Creating codec to codec dai link for ALSA dapm

Mostly the flow of audio is always from CPU to codec so your system will look as below:

```
 ---------          ---------
|         |  dai   |         |
    CPU    ------->    codec
|         |        |         |
 ---------          ---------
```

In case your system looks as below:

```
                     ---------
                    |         |
                      codec-2
                    |         |
                    ---------
                         |
                       dai-2
                         |
 ----------          ---------
|          |  dai-1 |         |
    CPU     ------->  codec-1
|          |        |         |
 ----------          ---------
                         |
                       dai-3
                         |
                     ---------
                    |         |
                      codec-3
                    |         |
                     ---------
```

Suppose codec-2 is a bluetooth chip and codec-3 is connected to a speaker and you have a below scenario: codec-2 will receive the audio data and the user wants to play that audio through codec-3 without involving the CPU.This aforementioned case is the ideal case when codec to codec connection should be used.

Your dai_link should appear as below in your machine file:

```
/*
 * this pcm stream only supports 24 bit, 2 channel and
 * 48k sampling rate.
 */
static const struct snd_soc_pcm_stream dsp_codec_params = {
       .formats = SNDRV_PCM_FMTBIT_S24_LE,
       .rate_min = 48000,
       .rate_max = 48000,
       .channels_min = 2,
       .channels_max = 2,
};

{
   .name = "CPU-DSP",
   .stream_name = "CPU-DSP",
   .cpu_dai_name = "samsung-i2s.0",
   .codec_name = "codec-2,
   .codec_dai_name = "codec-2-dai_name",
   .platform_name = "samsung-i2s.0",
   .dai_fmt = SND_SOC_DAIFMT_I2S | SND_SOC_DAIFMT_NB_NF
           | SND_SOC_DAIFMT_CBM_CFM,
   .ignore_suspend = 1,
   .params = &dsp_codec_params,
},
{
   .name = "DSP-CODEC",
   .stream_name = "DSP-CODEC",
   .cpu_dai_name = "wm0010-sdi2",
   .codec_name = "codec-3,
   .codec_dai_name = "codec-3-dai_name",
   .dai_fmt = SND_SOC_DAIFMT_I2S | SND_SOC_DAIFMT_NB_NF
           | SND_SOC_DAIFMT_CBM_CFM,
   .ignore_suspend = 1,
   .params = &dsp_codec_params,
},
```

Above code snippet is motivated from sound/soc/samsung/speyside.c.

Note the "params" callback which lets the dapm know that this dai_link is a codec to codec connection.

In dapm core a route is created between cpu_dai playback widget and codec_dai capture widget for playback path and vice-versa is true for capture path. In order for this aforementioned route to get triggered, DAPM needs to find a valid endpoint which could be either a sink or source widget corresponding to playback and capture path respectively.

In order to trigger this dai_link widget, a thin codec driver for the speaker amp can be created as demonstrated in wm8727.c file, it sets appropriate constraints for the device even if it needs no control.

Make sure to name your corresponding cpu and codec playback and capture dai names ending with "Playback" and "Capture" respectively as dapm core will link and power those dais based on the name.

A dai_link in a "simple-audio-card" will automatically be detected as codec to codec when all DAIs on the link belong to codec components. The dai_link will be initialized with the subset of stream parameters (channels, format, sample rate) supported by all DAIs on the link. Since there is no way to provide these parameters in the device tree, this is mostly useful for communication with simple fixed-function codecs, such as a Bluetooth controller or cellular modem.