

Codec2 / ultra-low-bandwidth voice — design review (DR)

From DRs/20260617_codec2_ulbw_voice.md

Field	Value
Author	Anton Afanasyeu
Revision	R0
Creation date	2026-06-17
Last modification date	2026-06-17
Co-authored	Cursor Agent (project assistant)
Severity	low
State	approved for planning (implementation post-alpha)
Document type	DR
Pre-requisite to	ROADMAP.md §6.1 Opus/Speex JNI encode/decode; F1 SFU SPEC (post-alpha)

Table of contents

1. Executive summary
2. Problem statement
3. Vocabulary
4. Current state (mobile + BE)
 - 4.1 Audio codecs in app
 - 4.2 Video vs audio negotiation asymmetry
 - 4.3 Backend (gray)
5. Codec landscape and PO decisions
6. Goals and non-goals
 - Goals
 - Non-goals (R0)
7. Mobile peer negotiation (today)
8. Proposed mobile integration
 - 8.1 Settings and enum
 - 8.2 Native stack
 - 8.3 Priority (ULBW profile)
 - 8.4 Milestones
9. Future backend: MCU / SFU
 - 9.1 Negotiation shift (P2P → room)
 - 9.2 Codec2 on media plane (DR preference)
10. Protocol extensions
 - 10.1 CastCodecFlags
 - 10.2 MSG_AUDIO_SELECTED (recommended at M4)
 - 10.3 CastSettings.SETTINGS_VERSION
11. Third-party and licenses
12. Testing and validation
13. Task dependency graph
14. Risks
15. Open questions (remaining)
16. Changelog

Source: 20260617_codec2_ulbw_voice.md — section links work in PDF viewers that support internal anchors.

Document type: DR (Design Review)

Source draft: docs/drafts/20260617_codec2_ulbw_voice_integration.txt

PDF: 20260617_codec2_ulbw_voice.pdf · Regenerate: `bash scripts/build-all-docs-pdf.sh`

Status: PO-reviewed (inline draft comments incorporated) — **no native implementation** until Opus/Speex JNI sinks land

Scope: Voice-only / ULBW audio cast on P2P (LAN, BT, mesh); future MCU/SFU relay hooks

Related: OPUS_SPEEX_VALIDATION.md · ndk/README.md · 20260608_BE_SERVICES_and_infra.md § SFU · specs/20100612_1_scaling.md F1

1. Executive summary

Question: Should androidcast integrate **Codec2** for ultra-low-bandwidth (ULBW) **voice-only** casts when normal AAC/Opus paths are too heavy for the link?

Short answer: **Yes, as a post-alpha product slice** — not a lab experiment. End users on marginal links (BT P2P, slow mesh, very poor LTE) should get **intelligible audio** (“something rather than nothing”). **Codec2 @ 3200 bit/s** (and lower modes where needed) fits that goal better than Opus narrowband @ ~6 kbps + FEC in PO’s practical experience (lower sustained bandwidth, less audible noise on those links).

DR decision (R0):

Item	Decision
Primary use case	Voice-only ULBW cast; not the audio leg of full screen cast (8 kHz telephony; no lip-sync with HD video)
Codec2 modes	3200 bit/s default target; 700C for extreme BT/mesh; user/dev sub-mode selection
MELP / TWELP	Out of scope (patent / slim OSS path)
AAC	Keep as platform MediaCodec path; no license conflict for screen cast
Opus / Speex	Finish JNI encode/decode first (ROADMAP.md — cross-compile done, sinks TODO); negotiation/scaffolding exists today but wire path is stub → AAC without <code>libopus.a</code> / encode JNI
Codec2 timing	After 6.1 JNI baseline; behind dev flag until E2E validated
SFU / MCU	Plan transcode at ingress (Codec2 → Opus) for mixed WebRTC rooms; separate F1 SPEC — signaling reuses extended audio caps
MSG_AUDIO_SELECTED	Add in handshake when Codec2 implementation starts (SETTINGS_VERSION bump) — aligns P2P multicast and SFU data channel

2. Problem statement

Screen cast defaults to **AAC** (32–128 kbps class). Negotiation already advertises **Opus** and **Speex**, but without native encode/decode the sender often still ships **AAC** while labeling Opus in diagnostics.

For **ULBW** links:

- Opus @ 6 kbps + FEC/NACK can exceed the sustained budget PO has seen on BT/mesh and adds noise versus Codec2-class vocoders.
- Users still want **audio cast** (commentary, coordination, voice relay) when video is impossible — a deliberate **voice-only** mode.

Future **multipoint** (F1 SFU/MCU) cannot rely on pairwise discovery bitmasks alone; room-level **negotiated_audio** must be defined before Codec2 legs join a relay.

3. Vocabulary

Term	Meaning
ULBW	Ultra-low bandwidth; sustained < 3–4 kbps audio payload target on worst links
Voice-only cast	No video track; discovery may send <code>videoCodecs = 0</code>
Radio mode	Dev/user profile preferring Codec2 in AUTO when both peers advertise <code>AUDIO_CODEC2</code>
P2P negotiation	<code>CastProtoHeader.audioCodecs</code> bitmask + <code>AudioNegotiator</code> (today)
Room negotiation	Signaling API + server-published codec (SFU future)
SFU	Selective forwarding unit — relay without per-subscriber transcode
MCU	Mix decode → PCM → encode; needed for heterogeneous legs

4. Current state (mobile + BE)

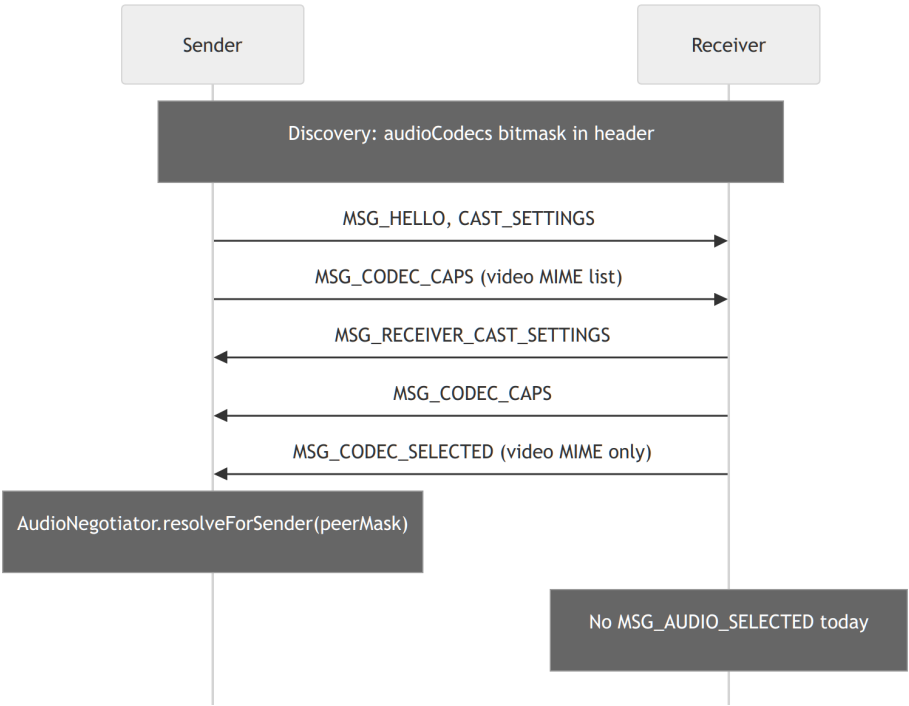
4.1 Audio codecs in app

Layer	Opus / Speex today	Notes
<code>CastCodecFlags</code>	<code>AUDIO_OPUS=4</code> , <code>AUDIO_SPEEX=8</code>	Masks advertised in discovery
<code>AudioNegotiator</code>	Works	AUTO scoring via <code>CodecPriorityCatalog</code>

NativeCodecBridge	Probe only (nativeIsOpusAvailable)	No encode/decode JNI in opus_bridge.c / speex_bridge.c
Wire path	OPUS_STUB / SPEEX_STUB → AAC	PassthroughCodecPolicy, AudioCodecFactory
Build	scripts/build-native-codecs.sh can link .a	ROADMAP.md: JNI sinks TODO

Clarification for PO: Prior agent work completed **negotiation, settings UI, priority catalog, and native autolink scaffolding** — not full Opus/Speex on-the-wire encoding. Task **6.1** remains open for JNI sinks + E2E per OPUS_SPEEX_VALIDATION.md.

4.2 Video vs audio negotiation asymmetry



4.3 Backend (gray)

- SfuRelayGate.PREVIEW_ENABLED = false; api/sfu_health.php returns disabled.
- Janus on :8089 is **other stack** — androidcast signaling stays on apps.f0xx.org PHP (INFRA.md).

5. Codec landscape and PO decisions

Codec	Bitrate	Role in androidcast	DR decision
Codec2	700–3200 bps	ULBW voice-only	Integrate (post-6.1)
Opus SILK NB	~6–12 kbps	General narrowband when link allows	Finish JNI; keep in AUTO above Codec2 only when bandwidth sufficient
Speex NB	~2–15 kbps	Legacy / fallback	Finish JNI
AAC-LC	32+ kbps	Screen cast default	Unchanged
MELP/TWELP	~2.4 kbps	—	Rejected

PO rationale (2026-06-17): Opus @ 6k + FEC is **heavier and noisier** than Codec2 on target links; Codec2 **3200 bps** meets product goals; intent is **end-user value**, not codec shootout lab work.

Maintainer risk: Codec2 upstream is in active re-development; pin submodule tag (e.g. 1.2.0) and mirror on Gitea like third-party/opus.

6. Goals and non-goals

Goals

- Voice-only cast mode with **Codec2 encode/decode** on Android (NDK).
- Extend **peer negotiation** (AUDIO_CODEC2 = 16, optional MSG_AUDIO_SELECTED).
- Document **SFU ingress transcode** and room caps for F1 SPEC.
- LGPL compliance + license files.

Non-goals (R0)

- MELP/TWELP integration.
- Codec2 as default for **screen cast with video**.
- SFU passthrough-only design (closed Android mesh edge case only).
- Alpha ship — hidden until E2E + PO QA.

7. Mobile peer negotiation (today)

Step	Mechanism	Codec2 impact
Advertise	CastCodecFlags.localAudioMask() in discovery	Add bit 16 when native probe passes
Resolve	AudioNegotiator.negotiate(senderMask, receiverMask, pref, settings)	Add CODEC2 to enum + CodecPriorityId
Score	codecs.json + CodecPriorityCatalog	ULBW profile: CODEC2 lowest bitrate_score
Wire	Audio frames (today AAC ADTS)	New codec tag or MSG_AUDIO_FRAME subtype

Gap: Receiver does not receive explicit audio selection — sender-resolved only. **Must fix** before multicast + SFU (\$10).

8. Proposed mobile integration

8.1 Settings and enum

```
CastSettings.AudioCodec.CODEC2
CastSettings.codec2Mode: 3200 | 1300 | 700C (default 3200 per P0)
CastSettings.VoiceOnlyMode: boolean
```

8.2 Native stack

1. third-party/codec2 submodule + scripts/init-third-party-submodules.sh
2. scripts/build-native-codecs.sh → libcodec2.a, ANDROIDCAST_HAVE_CODEC2
3. JNI: isCodec2Available(), encode/decode **8 kHz mono** frames
4. Link via existing libandroidcast_codecs.so

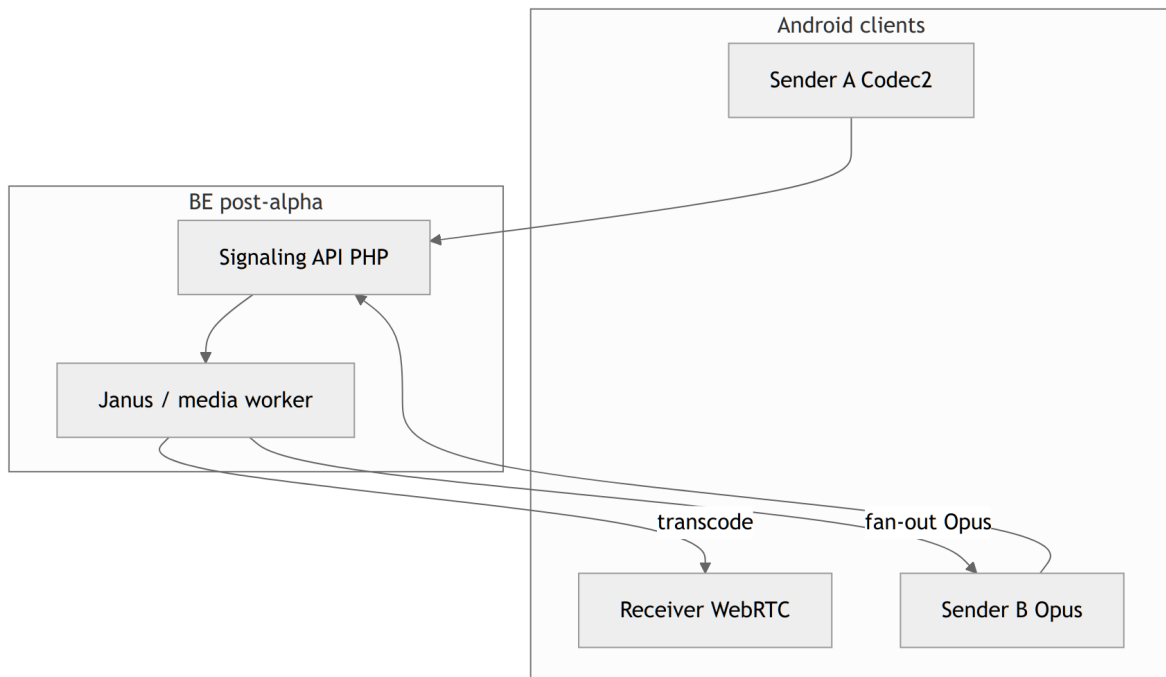
8.3 Priority (ULBW profile)

Suggested AUTO order when NetworkAdoption.ESTIMATED or **radio mode** on:
CODEC2 → OPUS → SPEEX → AAC → PCM

8.4 Milestones

#	Deliverable	Depends on
M1	Opus/Speex JNI encode/decode + E2E	6.1
M2	Codec2 probe + round-trip unit test	M1
M3	Voice-only UI slice (dev gate)	M2
M4	MSG_AUDIO_SELECTED + SETTINGS_VERSION	M2
M5	SFU signaling fields in F1 SPEC	Owner SFU spec

9. Future backend: MCU / SFU



9.1 Negotiation shift (P2P → room)

Layer	P2P today	SFU future
Caps	Header bitmask	join JSON: audio_caps.bitmask, codec2_modes[]
Pick	AudioNegotiator local	Server negotiated_audio
Auth	PIN / LAN	Session RBAC + room token

9.2 Codec2 on media plane (DR preference)

Mode	When	CPU on be-media-N
Transcode ingress	Mixed room (Android + browser)	Recommended — decode Codec2 → Opus egress
Passthrough	All Codec2 clients	Low CPU; no WebRTC viewers
MCU mix	Conference voice	High CPU; radio bridge legs only

Signaling sketch (F1 SPEC — not implemented):

- POST /api/sfu/room — ingress_caps: [opus, codec2], egress_caps: [opus]
- POST /api/sfu/room/{id}/join — returns transport params + negotiated_audio

Mobile: SfuSignalingClient fills CastSettings.resolvedAudioCodec from server; **LAN fallback** if signaling down.

10. Protocol extensions

10.1 CastCodecFlags

AUDIO_CODEC2 = 16 // next free bit after SPEEX=8

10.2 MSG_AUDIO_SELECTED (recommended at M4)

payload: uint32 audio_flag (little-endian) + uint8 codec2_mode_id

Symmetric with MSG_CODEC_SELECTED; reusable on WebRTC data channel.

10.3 CastSettings.SETTINGS_VERSION

Bump when adding sfu_room_id, relay_mode, server_negotiated_audio (SFU phase).

11. Third-party and licenses

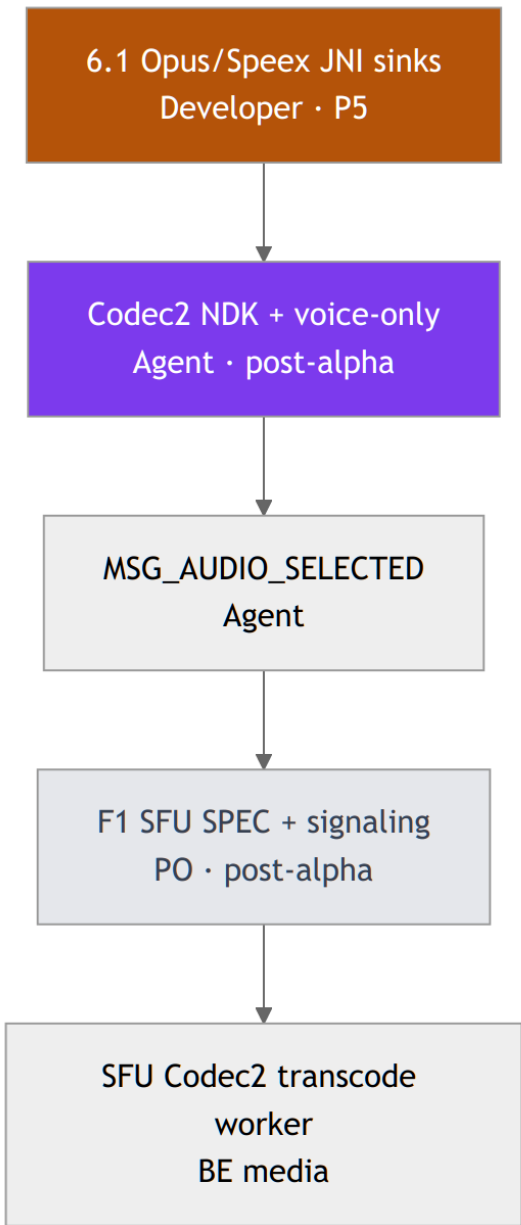
Component	License	Action
codec2 (libcodec2)	LGPL-2.1	Submodule + Gitea mirror; dynamic link in .so
Opus / Speex	BSD	Existing
ffmpeg (SFU transcode)	LGPL/GPL	BE media worker SPEC only

Update mobile LICENSES / notices when submodule added (bottomline_reminder.txt).

12. Testing and validation

Phase	Tests
Unit	AudioNegotiatorTest CODEC2 mask; CastProtoHeader round-trip
Native	JNI c2enc/c2dec parity arm64-v8a
E2E	Extend OPUS_SPEEX_VALIDATION.md — voice-only, UDP/TCP, diagnostics kbps
SFU	Post-alpha: 2× Codec2 Android + 1× WebRTC viewer via transcode

13. Task dependency graph



- Numbered priority (PO overrides AI):
- 1. **6.1** — Opus/Speex JNI (alpha LAN quality path; unblocks honest ULBW comparison)
 - 2. **Codec2 ULBW** — this DR (parallel after M1 green)
 - 3. **F1 SFU SPEC** — room negotiation + transcode
 - 4. Alpha blockers **5.1 / 5.2 / 4.rssh** remain above on OPEN_TASKS_GRAPH.md

14. Risks

Risk	Mitigation
------	------------

Codec2 upstream churn	Pin tag; Gitea mirror
8 kHz-only confuses users expecting music	Voice-only mode UX; no video leg
SFU CPU cost	Ingress transcode budget in scaling SPEC
Opus/Speex "done" confusion	ROADMAP + this DR state JNI sinks explicitly
BT latency 150–300 ms	Document; jitter buffer 80–120 ms

15. Open questions (remaining)

#	Question	Owner
1	SFU commercial story: always transcode vs optional passthrough mesh?	PO
2	Ship Codec2 in AUTO on ULBW profile or explicit "radio mode" only?	PO + UX
3	700C exposed to end users or dev-only?	PO

Resolved by PO (2026-06-17): voice-only ULBW product intent; Codec2 3200 OK; skip MELP/TWELP; Opus @ 6k insufficient for target links; AAC platform use OK.

16. Changelog

Rev	Date	Change
R0	2026-06-17	Initial DR from draft; PO inline comments; Opus/Speex JNI status clarified