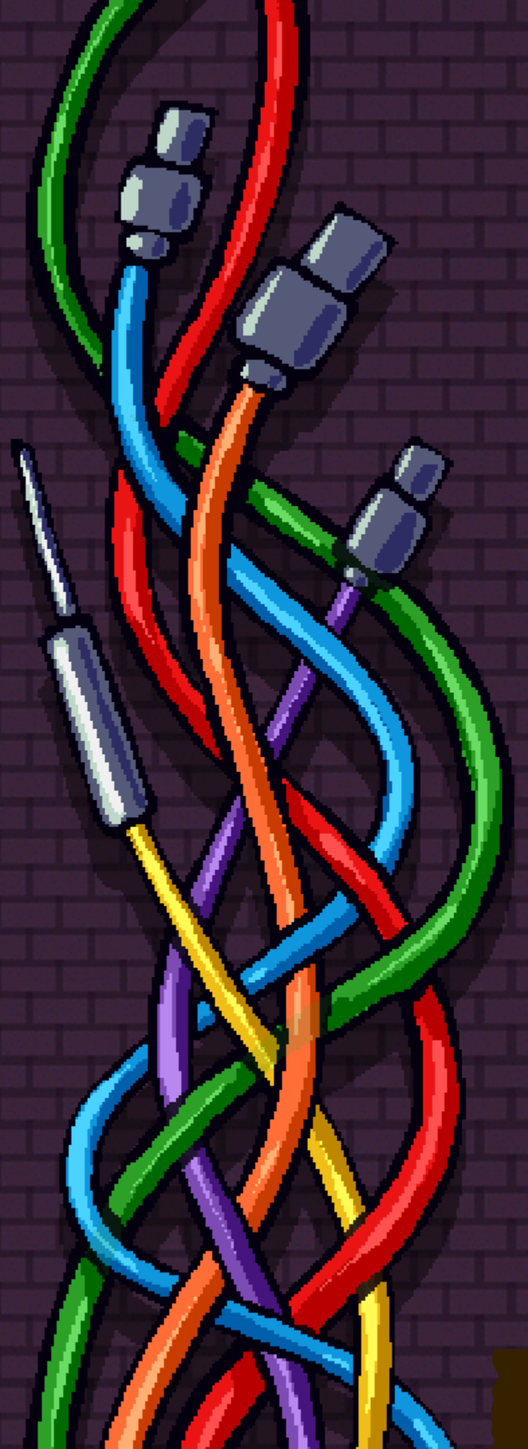


ADCx GATHER

AUDIO CODEC SWITCHING IN THE LINUX KERNEL FOR AUTOMOTIVE EDGE DEVICES

RUTVIJ TRIVEDI

rutvij.trivedi@siliconsignals.io

+91-94087 30545

About **Me**

- Co-Founder & M.D. of **Silicon Signals Pvt. Ltd. (Guj, IN)**

- Built a team which contributes to open source e.g Linux, ZephyrOS, AOSP, U-boot

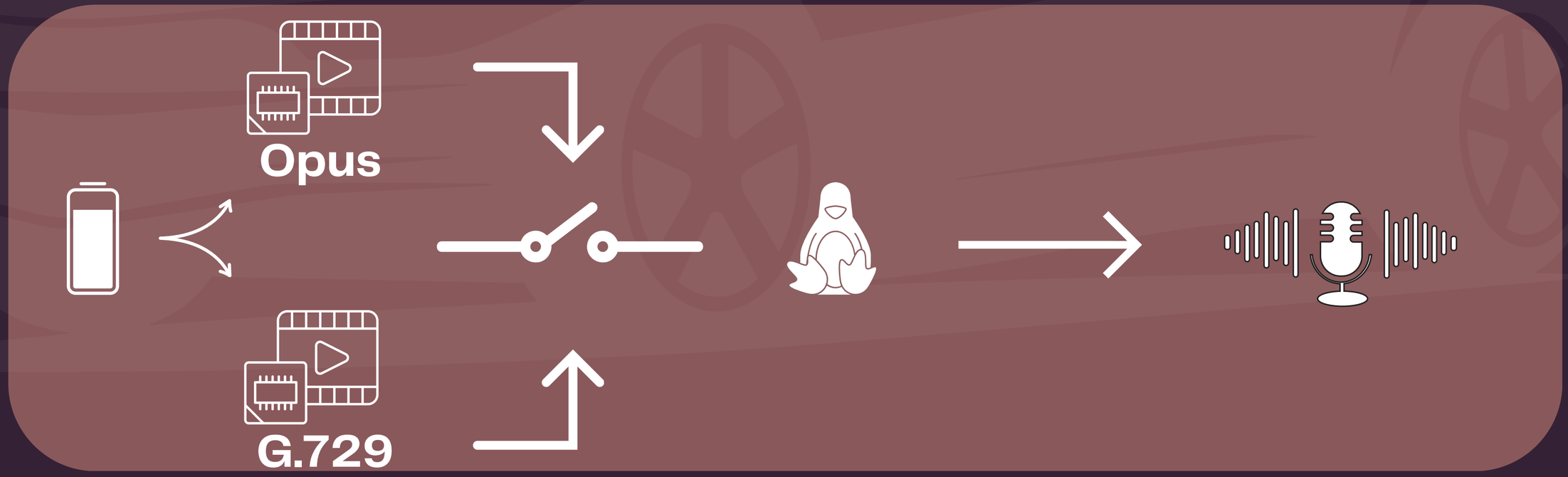
- Over decade of diverse experience in Embedded Product Engineering, Software Development, and Embedded System Development

- Consumer, Healthcare, IOT, Audio, Multimedia/Camera, Automotive and many more products

Motivation

Opus, high quality, high bitrate = IVI (infotainment)

G.729, low power, low latency = eCall (safety)



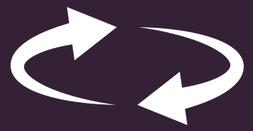
The Problem **Today**

- In Linux ALSA - ASOC

- User-space *snd_pcm_hw_params()* → ASoC DPCM FE handler → *dpcm_be_dai_hw_params()* → *snd_soc_dai_hw_params()* → codec driver registers are programmed.

- Once you call *hw_params()*, the codec is fixed. If you need a different codec, ALSA forces you to stop the stream and reopen it

- This means, ✖
  - **No runtime switching**
  - **100 ms dropout**
  - **10–15% CPU spike**



What Engineers Tried So **Far**

- **Userspace Transcoding**
  - Uses CPU for re-encoding/decoding
  - Power hungry, adds latency

- **Multiple ALSA Devices**
  - One PCM device per codec
  - No seamless runtime switching

- **Restarting PCM Stream**
  - Tear down & reopen stream with new codec
  - Causes >100 ms dropouts, unsafe for eCall

🚫 ***None of these approaches meet automotive latency, power, or safety requirements***

Proposed **Architecture**

SILICON SIGNALS

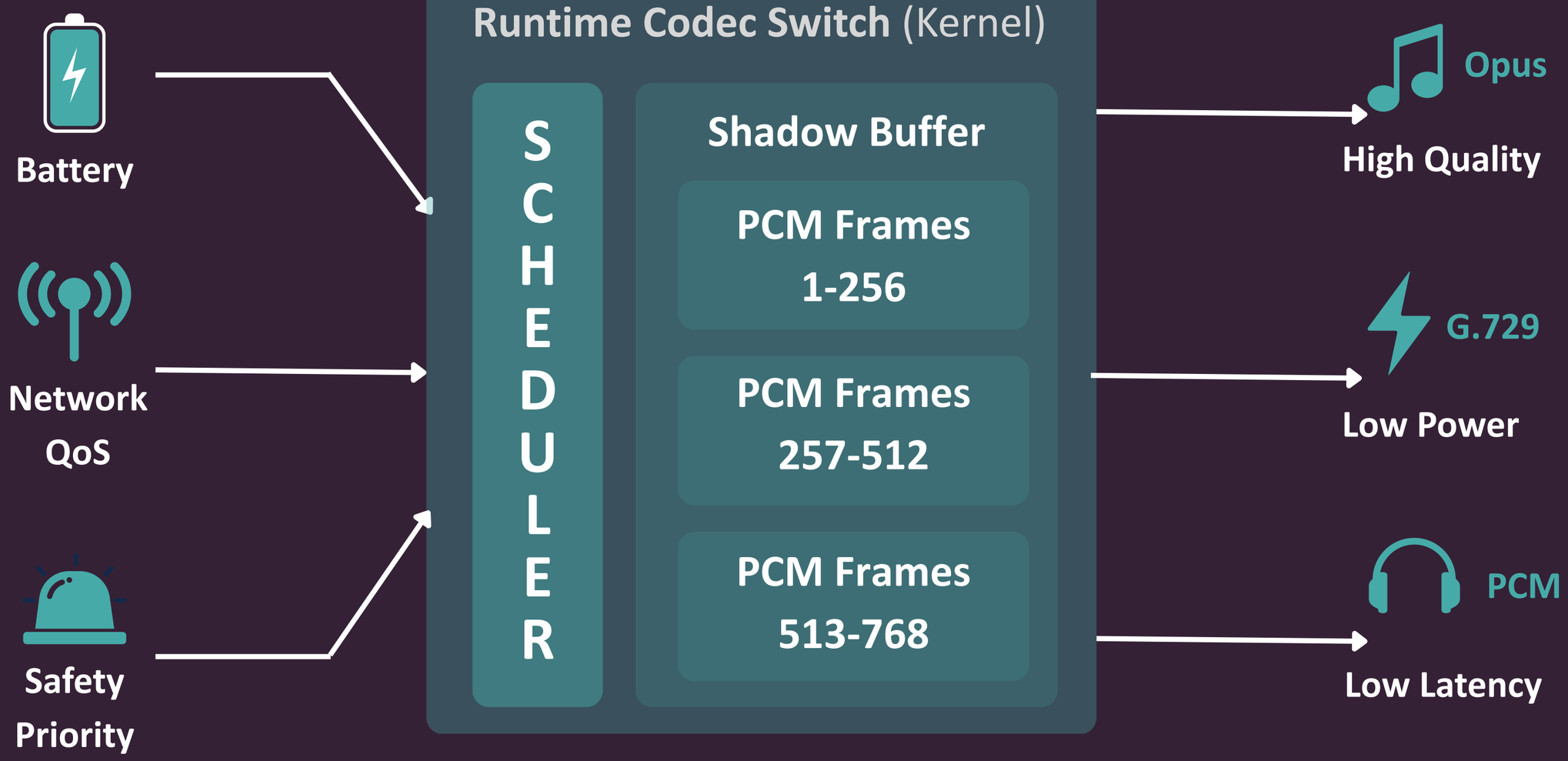
ALSA control interface + **Power-aware codec scheduler** + **Triple buffering in SoC driver** + **Device Tree profiles**

```
/* Enum control to select codec */
static const char * const codec_texts[] = { "Opus", "G729" };
static SOC_ENUM_SINGLE_EXT(codec_enum, codec_texts);

static int codec_set(struct snd_kcontrol *kcontrol,
                     struct snd_ctl_elem_value *ucontrol)
{
    int new_codec = ucontrol->value.enumerated.item[0];
    schedule_codec_switch(new_codec);
    return 0;
}
```

Switching **Concept**

SILICON SIGNALS



Switching **Concept (cont.)**

SILICON SIGNALS

```
/* Prepare shadow buffer */
void prepare_shadow_buffer(struct codec_stream *codec,
                           struct snd_pcm_substream *sub)
{
    fill_dma_buffer(codec->shadow_buf, sub);
}

/* Swap at frame boundary */
void switch_codec(struct codec_stream *old,
                  struct codec_stream *new)
{
    wait_for_period_boundary();
    dma_set_buffer(new->shadow_buf);
    active_codec = new;
}
```

Switching **Concept (cont.)**

SILICON SIGNALS

DTS

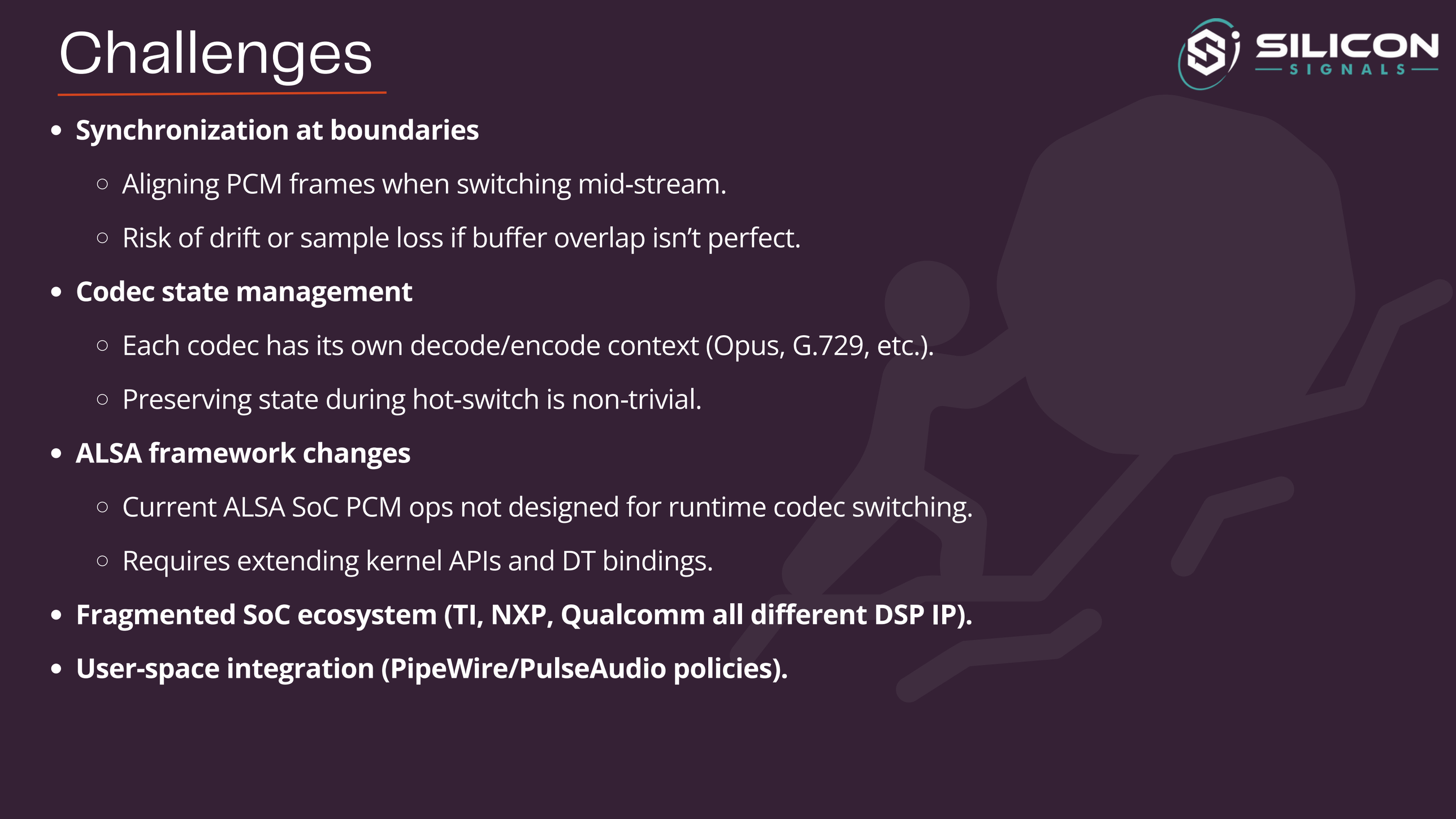
```
sound {
    compatible = "fsl,imx-audio";
    model = "imx8mp-codec-switch";
    codecs {
        opus {
            power-mw = <25>;
            priority = <2>;
        };
        g729 {
            power-mw = <5>;
            priority = <1>;
        };
    };
};
```

Expected **Benifits**

SILICON SIGNALS

- Based on analysis of ALSA’s buffering and DMA timing, this approach could reduce gaps to under a frame boundary (10–20 ms).
- CPU usage would also be lower than tearing down and rebuilding streams.
- Better battery life
- Meets safety requirements
- Works with multiple codecs/SoCs
- These are projected benefits, not measured results.

Challenges



- **Synchronization at boundaries**
  - Aligning PCM frames when switching mid-stream.
  - Risk of drift or sample loss if buffer overlap isn’t perfect.
- **Codec state management**
  - Each codec has its own decode/encode context (Opus, G.729, etc.).
  - Preserving state during hot-switch is non-trivial.
- **ALSA framework changes**
  - Current ALSA SoC PCM ops not designed for runtime codec switching.
  - Requires extending kernel APIs and DT bindings.
- **Fragmented SoC ecosystem (TI, NXP, Qualcomm all different DSP IP).**
- **User-space integration (PipeWire/PulseAudio policies).**

Work **Roadmap**

SILICON SIGNALS

Kernel Prototype

Upstream to ALSA

Codec topology integration

Multiple codec support

Policy integration (PipeWire)

SILICON SIGNALS

Thank You

+91-94087 30545

rutvij.trivedi@siliconsignals.io

www.siliconsignals.io