```

2.3.2 Wolfson wm8994 input configuration

Control commands to configure headset microphone input (IN1LN) to aif2 interface, on wm8994 codec:

```
amixer -c STM32MP15EV cset name='IN1L PGA IN1LN Switch' 'on'
amixer -c STM32MP15EV cset name='IN1L PGA IN1LP Switch' 'off'
amixer -c STM32MP15EV cset name='IN1L Volume' '25'
amixer -c STM32MP15EV cset name='IN1L Switch' 'on'
amixer -c STM32MP15EV cset name='MIXINL IN1L Switch' 'on'
amixer -c STM32MP15EV cset name='MIXINL IN1L Volume' '1'
amixer -c STM32MP15EV cset name='MIXINL IN1LP Volume' '0'
amixer -c STM32MP15EV cset name='AIF1ADCL Source' 'Left'
amixer -c STM32MP15EV cset name='ADCL Mux' 'ADC'
amixer -c STM32MP15EV cset name='DAC2 Left Sidetone Volume' '12'
amixer -c STM32MP15EV cset name='DAC2 Right Sidetone Volume' '12'
amixer -c STM32MP15EV cset name='AIF2DAC2L Mixer Left Sidetone Switch' 'on'
amixer -c STM32MP15EV cset name='AIF2DAC2R Mixer Right Sidetone Switch' 'on'
amixer -c STM32MP15EV cset name='DAC2 Volume' '96' '96'
amixer -c STM32MP15EV cset name='DAC2 Switch' 'on' 'on'
amixer -c STM32MP15EV cset name='AIF2ADC Volume' '96' '96'
amixer -c STM32MP15EV cset name='AIF2ADC Mux' 'AIF2ADCDAT'
amixer -c STM32MP15EV cset name='AIF2 Boost Volume' '1'
amixer -c STM32MP15EV cset name='ADC OSR' 'Low Power'
```

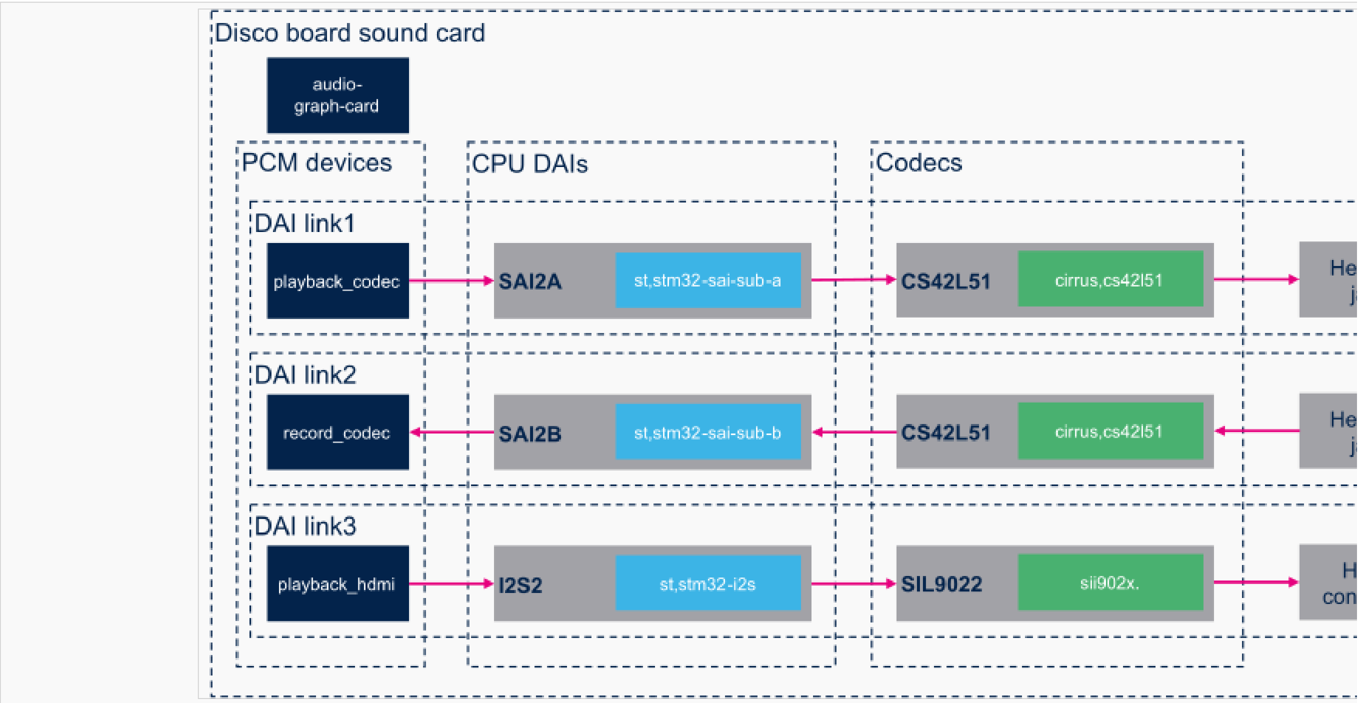
2.3.3 SPDFIRX input configuration

Control commands to configure rx1 input path on SPDFIRX:

```
amixer -c STM32MP15EV cset name='SPDIFRX input' 1
```

3 STM32MP15 disco board sound card configuration

3.1 Sound card overview



STM32MP15 Disco board sound card

3.2 Static configuration

The extract below is from the STM32MP15 disco board device tree. Only the nodes associated to the sound card, and the most relevant properties are shown here. As an example, the properties linking nodes to form the first DAI link are emphasized with green font.

```
/ {
/ * SOUNDCARD */
        sound {
                compatible = "audio-graph-card";
                label = "STM32MP15-DK";                        /* Sound card identified
as STM32MP15DK in ALSA */
                routing =
                        "Playback" , "MCLK",
                        "Capture" , "MCLK",
                        "MICL" , "Mic Bias";
                dais = <&sai2a_port &sai2b_port &i2s2_port>;
                status = "okay";
        };
};

 / * CODECS */
&i2c1 {
        cs42l51: cs42l51@4a {
                compatible = "cirrus,cs42l51";
```

```
                ...

                clocks = <&sai2a>;
                clock-names = "MCLK";

                cs42l51_port: port {
                        #address-cells = <1>;
                        #size-cells = <0>;

                        cs42l51_tx_endpoint: endpoint@0 {
                                reg = <0>;
                                remote-endpoint = <&sai2a_endpoint;
                                frame-master;                        /* codec is
master */
                                bitclock-master;
                        };

                        cs42l51_rx_endpoint: endpoint@1 {
                                reg = <1>;
                                remote-endpoint = <&sai2b_endpoint>;
                                frame-master;                        /* codec is
master */
                                bitclock-master;
                        };
                };
        };

        hdmi-transmitter@39 {
                compatible = "sil,sii9022";
                ...

                ports {
                        #address-cells = <1>;
                        #size-cells = <0>;

                        port@0 {
                                reg = <0>;
                                sii9022_in: endpoint {
                                        remote-endpoint = <&ltdc_ep0_out>;
                                };
                        };

                        port@1 {
                                reg = <1>;
                                sii9022_tx_endpoint: endpoint {
                                        remote-endpoint = <&i2s2_endpoint>;
                                };
                        };
                };
        };
};


/* CPU DAIS */
&sai2 {
        clocks = <&rcc SAI2>, <&rcc PLL3_Q>, <&rcc PLL3_Q>;
        clock-names = "pclk", "x8k", "x11k";
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&sai2a_pins_a>, <&sai2b_pins_b>;
        pinctrl-1 = <&sai2a_sleep_pins_a>, <&sai2b_sleep_pins_b>;
        status = "okay";

        sai2a: audio-controller@4400b004 {
                compatible = "st,stm32-sai-sub-a";
                #clock-cells = <0>;
                dma-names = "tx";                              /* SAI set as
```

```
transmitter */
                        clocks = <&rcc SAI2_K>;
                        clock-names = "sai_ck";

                        sai2a_port: port {
                                sai2a_endpoint: endpoint {
                                        remote-endpoint = <&cs42l51_tx_endpoint>;
                                        format = "i2s";
                                        mclk-fs = <256>;
                                        dai-tdm-slot-num = <2>;
                                        dai-tdm-slot-width = <32>;
                                };
                        };
                };

                sai2b: audio-controller@4400b024 {
                        dma-names = "rx";                                /* SAI set as receiver */
                        st,sync = <&sai2a 2>;                          /* SAI2B is slave of SAI2A */
                        clocks = <&rcc SAI2_K>, <&sai2a>;
                        clock-names = "sai_ck", "MCLK";

                        sai2b_port: port {
                                sai2b_endpoint: endpoint {
                                        remote-endpoint = <&cs42l51_rx_endpoint>;
                                        format = "i2s";
                                        mclk-fs = <256>;
                                        dai-tdm-slot-num = <2>;
                                        dai-tdm-slot-width = <32>;
                                };
                        };
                };
};

&i2s2 {
        clocks = <&rcc SPI2>, <&rcc SPI2_K>, <&rcc PLL3_Q>, <&rcc PLL4_Q>;
        clock-names = "pclk", "i2sclk", "x8k", "x11k";
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&i2s2_pins_a>;
        pinctrl-1 = <&i2s2_pins_sleep_a>;
        status = "okay";

        i2s2_port: port {
                i2s2_endpoint: endpoint {
                        remote-endpoint = <&sii9022_tx_endpoint>;
                        format = "i2s";
                        mclk-fs = <256>;
                };
        };
};
```

The card-specific alsa-lib configuration file for STMP32MP15 Disco board is /usr/share/alsa/cards/STM32MP15DK.conf.

3.3 Dynamic configuration

The table below gives an overview of the controls allowing the configuration of the STM32MPU disco board sound card.

audio device	CPU DAI	codec
playback_codec	no controls available	configure codec output path
	no controls	

audio device	CPU DAI	codec
record_codec	available	configure codec input path
playback_hdmi	no controls available	no controls available

3.3.1 Cirrus cs42l51 output configuration

Control commands to configure the aif interface to headset output (AOUTA/B) path, on the cs42l51 codec:

```
amixer -c STM32MP15DK cset name='PCM Playback Switch' 'on','on'
amixer -c STM32MP15DK cset name='PCM Playback Volume' '63','63'
amixer -c STM32MP15DK cset name='Analog Playback Volume' '204','204'
amixer -c STM32MP15DK cset name='PCM channel mixer' 'L R'
```

3.3.2 Cirrus cs42l51 input configuration

Control commands to configure headset microphone input (MICIN1/AIN3A) to the aif interface, on the cs42l51 codec:

```
amixer -c STM32MP15DK cset name='PGA-ADC Mux Left' '3'
amixer -c STM32MP15DK cset name='Mic Boost Volume' '1','1'
```

4 References

cite_references_link_many cite_references_link_one cite_references_link_many cite_references_link_one cite_references_link_one

Linux® is a registered trademark of Linus Torvalds.

Operating System

Advanced Linux sound architecture

ALSA System on Chip

Digital Audio Interface

Microprocessor Unit

Central processing unit

Serial Audio Interface (Mechanism used to transfer non-buffered audio data between processors and/or audio converters.)

Sony/Philips Digital Interface Format (Protocol (IEC-60958))

Digital Filter for Sigma-Delta Modulator

Serial Peripheral Interface

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.