Audio Processing Reference Guide | Version 1.0

🎵 Audio Processing & FFmpeg Parameters Guide

A comprehensive guide to understanding audio decoding, WAV format, and FFmpeg configuration for audio fingerprinting applications

1. The Processing Flow

```
Browser (WebRTC Recording)
↓
WebM/Opus Container (Compressed)
↓
Spring Boot Buffer (SessionAudioBuffer)
↓
FFmpeg Decoder
↓
WAV File (Uncompressed PCM)
↓
Fingerprinting Service
↓
Match Result
```

💡 **Why this flow?**

• Browser records as WebM/Opus (compressed, efficient for transmission)

• FFmpeg decodes to WAV (raw audio samples, easy to analyze)

• Fingerprinting algorithms require uncompressed PCM data to analyze acoustic patterns

2. Core Audio Concepts

2.1 Sample Rate (Frequency)

> **Definition:** The number of audio samples captured per second, measured in Hertz (Hz)
>
> **Analogy:** Like frames per second (FPS) in video—higher sample rates capture more detail and produce smoother, more accurate audio reproduction

Sample Rate	Quality Level	Common Use
8,000 Hz	Very Low	Telephone systems
16,000 Hz	Low	Voice calls, speech recognition
22,050 Hz	Medium	Web audio, podcasts
44,100 Hz	**Standard (CD Quality)**	**Music, fingerprinting, consumer audio**
48,000 Hz	Professional	Video production, broadcasting
96,000 Hz	Studio	High-resolution recording

> **Why 44,100 Hz?**
> Based on the Nyquist-Shannon sampling theorem: to accurately capture audio, you need a sample rate at least twice the highest frequency you want to record. Human hearing ranges from 20 Hz to 20,000 Hz, so 44,100 Hz (which captures up to ~22,050 Hz) is perfect for music.

2.2 Audio Channels

> **Definition:** The number of independent audio signals (streams) in an audio file
>
> **Analogy:** Like camera angles—mono is a single viewpoint, stereo is left and right perspectives

Channels	Name	Description	File Size
1	**Mono**	**Single channel, center audio**	**Baseline (100%)**
2	Stereo	Left and right channels	2× larger
6	5.1 Surround	Front L/R, Center, Rear L/R, Subwoofer	6× larger

> **For Fingerprinting: Use Mono (-ac 1)**
> ✓ Reduces file size by 50% compared to stereo
> ✓ Faster processing time
> ✓ Most fingerprinting algorithms convert to mono internally anyway
> ✓ Spatial information (stereo separation) is not needed for acoustic matching

2.3 Bit Depth (Sample Resolution)

> **Definition:** The number of bits used to represent each audio sample, determining the precision and dynamic range
>
> **Analogy:** Like color depth in images—8-bit has 256 colors, 16-bit has 65,536 shades of color

Bit Depth	Possible Values	Dynamic Range	Use Case
8-bit	256 levels	48 dB	Low quality, retro games
16-bit	**65,536 levels**	**96 dB**	**CD quality, standard audio**
24-bit	16.7 million levels	144 dB	Professional recording
32-bit float	4.3 billion levels	1,680 dB	Studio mastering, editing

For Fingerprinting: Use 16-bit (pcm_s16le)
• Perfect balance of quality and file size
• Industry standard for music distribution
• More than sufficient dynamic range for analysis
• Supported by all audio processing libraries

3. Understanding pcm_s16le

> **Breaking down the codec name: pcm_s16le**
>
> **PCM** = Pulse Code Modulation (uncompressed digital audio)
> **s16** = Signed 16-bit (values from -32,768 to +32,767)
> **le** = Little Endian (byte order used by most computers)

Why "signed"? Audio waveforms oscillate above and below zero (positive and negative pressure), so we need both positive and negative numbers to represent them accurately.

What is Little Endian? It's the order in which bytes are stored in memory. Modern Intel/AMD processors use little endian, making this the standard choice for PC-based systems.

4. Bit Rate Calculation

```
Bit Rate Formula:

Bit Rate (bits/second) = Sample Rate × Bit Depth ×
Channels

Example (Your Configuration):
= 44,100 samples/sec × 16 bits × 1 channel
= 705,600 bits/second
= 705.6 kbps
= 86.4 KB/second
```

File Size Calculation

Duration	File Size (44.1kHz, 16-bit, Mono)
10 seconds	~864 KB

Duration	File Size (44.1kHz, 16-bit, Mono)
30 seconds	~2.5 MB
1 minute	~5.2 MB
3 minutes	~15.6 MB
5 minutes	~26 MB

5. WAV File Format

What is WAV?

WAV (Waveform Audio File Format) is a Microsoft/IBM audio file format standard for storing an audio bitstream on PCs. It is the main format used on Windows systems for raw and typically uncompressed audio.

WAV File Structure

```
┌──────────────────────────────────────┐
│ RIFF Header │
│ (File identifier and size) │
├──────────────────────────────────────┤
│ fmt Chunk │
│ • Sample rate: 44100 Hz │
│ • Channels: 1 (mono) │
│ • Bit depth: 16-bit │
│ • Codec: PCM │
├──────────────────────────────────────┤
│ data Chunk │
│ [Raw PCM samples] │
│ Sample 1: -5432 │
│ Sample 2: -3211 │
│ Sample 3: 1024 │
│ ... │
└──────────────────────────────────────┘
```

Why WAV for Fingerprinting?

✓ **No Compression Artifacts:** Pure, unaltered audio samples
✓ **Direct Sample Access:** Easy to read sequential samples for analysis
✓ **Universal Support:** Every audio processing library understands

> WAV
> ✓ **Fast Decoding:** Already uncompressed, no decode overhead
> ✓ **Predictable Format:** Simple structure makes parsing reliable

Opus vs WAV Comparison

Aspect	Opus (Input)	WAV (Output)
Compression	Lossy (psychoacoustic model)	Uncompressed (lossless)
File Size (1 min)	~1.5 MB	~5.2 MB
Data Rate	~20-50 KB/sec (variable)	86.4 KB/sec (constant)
Processing Speed	Requires decoding	Direct access to samples
Use Case	Streaming, storage, transmission	Analysis, processing, editing
Quality Loss	Yes (imperceptible at high bitrates)	No

6. FFmpeg Command Breakdown

Your Current Command

```
ffmpeg -y -i input.webm -ac 1 -ar 44100 -acodec pcm_s16le -f
wav output.wav
```

Parameter Explanation

Parameter	Full Name	Purpose
-y	Yes/Overwrite	Automatically overwrite output file if it exists (no prompt)
-i input.webm	Input file	Specifies the source file containing Opus-encoded audio
-ac 1	Audio Channels	Convert to mono (1 channel) for efficient processing
-ar 44100	Audio Rate	Set sample rate to 44,100 Hz (CD quality)
-acodec pcm_s16le	Audio Codec	Encode as 16-bit signed PCM, little-endian
-f wav	Format	Force output format to WAV container
output.wav	Output file	Destination file path

The Conversion Process (Step-by-Step)

1. **Read Container:** FFmpeg opens the WebM container and identifies the Opus audio stream
2. **Decode Opus:** Decompresses Opus-encoded audio to raw PCM samples
3. **Downmix Channels:** If input is stereo, combines to mono using -ac 1

4. **Resample:** Converts sample rate to 44,100 Hz if different
5. **Format Conversion:** Ensures data is in signed 16-bit little-endian format
6. **Create WAV:** Wraps PCM data in WAV container with proper headers
7. **Write File:** Saves the complete WAV file to disk

7. Optimal Settings for Different Use Cases

Music Fingerprinting (Recommended - Your Use Case)

```
ffmpeg -y -i input.webm -ac 1 -ar 44100 -acodec pcm_s16le -f
wav output.wav
```

Quality: ★★★★★ | **File Size:** 5.2 MB/min | **Processing:** Medium

Best for: Music recognition, acoustic fingerprinting, song identification

Voice/Speech Recognition

```
ffmpeg -y -i input.webm -ac 1 -ar 16000 -acodec pcm_s16le -f
wav output.wav
```

Quality: ★★★☆☆ | **File Size:** 1.9 MB/min | **Processing:** Fast

Best for: Speech-to-text, voice commands, telephone quality audio

Maximum Quality (Overkill)

```
ffmpeg -y -i input.webm -ac 2 -ar 48000 -acodec pcm_s24le -f
wav output.wav
```

Quality: ★★★★★★ | **File Size:** 16.9 MB/min | **Processing:** Slow

Best for: Archival, professional audio analysis, research

Minimum Viable (Fast & Small)

```
ffmpeg -y -i input.webm -ac 1 -ar 22050 -acodec pcm_s16le -f
wav output.wav
```

Quality: ★★★☆☆ | **File Size:** 2.6 MB/min | **Processing:** Very Fast

Best for: Quick prototypes, low-resource environments, testing

8. Common Questions & Answers

Q: Why not fingerprint Opus directly?

A: Fingerprinting algorithms analyze raw amplitude values over time to create acoustic signatures. Opus is compressed using psychoacoustic models that discard "inaudible" information, making direct analysis complex and unreliable. WAV provides direct access to sample data.

Q: Can I use lower quality settings to save space?

A: Yes, but with trade-offs:
• **Lower sample rate (22050 Hz):** Works for speech, may miss high-frequency content in music
• **8-bit audio:** Noticeable quality degradation, not recommended
• **Consider:** Processing speed vs. accuracy for your use case

Q: What's the minimum quality for reliable fingerprinting?

A: Minimum recommended specifications:
• Sample Rate: 11,025 Hz absolute minimum, 44,100 Hz recommended
• Bit Depth: 16-bit minimum
• Channels: Mono acceptable (stereo unnecessary)
• Format: Uncompressed PCM (WAV, AIFF, or raw PCM)

Q: Why do WAV files grow so large?

A: You're converting from compressed to uncompressed format. Think of it like unzipping a ZIP file—the original data is restored to its full size. For 44.1kHz/16-bit/mono, you need 86.4 KB of storage for every second of audio.

9. Implementation Best Practices

Resource Management

- **Delete temporary files immediately:** WAV files are large; clean up after processing
- **Use streaming when possible:** Process audio in chunks for large files
- **Monitor disk space:** Ensure adequate temporary storage (100MB+ buffer recommended)
- **Set timeouts:** FFmpeg processes can hang on corrupted files

Error Handling

- **Check exit codes:** FFmpeg returns 0 on success, non-zero on failure
- **Verify output:** Confirm WAV file exists and has non-zero size
- **Handle corrupted input:** Catch and gracefully handle decoding failures
- **Log FFmpeg output:** Capture stderr for debugging purposes

Performance Optimization

- **Process asynchronously:** Use thread pools for concurrent processing
- **Limit queue size:** Prevent memory exhaustion under heavy load
- **Consider sample rate:** Lower rates (22050 Hz) can halve processing time
- **Batch processing:** Group small files together when possible

10. Quick Reference Table

Setting	Recommended Value	Rationale
Sample Rate	**44,100 Hz**	**CD quality, captures full frequency range**
Channels	**1 (Mono)**	**Reduces size, sufficient for fingerprinting**
Bit Depth	**16-bit**	**Industry standard, excellent dynamic range**
Codec	**pcm_s16le**	**Uncompressed PCM, universal compatibility**
Format	**WAV**	**Simple structure, easy to process**
Bit Rate	705.6 kbps	Calculated automatically from above settings
File Size	~5.2 MB/minute	Expected size for mono 44.1kHz 16-bit audio

11. Docker Deployment Checklist

Essential Requirements:

✓ Install FFmpeg in Docker container (apt-get install ffmpeg)
✓ Verify FFmpeg with: docker exec container ffmpeg -version
✓ Ensure adequate temp storage (/tmp directory)
✓ Set appropriate file permissions for temp files
✓ Configure resource limits (memory, CPU) in docker-compose
✓ Implement cleanup for orphaned temp files on restart

Dockerfile Example

```
FROM eclipse-temurin:17-jre-jammy

# Install FFmpeg
RUN apt-get update && \
    apt-get install -y ffmpeg && \
```

```
    rm -rf /var/lib/apt/lists/*

# Verify installation
RUN ffmpeg -version

WORKDIR /app
COPY target/*.jar app.jar

# Create temp directory
RUN mkdir -p /tmp/audio && chmod 777 /tmp/audio

EXPOSE 8080
ENTRYPOINT ["java", "-jar", "app.jar"]
```

12. Troubleshooting Guide

Problem: "Cannot run program 'ffmpeg': No such file or directory"

Cause: FFmpeg not installed or not in system PATH

Solutions:
1. Install FFmpeg: sudo apt install ffmpeg
2. Use full path: /usr/bin/ffmpeg instead of ffmpeg
3. Verify with: which ffmpeg or ffmpeg -version
4. In Docker: Add RUN apt-get install -y ffmpeg to Dockerfile

Problem: "FFmpeg conversion failed, exit code non-zero"

Causes: Corrupted input file, unsupported codec, insufficient permissions

Solutions:
1. Check input file validity
2. Capture and log FFmpeg stderr output for details
3. Verify temp directory write permissions
4. Test command manually: ffmpeg -i input.webm output.wav

Problem: "Output file is empty or very small"

Causes: Input file has no audio, wrong stream selected, format mismatch

Solutions:
1. Check input file duration and streams: ffmpeg -i input.webm
2. Verify input file size is reasonable

> 3. Ensure input contains audio (not just video)
> 4. Add -vn flag to explicitly ignore video streams

Problem: "Process takes too long or hangs"

> **Causes:** Large files, high sample rates, system resource constraints
>
> **Solutions:**
> 1. Implement timeout on Process.waitFor()
> 2. Consider lower sample rate (22050 Hz) for faster processing
> 3. Add progress monitoring using FFmpeg's -progress flag
> 4. Check system CPU and memory availability

Problem: "Disk space issues with temp files"

> **Causes:** Temp files not being cleaned up, high processing volume
>
> **Solutions:**
> 1. Always delete temp files in finally blocks
> 2. Use Files.deleteIfExists() to handle missing files gracefully
> 3. Monitor /tmp directory size
> 4. Implement maximum queue size to prevent overwhelming system
> 5. Consider streaming processing for very large files

13. Audio Format Comparison

Format	Type	Size (1 min)	Quality	Use Case
Opus	Lossy Compressed	~1.5 MB	High	Streaming, VoIP, recording
WAV (PCM)	**Uncompressed**	**~5.2 MB**	**Perfect**	**Processing, analysis, editing**
MP3	Lossy Compressed	~1 MB	Good	Distribution, playback
FLAC	Lossless Compressed	~3 MB	Perfect	Archival, audiophile
AAC	Lossy Compressed	~0.9 MB	High	Mobile, streaming

14. Performance Metrics

Expected Processing Times (approximate)

Audio Duration	Processing Time	CPU Usage	Memory Usage
10 seconds	0.5 - 1 second	Low-Medium	~50 MB
30 seconds	1 - 2 seconds	Medium	~75 MB
1 minute	2 - 4 seconds	Medium	~100 MB
3 minutes	6 - 12 seconds	Medium-High	~200 MB
5 minutes	10 - 20 seconds	High	~300 MB

Note: Times vary based on system specifications, CPU speed, and concurrent load

15. Summary & Key Takeaways

Your Current Configuration is Optimal:

The settings you're using (44.1kHz, mono, 16-bit PCM WAV) represent the industry standard for audio fingerprinting and music analysis. This configuration provides:

✓ Full frequency range capture (20 Hz - 22 kHz)
✓ Excellent dynamic range (96 dB)
✓ Reasonable file sizes (~5 MB per minute)
✓ Fast processing times
✓ Universal compatibility with fingerprinting algorithms

Critical Points to Remember

1. **Sample Rate (44,100 Hz)** determines how often audio is measured—higher rates capture more detail
2. **Channels (Mono = 1)** reduces file size by 50% without impacting fingerprint accuracy
3. **Bit Depth (16-bit)** provides 65,536 levels of precision—perfect for music analysis
4. **WAV format** stores uncompressed PCM data, making it ideal for acoustic analysis
5. **FFmpeg** acts as the bridge between compressed browser recordings and analysis-ready audio
6. **Docker deployment** requires FFmpeg installation in the container, not on the host
7. **Cleanup is essential**—always delete temporary files to prevent disk space issues

When to Consider Different Settings

- **Use 16 kHz sample rate** if processing only speech/voice (not music)
- **Use 22.05 kHz** if you need faster processing and can accept slightly reduced quality
- **Keep stereo (2 channels)** only if your algorithm specifically requires spatial information
- **Use 24-bit** only for archival or research purposes—overkill for fingerprinting

16. Additional Resources

FFmpeg Documentation

- Official Documentation: https://ffmpeg.org/documentation.html
- Audio Filters Guide: https://ffmpeg.org/ffmpeg-filters.html#Audio-Filters
- Format Reference: https://ffmpeg.org/ffmpeg-formats.html

Audio Processing Concepts

- Nyquist-Shannon Sampling Theorem: Foundation of digital audio
- PCM (Pulse Code Modulation): Standard digital audio representation
- Bit Depth & Dynamic Range: Understanding audio precision
- Sample Rate Theory: Why 44.1 kHz became the standard

Related Technologies

- **Chromaprint/AcoustID:** Open-source audio fingerprinting
- **Shazam Algorithm:** Commercial music recognition
- **Echoprint:** Open-source music identification
- **WebRTC:** Browser-based real-time audio capture

Audio Processing & FFmpeg Guide | Version 1.0

For audio fingerprinting applications using Spring Boot and FFmpeg

Generated: October 17, 2025 at 02:51 PM