MỤC LỤC

Chương 3. Số hóa âm thanh.....7
3.1 Lấy mẫu và định lý Nyquist – Shannon.....8
3.2 Lượng tử hóa và độ sâu bit.....13
3.3 Thang dBFS và biểu diễn mẫu trong máy tính.....16
3.4 Định dạng và chuẩn mã hóa.....17
3.5 Cập nhật công nghệ: mã hóa âm thanh nơ-ron.....20
3.6 Hướng dẫn thực hành.....22
3.7 Hệ thống bài tập và câu hỏi ôn tập.....26
3.8 Thuật ngữ chuyên ngành Anh – Việt.....29

DANH MỤC HÌNH VẼ

Hình 3.1 Mặt trước audio interface 18 ngõ vào/8 ngõ ra (Focusrite Scarlett 18i8), nơi tín hiệu micro được khuếch đại rồi đưa vào khối ADC.....9
Hình 3.2 Dạng sóng âm sin 440 Hz phóng đại tối đa trong Audacity: mỗi chấm là một mẫu, trục thời gian tính bằng giây.....9
Hình 3.3 Rời rạc hóa trục thời gian: sóng analog liên tục được đo tại các mốc thời gian cách đều nhau.....10
Hình 3.4 Hiện tượng chồng phổ: sóng tần số cao vượt ngưỡng Nyquist bị lấy mẫu thưa và biến thành sóng giả mạo tần số thấp.....11
Hình 3.5 Ảnh hiển vi điện tử quét các rãnh dữ liệu trên bề mặt đĩa compact; vạch chuẩn dài 5 µm.....12
Hình 3.6 Khuôn bán dẫn của bộ chuyển đổi ADC 12 bit Analog Devices AD574 (ảnh chụp dưới kính hiển vi).....13
Hình 3.7 Lượng tử hóa trục biên độ, sai số lượng tử và quan hệ tuyến tính giữa độ sâu bit với dải động.....14
Hình 3.8 Phổ của âm thuần 750 Hz sau khi lượng tử hóa 4 bit, không dither, không định dạng nhiễu: ngoài vạch gốc còn nhiều vạch méo.....15
Hình 3.9 Ba nhóm định dạng tệp âm thanh và vị trí sử dụng trong quy trình sản xuất..17
Hình 3.10 Hộp thoại Export Audio của Audacity với danh sách định dạng tệp xuất.…18
Hình 3.11 Phổ của cùng đoạn nhạc ở WAV 16 bit (trái) và MP3 128 kbps (phải), trục tần số tuyến tính, cửa sổ Hann, FFT 4096 (Audacity).....20
Hình 3.12 Dùng phổ đồ để nhận diện tệp “Fake Lossless”: vạch cắt brick-wall là dấu vết không thể chối cãi của nén có tổn hao.....20
Hình 3.13 Kiến trúc bộ mã hóa âm thanh nơ-ron: encoder – lượng tử RVQ – chuỗi token – decoder.....22
Hình 3.14 Thông số kỹ thuật của tệp WAV (trên) và tệp MP3 128 kbps (dưới) trong MediaInfo, chế độ Tree.....24

DANH MỤC BẢNG BIỂU

Bảng 3.1 Ký hiệu và viết tắt dùng trong Chương 3.....7
Bảng 3.2 Các tần số lấy mẫu chuẩn và ứng dụng điển hình.....12
Bảng 3.3 So sánh độ sâu bit và dải động kỹ thuật.....14
Bảng 3.4 Bảng tổng hợp đặc tính các nhóm định dạng âm thanh.....19
Bảng 3.5 Các cột mốc Neural Audio Codec tiên tiến.....21
Bảng 3.6 Bảng tra cứu thuật ngữ Anh — Việt số hóa âm thanh.....29

DANH MỤC KÝ HIỆU VÀ CHỮ VIẾT TẮT

Ký hiệu viết tắt	Thuật ngữ tiếng Anh đầy đủ	Ý nghĩa chuyên môn trong kỹ thuật Audio - Video
ADSR	Attack - Decay - Sustain - Release	Mô hình đường bao biên độ 4 giai đoạn theo thời gian trong thiết kế tiếng động
AES	Audio Engineering Society	Hội Kỹ sư Âm thanh Quốc tế — tổ chức ban hành các tiêu chuẩn âm thanh chuyên nghiệp
DAW	Digital Audio Workstation	Trạm làm việc âm thanh số (Pro Tools, DaVinci Resolve Fairlight, Nuendo, Reaper)
dB	Decibel	Đơn vị logarit biểu diễn tỷ số giữa hai đại lượng biên độ hoặc công suất
dB SPL	Decibel Sound Pressure Level	Mức áp suất âm thanh trong không khí quy chiếu theo ngưỡng nghe 20 µPa
dBFS	Decibels Relative to Full Scale	Thang đo decibel số toàn dải; 0 dBFS là trần tuyệt đối của hệ thống kỹ thuật số
dBTP	Decibels True Peak	Mức đỉnh thực của tín hiệu số có tính toán hiện tượng liên mẫu (inter-sample peaks)
dBA / dBC	A-weighted / C-weighted Decibels	Mức đo decibel qua mạch lọc bù trừ đặc tuyến độ nhạy của tai người
EBU	European Broadcasting Union	Hiệp hội Phát thanh Truyền hình Châu Âu (tác giả tiêu chuẩn EBU R128)
HRTF	Head-Related Transfer Function	Hàm truyền liên quan đến đầu người — cơ sở định vị không gian 3 chiều và âm thanh nhị nhĩ
ILD	Interaural Level Difference	Độ chênh lệch mức áp suất âm thanh giữa hai tai (chủ đạo ở tần số cao > 1,5 kHz)
ITD	Interaural Time Difference	Độ chênh lệch thời gian âm thanh truyền đến hai tai (chủ đạo ở tần số thấp < 1,5 kHz)
ITU-R	ITU Radiocommunication Sector	Phân ban Vô tuyến của Liên minh Viễn thông Quốc tế (tác giả chuẩn BS.1770)
LKFS / LUFS	Loudness K-weighted / Units Full Scale	Đơn vị đo lường độ to cảm nhận chuẩn hóa theo ITU-R BS.1770
LRA	Loudness Range	Dải độ to — chỉ số mô tả độ động giữa đoạn to và đoạn nhỏ của chương trình (đơn vị LU)
NC / NR	Noise Criteria / Noise Rating	Đường cong tiêu chí đánh giá mức độ ồn nền của phòng thu

Ký hiệu viết tắt	Thuật ngữ tiếng Anh đầy đủ	Ý nghĩa chuyên môn trong kỹ thuật Audio - Video
NIOSH	National Institute for Occupational Safety and Health	Viện Quốc gia về An toàn và Sức khỏe Nghề nghiệp Hoa Kỳ
RMS	Root Mean Square	Giá trị căn bậc hai trung bình bình phương (năng lượng hiệu dụng của tín hiệu)
RT60	Reverberation Time 60 dB	Thời gian để năng lượng âm vang suy giảm 60 dB sau khi ngắt nguồn âm
SMPTE	Society of Motion Picture and Television Engineers	Hiệp hội Kỹ sư Điện ảnh và Truyền hình Quốc tế
SNR	Signal-to-Noise Ratio	Tỷ số tín hiệu trên tạp âm (đo bằng dB)

CHƯƠNG 3
SỐ HÓA ÂM THANH

Chương 3 trình bày hai phép rời rạc hóa nền tảng của toàn bộ kỹ thuật audio số là lấy mẫu theo trục thời gian và lượng tử hóa theo trục biên độ, từ đó đi tới định lý Nyquist–Shannon, hiện tượng chồng phổ và bộ lọc chống răng cưa, quan hệ giữa độ sâu bit với dải động, kỹ thuật dither, thang đo dBFS, hệ thống định dạng tệp âm thanh và công nghệ mã hóa nơ-ron hiện đại.

Mục tiêu của chương

Sau khi đọc xong chương này, người đọc có thể:

1. **Về kiến thức lý thuyết:** Trình bày bản chất của số hóa âm thanh gồm hai phép rời rạc hóa là lấy mẫu và lượng tử hóa; phát biểu định lý Nyquist–Shannon, giải thích hiện tượng chồng phổ (aliasing) và vai trò bắt buộc của bộ lọc chống răng cưa đặt trước ADC; phân tích quan hệ giữa độ sâu bit, nhiễu lượng tử và dải động; giải thích kỹ thuật dither và thang đo dBFS; phân loại ba nhóm định dạng tệp âm thanh (không nén, nén không tổn hao, nén có tổn hao); mô tả nguyên lý của codec âm thanh nơ-ron.
2. **Về kỹ năng chuyên môn:** Tính được tần số Nyquist, tần số giả mạo, dải động và dung lượng tệp âm thanh không nén; lập trình Python để tái hiện hiện tượng aliasing, đọc siêu dữ liệu tệp và chuyển đổi độ sâu bit; sử dụng Spek hoặc Audacity để nhận diện tệp "Fake Lossless" qua vạch cắt trên phổ đồ.
3. **Về thói quen làm việc:** Tuân thủ quy tắc xử lý trên số thực float32 và chỉ lượng tử hóa ở bước ghi tệp cuối cùng; chọn tần số lấy mẫu, độ sâu bit và định dạng tệp phù hợp với từng khâu sản xuất; kiểm tra chất lượng thực của tệp nguồn trước khi đưa vào chuỗi sản xuất.

Ký hiệu và chữ viết tắt dùng trong chương

Bảng 3.1 Ký hiệu và viết tắt dùng trong Chương 3

Ký hiệu / Viết tắt	Chữ tiếng Anh nguyên bản	Ý nghĩa / Định nghĩa kỹ thuật trong âm thanh số
ADC	Analog-to-Digital Converter	Bộ chuyển đổi tín hiệu tương tự sang tín hiệu số
DAC	Digital-to-Analog Converter	Bộ chuyển đổi tín hiệu số sang tín hiệu tương tự
PCM	Pulse-Code Modulation	Điều chế mã xung — phương pháp số hóa âm thanh không nén chuẩn

Ký hiệu / Viết tắt	Chữ tiếng Anh nguyên bản	Ý nghĩa / Định nghĩa kỹ thuật trong âm thanh số
fs	Sampling Frequency	Tần số lấy mẫu: Số mẫu đo được trong 1 giây (đơn vị Hertz)
dBFS	Decibels Relative to Full Scale	Mức decibel so với mức trần biên độ lớn nhất có thể biểu diễn
SQNR	Signal-to-Quantization-Noise Ratio	Tỷ số tín hiệu trên nhiễu lượng tử (đo bằng dB)
WAV	Waveform Audio File Format	Định dạng tệp âm thanh không nén tiêu chuẩn trên Windows
FLAC	Free Lossless Audio Codec	Chuẩn mã hóa nén âm thanh hoàn toàn không tổn hao
RVQ	Residual Vector Quantization	Lượng tử hóa vector thặng dư dùng trong nén âm thanh AI
LLM	Large Language Model	Mô hình ngôn ngữ lớn — cầu nối ứng dụng của neural codec

3.1 Lấy mẫu và định lý Nyquist – Shannon

Ở Chương 2, hành trình của tín hiệu analog đã được theo dõi đến ngay trước khối chuyển đổi tương tự - số (ADC) của audio interface. Số hóa (digitization) là quá trình biến một tín hiệu tương tự liên tục theo thời gian và liên tục theo biên độ thành một dãy các con số hữu hạn rời rạc mà máy tính có thể lưu trữ, đọc ghi và thực hiện các phép xử lý toán học. Quá trình này được tạo thành bởi hai phép rời rạc hóa độc lập và nối tiếp nhau: rời rạc hóa trục thời gian (gọi là lấy mẫu) và rời rạc hóa trục biên độ (gọi là lượng tử hóa) [2], [6].

3.1.1 Rời rạc hóa trục thời gian

Lấy mẫu (sampling) là phép đo các giá trị biên độ tức thời của tín hiệu tương tự tại các thời điểm cách đều nhau trong không gian thời gian. Số lượng phép đo thực hiện được trong thời gian một giây được gọi là tần số lấy mẫu (sampling rate), ký hiệu là fs, đo bằng Hertz (Hz) hoặc số mẫu/giây (samples/second), còn khoảng thời gian không đổi giữa hai lần lấy mẫu liên tiếp là chu kỳ lấy mẫu, ký hiệu T và được xác định bằng công thức $T = 1/fs$.

Hình 3.1 cho thấy mặt trước của một audio interface thương mại với các ngõ vào micro/nhạc cụ, núm khuếch đại (gain) và núm điều chỉnh mức nghe kiểm tra; bên trong thiết bị này là các khối chuyển đổi tương tự – số thực hiện phép lấy mẫu mô tả ở mục này.

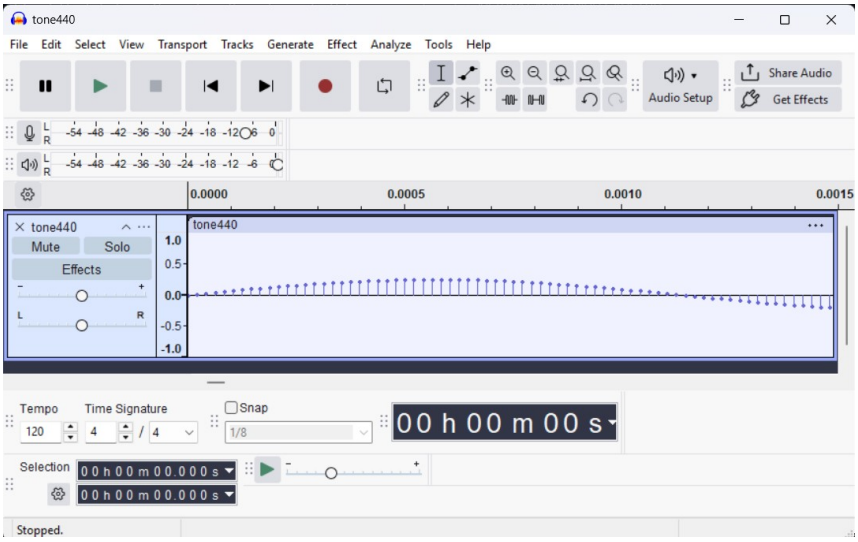


Hình 3.1 Mặt trước audio interface 18 ngõ vào/8 ngõ ra (Focusrite Scarlett 18i8), nơi tín hiệu micro được khuếch đại rồi đưa vào khối ADC

Nguồn: ParsecAU, CC0, Wikimedia Commons (đã cắt) (truy cập 05/11/2026).

Kết quả của phép lấy mẫu là một dãy các giá trị biên độ rời rạc theo thời gian p(nT) với n = 0, 1, 2, 3... Dãy giá trị này chính là mảng một chiều biểu diễn tín hiệu âm thanh trong bộ nhớ máy tính, được lập trình xử lý trực tiếp.

Hình 3.2 là dạng sóng của một âm sin 440 Hz được phóng to tối đa trong Audacity: mỗi chấm trên que thẳng đứng là một mẫu rời rạc, hai mẫu liền kề cách nhau 1/44100 s ≈ 22,7 µs.



Hình 3.2 Dạng sóng âm sin 440 Hz phóng đại tối đa trong Audacity: mỗi chấm là một mẫu, trục thời gian tính bằng giây

Nguồn: ảnh chụp màn hình do người soạn thực hiện (Audacity 3.7.7), 07/11/2026.

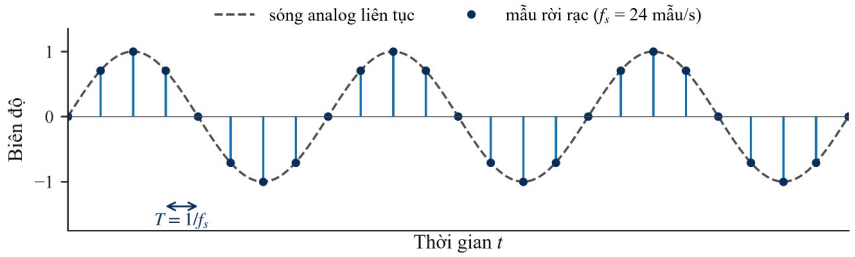
CẦU NỐI LẬP TRÌNH 3.1 — TẦN SỐ LẤY MẪU fs LÀ THAM SỐ GỐC BẮT BUỘC

Trong Python, mọi thao tác đọc/ghi tệp hoặc xử lý tín hiệu số đều yêu cầu hoặc trả về tham số fs đi kèm với mảng mẫu âm thanh, bởi một mảng số p thuần túy hoàn toàn vô nghĩa nếu không biết khoảng thời gian chu kỳ T giữa hai phần tử kề nhau.

Cú pháp đọc file WAV chuẩn bằng thư viện soundfile:

```
x, fs = sf.read('giong_noi.wav') # x: mảng NumPy một chiều, fs: tần số lấy mẫu
```

Nếu giá trị fs bị thay đổi tùy tiện khi lưu tệp (ví dụ: một mảng âm thanh được thu ở 48 kHz nhưng khi ghi ra tệp lại gán fs = 24 kHz), tốc độ phát sẽ chậm đi một nửa và cao độ (pitch) bị kéo trầm xuống đúng một quãng tám (octave). Vì vậy, fs và mảng giá trị luôn là cặp không thể tách rời trong lập trình âm thanh.



Hình 3.3 Rời rạc hóa trục thời gian: sóng analog liên tục được đo tại các mốc thời gian cách đều nhau

3.1.2 Định lý lấy mẫu Nyquist – Shannon

Khi rời rạc hóa trục thời gian, một câu hỏi khoa học cốt lõi được đặt ra: tần số lấy mẫu fs phải là bao nhiêu lần một giây để không bị thất thoát thông tin và tín hiệu có thể khôi phục hoàn toàn chính xác sang dạng tương tự (tái lập hoàn hảo).

Định lý lấy mẫu Nyquist–Shannon (Whittaker-Shannon-Nyquist theorem) đưa ra câu trả lời và điều kiện đủ dứt khoát: để tái lập hoàn hảo một tín hiệu liên tục có tần số giới hạn từ phiên bản mẫu rời rạc của nó, tần số lấy mẫu fs của hệ thống số phải lớn hơn ít nhất gấp hai lần tần số cao nhất f_{max} có trong tín hiệu gốc ban đầu [1]:

$$f_s > 2f_{\max} \quad ; \quad f_{\text{Nyquist}} = \frac{f_s}{2} \tag{3.1}$$

Tần số cực đại f_N = fs/2 được định nghĩa là tần số Nyquist của hệ thống lấy mẫu, tức là giới hạn tần số cao nhất mà hệ thống số có thể biểu diễn trung thực và không bị sai lệch phổ.

Trực giác hình học: Để ghi lại thông tin biến thiên của một chu kỳ sóng sin (một dao động lên và xuống), cần tối thiểu hai điểm mẫu: một điểm ở nửa chu kỳ dương (đỉnh) và một điểm ở nửa chu kỳ âm (đáy), nhờ đó mới xác định được sóng có tồn tại dao động. Nếu tần số lấy mẫu thấp hơn mức Nyquist, các điểm mẫu rơi vào những vị trí sai lệch và mô tả sai thành một sóng hình sin khác có tần số thấp hơn nhiều.

Ứng dụng thực tế chuẩn CD: Tai người có dải nghe tối đa tới f_{max} = 20.000 Hz (20 kHz), do đó tần số lấy mẫu lý thuyết của hệ thống số phải đạt tối thiểu fs > 40.000 Hz. Chuẩn đĩa Compact Disc (CD) chọn tần số lấy mẫu fs = 44.100 Hz (44.1 kHz) để đáp ứng trọn vẹn dải nghe của người, đồng thời chừa một khoảng đệm an toàn 4.100 Hz cho vùng chuyển tiếp của các bộ lọc thông thấp tương tự trong thiết bị.

3.1.3 Hiện tượng chồng phổ và bộ lọc chống răng cưa

Khi tín hiệu tương tự đưa vào chứa các thành phần tần số vượt quá tần số giới hạn Nyquist (f > fs/2), phép lấy mẫu rời rạc không còn đủ số điểm mẫu để mô tả đúng hình dáng sóng. Lúc này, thành phần tần số cao đó bị 'gấp' (hoặc chồng chéo) qua ranh giới Nyquist và hiện diện dưới dạng một tần số thấp giả mạo trong mảng số; hiện tượng vật lý này gọi là aliasing (hiện tượng chồng phổ hay méo phổ răng cưa) [1], [3].

Tần số giả mạo f_{alias} xuất hiện trong hệ thống sau khi lấy mẫu có thể được tính định lượng bằng khoảng cách gấp đối xứng quanh dải Nyquist. Ví dụ, nếu một tín hiệu chứa sóng sin tần số f = 5.000 Hz được đưa vào một hệ thống số lấy mẫu với tần số fs = 8.000 Hz (tần số Nyquist tương ứng là f_N = 4.000 Hz), do 5.000 Hz > 4.000 Hz nên hiện tượng aliasing sẽ xảy ra và biến tần số này thành tần số giả mạo có độ lớn:

$$f_{\text{alias}} = |f_s - f| \quad \text{(Hz)} \tag{3.2}$$

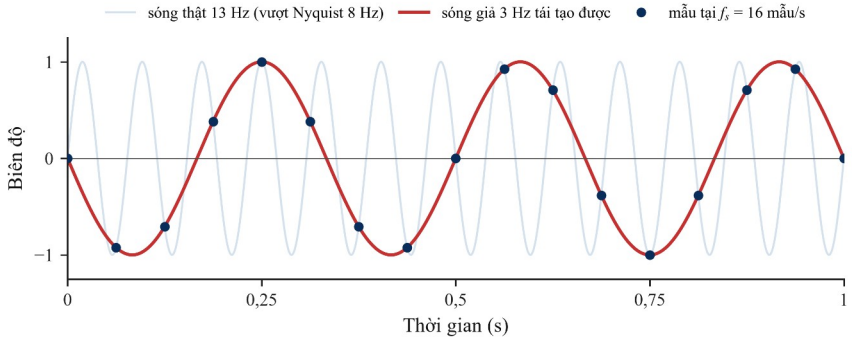
VÍ DỤ 3.1 — HIỆU ỨNG BÁNH XE QUAY NGƯỢC KHI GHI HÌNH

Hiện tượng aliasing không chỉ xảy ra trong miền âm thanh mà còn xuất hiện phổ biến trong kỹ thuật ghi hình. Khi xem video quay một chiếc ô tô bắt đầu tăng tốc chạy tới, có những khoảnh khắc người xem thấy các nan hoa của bánh xe dường như quay chậm dần rồi quay giật lùi ngược lại hướng chuyển động thực tế.

Nguyên nhân là máy quay phim thực hiện việc 'lấy mẫu' hình ảnh liên tục với tần số khung hình chuẩn là 24 hình/giây. Khi bánh xe quay với tốc độ quá nhanh (vượt quá tần số Nyquist của tốc độ lấy mẫu khung hình), các nan hoa di chuyển qua các vị trí góc lớn hơn góc phân giải cho phép, nên các điểm ảnh thu được bị 'gấp' ngược hướng và tạo ra ảo giác chuyển động chậm quay ngược chiều. Đây chính là bản chất của aliasing trong miền không gian - thời gian trực quan.

Giải pháp triệt tiêu aliasing: Một khi tín hiệu đã bị lấy mẫu và xảy ra aliasing, tần số giả mạo trộn lẫn và trùng khít lên các tần số thật trong mảng số, khiến việc bóc tách hay sửa chữa bằng phần mềm sau đó hoàn toàn bất khả thi. Do đó, kỹ thuật phải chặn aliasing ngay từ môi trường analog trước khi lấy mẫu.

Một bộ lọc thông thấp chống răng cưa (anti-aliasing filter) có tần số cắt đặt đúng tại tần số Nyquist (fs/2) được tích hợp ngay trước khối chuyển đổi ADC. Bộ lọc này có nhiệm vụ cắt bỏ mọi thành phần tần số cao vượt ngưỡng Nyquist, bảo đảm tín hiệu đi vào lấy mẫu không còn các tần số có nguy cơ gây răng cưa phổ.

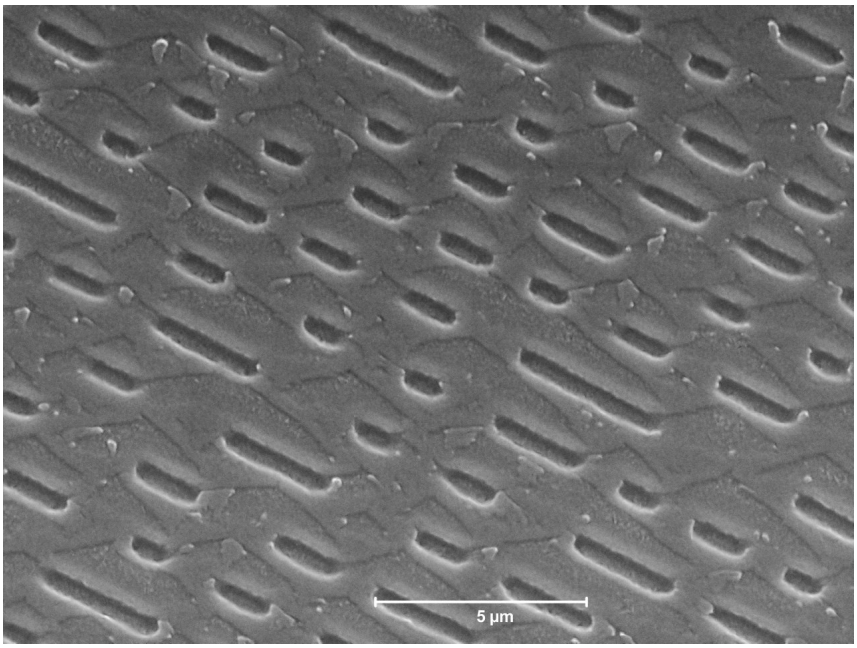


Hình 3.4 Hiện tượng chồng phổ: sóng tần số cao vượt ngưỡng Nyquist bị lấy mẫu thưa và biến thành sóng giả mạo tần số thấp

3.1.4 Các tần số lấy mẫu tiêu chuẩn

Trong kỹ thuật đa phương tiện, các tần số lấy mẫu được tiêu chuẩn hóa chặt chẽ để tương thích giữa các hệ thống lưu trữ, phân phối và thiết bị xử lý:

Hình 3.5 là ảnh hiển vi điện tử quét các rãnh lõm (pit) được ép trên lớp polycarbonate của đĩa compact, trong đó chuỗi bit biểu diễn các mẫu âm thanh được mã hóa bằng chiều dài và vị trí của các rãnh này.



Hình 3.5 Ảnh hiển vi điện tử quét các rãnh dữ liệu trên bề mặt đĩa compact; vạch chuẩn dài 5 µm

Nguồn: [Tycho] (shansov.net), CC0, Wikimedia Commons (truy cập 05/11/2026).

Bảng 3.2 Các tần số lấy mẫu chuẩn và ứng dụng điển hình

Tần số lấy mẫu (fs)	Tần số Nyquist (fs/2)	Phạm vi ứng dụng thực tế
44.100 Hz (44.1 kHz)	22.050 Hz	Tiêu chuẩn đĩa CD, nhạc số phổ thông và phân phối trực tuyến
48.000 Hz (48.0 kHz)	24.000 Hz	Tiêu chuẩn video, truyền hình kỹ thuật số, âm thanh điện ảnh
96.000 Hz (96.0 kHz)	48.000 Hz	Sản xuất âm thanh phòng thu chuyên nghiệp, hậu kỳ độ phân giải cao
192.000 Hz (192 kHz)	96.000 Hz	Thu âm gốc, thiết kế âm thanh chuyên sâu, lưu trữ gốc

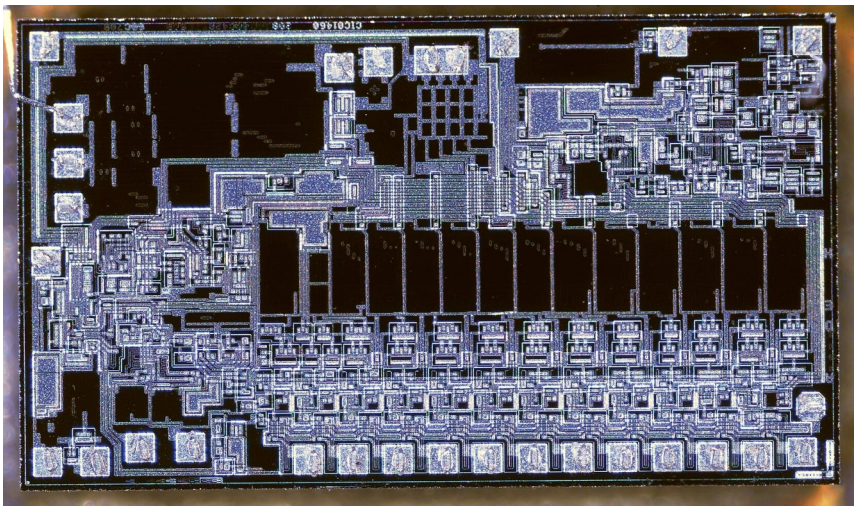
Lưu ý đối với người lập trình: Việc tăng tần số lấy mẫu lên các mức siêu cao (96 kHz, 192 kHz) không nhằm làm cho âm thanh 'hay hơn' đối với tai người, vì 20 kHz là giới hạn sinh lý của thính giác. Lợi ích chính là chừa ra một khoảng băng thông rộng cho các thuật toán xử lý tín hiệu số DSP (Digital Signal Processing) như dịch chuyển cao độ (pitch shifting) và kéo giãn thời gian (time stretching) mà không phát sinh méo phi tuyến, đồng thời đẩy nhiễu lượng tử lên vùng dải tần siêu âm.

3.2 Lượng tử hóa và độ sâu bit

3.2.1 Rời rạc hóa trục biên độ

Sau khi thực hiện phép lấy mẫu rời rạc hóa trục thời gian, các giá trị biên độ đo được tại mỗi mốc nT vẫn tồn tại dưới dạng một số thực liên tục có vô hạn chữ số thập phân sau dấu phẩy (ví dụ: a = 0.548172938491... Pa). Bộ nhớ thanh ghi và kiến trúc tính toán của máy tính số có giới hạn hữu hạn nên không thể lưu giữ các số thực vô hạn chữ số này.

Hình 3.6 minh họa cấu trúc vật lý của một khuôn bán dẫn thuộc bộ chuyển đổi tương tự – số 12 bit, thiết bị thực hiện phép lượng tử hóa biên độ thành các mã nhị phân rời rạc.



Hình 3.6 Khuôn bán dẫn của bộ chuyển đổi ADC 12 bit Analog Devices AD574 (ảnh chụp dưới kính hiển vi)

Nguồn: Lhfjames, CC BY-SA 4.0, Wikimedia Commons (đã cắt) (truy cập 05/11/2026).

Lượng tử hóa (quantization) là phép làm tròn mỗi giá trị mẫu thực tế về một trong số hữu hạn các mức biên độ được thiết lập sẵn của hệ thống. Số lượng các mức biên độ này được quyết định bởi độ sâu bit (bit depth), ký hiệu là n. Với một hệ thống số hóa có độ sâu bit là n, tổng số mức lượng tử hóa biểu diễn được xác định bằng công thức lũy thừa:

$$M = 2^n \tag{3.3}$$

Ví dụ: Một hệ thống lượng tử hóa 16-bit tiêu chuẩn cung cấp $2^{16} = 65.536$ mức làm tròn biên độ, còn một hệ thống hậu kỳ chuyên nghiệp 24-bit cung cấp $2^{24} = 16.777.216$ mức (hơn 16 triệu mức làm tròn rất tinh vi). Càng nhiều mức làm tròn, khoảng cách giữa hai mức kề nhau (bước lượng tử) càng nhỏ, nhờ đó các mẫu được làm tròn sát với giá trị thực tế của sóng analog ban đầu.

3.2.2 Nhiễu lượng tử và quan hệ tuyến tính với dải động

Sự chênh lệch biên độ giữa giá trị tương tự thực tế và giá trị đã được làm tròn sau lượng tử hóa được gọi là sai số lượng tử hay nhiễu lượng tử (quantization noise). Sai số này tồn tại thường trực trong mảng số dưới dạng một nền nhiễu ổn định, nghe như tiếng rào rạt.

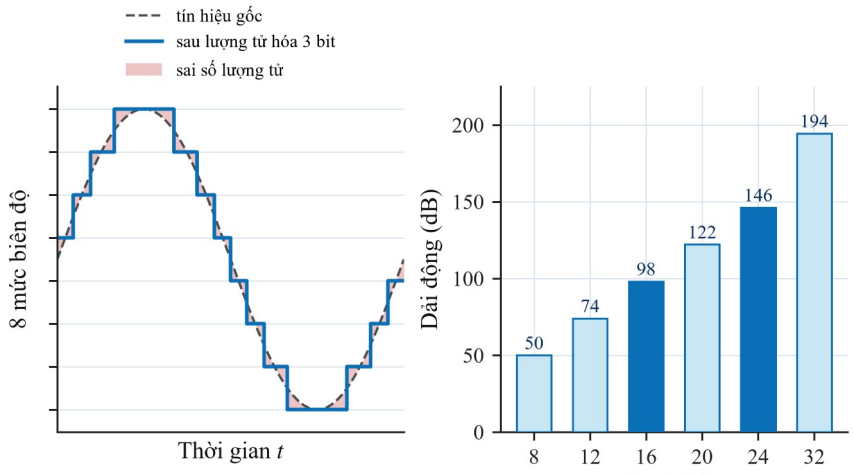
Mỗi khi tăng thêm 1 bit trong độ sâu lượng tử, số lượng mức biểu diễn tăng lên gấp đôi, đồng nghĩa với việc bước lượng tử bị chia đôi và biên độ của nhiễu lượng tử giảm đi một nửa (giảm tương ứng ~6 dB). Mối quan hệ toán học gần đúng giữa độ sâu bit n và tỷ số tín hiệu trên nhiễu lượng tử cực đại (đại diện cho dải động - Dynamic Range) của định dạng được xác định bởi công thức kinh điển [2]:

$$\mathrm{DR} \approx 6{,}02\,n + 1{,}76 \quad \text{(dB)} \tag{3.4}$$

Bảng 3.3 So sánh độ sâu bit và dải động kỹ thuật

Độ sâu bit	Số mức lượng tử hóa	Dải động lý thuyết	Phạm vi ứng dụng thực tiễn
16-bit	65.536	~96 dB	Đĩa CD, xuất bản nhạc số thương mại phổ thông
24-bit	16.777.216	~144 dB	Thu âm chuyên nghiệp, hậu kỳ âm thanh phòng thu
32-bit float	Dấu phẩy động (24-bit mantissa)	trên 1.500 dB (vô cực)	Xử lý nội bộ trong các phần mềm DAW, thu âm dải rộng

Ý nghĩa thực tiễn: Dải động 96 dB của chuẩn 16-bit đủ phủ kín dải động của hầu hết các thể loại âm nhạc phổ thông. Tuy nhiên, trong sản xuất âm thanh, dải động 144 dB của 24-bit được ưu tiên vì cung cấp khoảng biên độ an toàn (headroom) rất rộng, giúp kỹ sư thu âm căn chỉnh gain thuận lợi mà không lo tín hiệu bị clip gây méo.



(a) lượng tử hóa trục biên độ (b) mỗi bit thêm 6,02 dB dải động

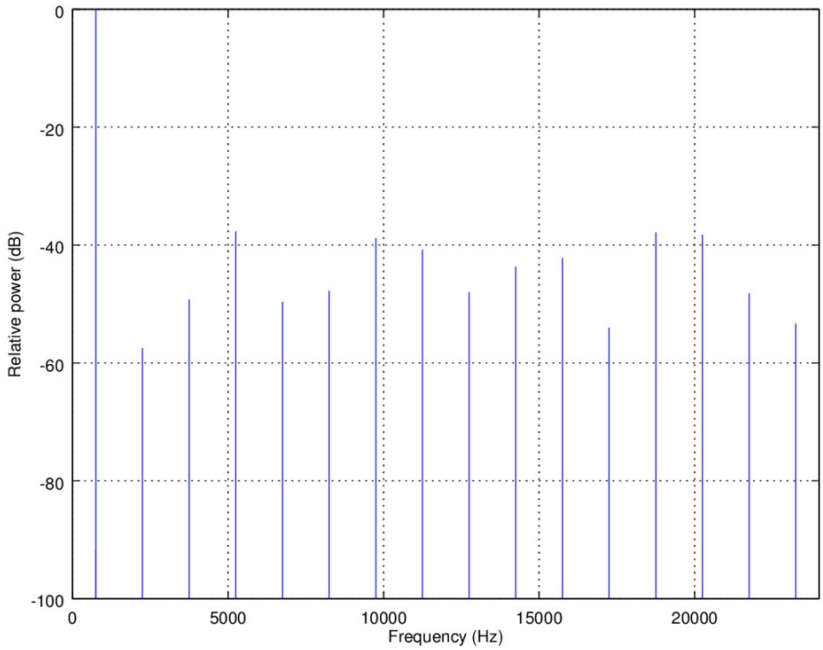
Hình 3.7 Lượng tử hóa trục biên độ, sai số lượng tử và quan hệ tuyến tính giữa độ sâu bit với dải động

3.2.3 Kỹ thuật dither

Khi giảm độ sâu bit của tệp âm thanh (ví dụ: chuyển từ tệp master phòng thu 24-bit về định dạng 16-bit để ghi đĩa CD), các giá trị biên độ nhỏ ở đuôi âm vang hoặc tiếng thì thầm sẽ bị làm tròn thô. Sai lệch lượng tử lúc này không còn ngẫu nhiên mà có tính chu kỳ, liên quan trực tiếp đến tín hiệu gốc (gây méo hài lượng tử - harmonic distortion), tạo ra một lớp tiếng sần khó chịu.

Hình 3.8 trình bày phổ của một âm thuần 750 Hz lấy mẫu ở 48 kHz và lượng tử hóa xuống 4 bit khi không dùng dither: sai số lượng tử không phân bố

đều như nhiễu mà tập trung thành các vạch méo hài, đúng là hiện tượng mà kỹ thuật dither nhằm khắc phục.



Hình 3.8 Phổ của âm thuần 750 Hz sau khi lượng tử hóa 4 bit, không dither, không định dạng nhiễu: ngoài vạch gốc còn nhiều vạch méo

Nguồn: Andrew D'Addesio, CC0, Wikimedia Commons (truy cập 05/11/2026).

Kỹ thuật dither giải quyết triệt để lỗi này bằng cách chủ động trộn thêm một lượng nhiễu ngẫu nhiên rất nhỏ (thường là nhiễu trắng dạng tam giác TPDF [10] có biên độ bằng đúng 1/2 bước lượng tử) vào tín hiệu âm thanh ngay trước khi thực hiện phép làm tròn hạ bit. Thao tác này ngẫu nhiên hóa sai số lượng tử, biến méo hài có hại thành một lớp nhiễu nền đều đặn, tự nhiên và dễ chịu đối với thính giác, nên đây là bước bắt buộc trong khâu xuất bản (mastering) cuối cùng của mọi chuỗi sản xuất đa phương tiện.

CẦU NỐI LẬP TRÌNH 3.2 — LƯỢNG TỬ HÓA CHÍNH LÀ PHÉP CHUYỂN ĐỔI float ↔ int

Trong máy tính, các thuật toán xử lý âm thanh luôn được thực hiện dưới dạng số thực float32 trong dải [-1.0, +1.0] để bảo toàn độ chính xác cao nhất và tránh tràn số giữa chừng, còn lượng tử hóa thực chất là phép nhân tỷ lệ và làm tròn kiểu dữ liệu từ float32 sang số nguyên int16 (16-bit) khi xuất file.

Phép biến đổi toán học lượng tử hóa 16-bit trong Python:

```
# float32 trong dải [-1.0, 1.0] -> int16 trong dải [-32768, 32767]
xi = (np.clip(xf, -1.0, 1.0) * 32767.0).astype(np.int16)
`# Ngược lại: Đọc file int16 chuyển đổi sang số thực float32 để xử lý`
xf = xi.astype(np.float32) / 32767.0
```

Từ bản chất này, lập trình viên cần tuân thủ quy tắc sau: chỉ thực hiện lượng tử hóa hạ bit ở bước ghi file cuối cùng, còn toàn bộ quá trình tính toán trung gian phải chạy trên float32.

3.3 Thang dBFS và biểu diễn mẫu trong máy tính

3.3.1 Thang decibel toàn dải dBFS

Trong âm thanh số, mức tín hiệu của mảng số trong máy tính được biểu diễn bằng thang đo decibel Full Scale (dBFS). Điểm khác biệt cốt lõi của dBFS so với thang đo áp suất âm dBSPL ngoài đời sống là mức cực đại mà hệ thống số biểu diễn được (tương ứng với biên độ tuyệt đối bằng |1.0| trong số thực) được quy ước là mốc trần 0 dBFS [2].

Do đó, mọi mức tín hiệu hợp lệ trong âm thanh số luôn mang giá trị âm (nhỏ hơn 0 dBFS). Công thức hai chiều quy đổi giữa biên độ số thực a và giá trị decibel dBFS được định nghĩa như sau:

$$L_{\text{dBFS}} = 20\log_{10}\frac{|x|}{x_{\max}} \quad \text{(dBFS)} \tag{3.5}$$

Ví dụ: Biên độ mẫu a = 1,0 tương ứng 0 dBFS; biên độ giảm một nửa a = 0,5 tương ứng -6 dBFS; biên độ a = 0,25 tương ứng -12 dBFS; và biên độ mẫu a = 0,125 tương ứng -18 dBFS. Mức -18 dBFS (hoặc -20 dBFS) thường được chọn làm mức thu âm tham chiếu an toàn, vì tạo ra một khoảng đệm headroom chống méo cắt đỉnh.

3.3.2 Biểu diễn mảng dữ liệu trong bộ nhớ

Dữ liệu âm thanh số khi được nạp vào bộ nhớ máy tính để lập trình được tổ chức theo hai đặc tính kỹ thuật cốt lõi:

Kiểu dữ liệu phần tử (Data Type):

- **Số nguyên (int16, int24):** Mỗi mẫu là một phần tử số nguyên nằm trong giới hạn dải của kiểu (int16: -32.768 đến 32.767), có ưu điểm dung lượng gọn nhẹ nên được dùng tối ưu khi ghi file lưu trữ hoặc truyền dẫn.
- **Số thực (float32, float64):** Mỗi mẫu là một số thực nằm trong dải [-1.0, +1.0]. Đây là kiểu dữ liệu bắt buộc khi viết thuật toán xử lý DSP, vì nó loại bỏ hoàn toàn hiện tượng méo do sai số tích lũy của phép cộng/phép nhân và cho phép vượt quá ±1.0 tạm thời trong các tính toán trung gian mà không bị clipping tức thì.

Cấu trúc số kênh (Channels layout):

- **Âm thanh đơn kênh (Mono):** Được biểu diễn bằng một mảng NumPy một chiều (1D Array) gồm N phần tử mẫu nối tiếp nhau.
- **Âm thanh lập thể (Stereo):** Được biểu diễn bằng một mảng hai chiều (2D Array) có kích thước [N mẫu × 2 kênh]; theo quy ước lập trình quốc tế, cột chỉ số 0 chứa mảng mẫu của kênh bên Trái (Left channel) và cột chỉ số 1 chứa mảng mẫu của kênh bên Phải (Right channel).
- **Đa kênh (5.1, 7.1, Dolby Atmos):** Được biểu diễn mở rộng bằng mảng [N mẫu × K kênh] tương ứng với các cột cho Center, Subwoofer, Surround L/R...

QUY TẮC 3.1 — BẤT BIẾN KHI LẬP TRÌNH XỬ LÝ ÂM THANH

Bước 1: Đọc tệp âm thanh vào bộ nhớ --> Chuyển đổi định dạng về mảng số thực float32 trong dải chuẩn [-1.0, +1.0].

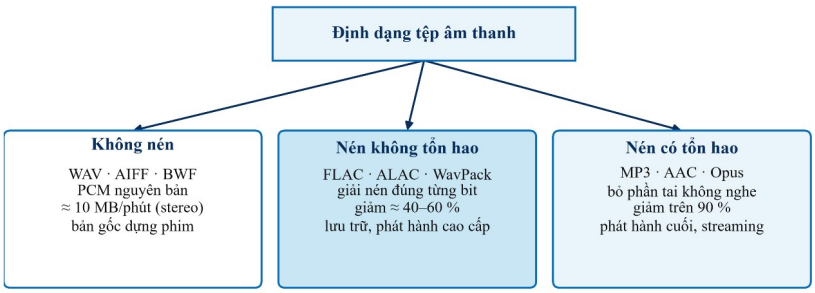
Bước 2: Thực hiện toàn bộ các thuật toán xử lý tín hiệu DSP (EQ, Reverb, Gain, nén, filter) trên mảng float32 này.

Bước 3: Thực hiện chuẩn hóa chống méo cắt đỉnh (amplitude normalization), áp dụng dither làm mịn nếu cần, sau đó mới ép kiểu lượng tử hóa về dạng số nguyên (int16/int24) ở dòng lệnh ghi tệp cuối cùng.

Việc thực hiện các thuật toán tính toán trực tiếp trên mảng số nguyên int16/int24 là một sai lầm nghiêm trọng, vì gây méo tiếng và làm tích lũy sai số làm tròn ở mức thô.

3.4 Định dạng và chuẩn mã hóa

Mảng mẫu âm thanh số sau khi thu nhận có thể được đóng gói và ghi lại dưới dạng nhiều định dạng tệp khác nhau, được phân chia thành ba nhóm chính dựa trên phương thức và thuật toán nén dữ liệu:

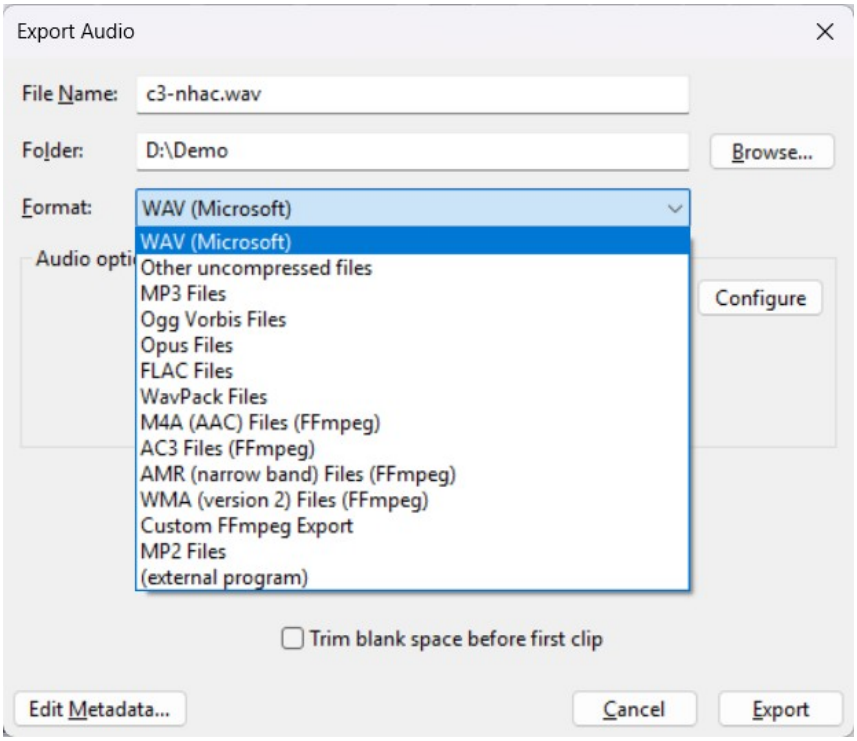


Hình 3.9 Ba nhóm định dạng tệp âm thanh và vị trí sử dụng trong quy trình sản xuất

3.4.1 Nhóm định dạng không nén: PCM/WAV, AIFF

Điều chế mã xung (Pulse-Code Modulation - PCM) là phương pháp lưu trữ trực tiếp, nguyên bản và đơn giản nhất toàn bộ các điểm mẫu đã lấy mẫu và lượng tử hóa, hoàn toàn không qua thuật toán nén nào. Hai định dạng vỏ tệp (wrapper) phổ biến nhất trên thế giới là WAV (phát triển bởi Microsoft) và AIFF (phát triển bởi Apple) [2].

Hình 3.10 là hộp thoại Export Audio của Audacity với danh sách định dạng đầu ra (WAV, MP3, FLAC, Ogg Vorbis…); với WAV, người dùng còn chọn số kênh, tần số lấy mẫu và độ sâu bit (Encoding) của dữ liệu PCM.



Hình 3.10 Hộp thoại Export Audio của Audacity với danh sách định dạng tệp xuất

Nguồn: ảnh chụp màn hình do người soạn thực hiện (Audacity 3.7.7), 07/11/2026.

- **Ưu điểm:** Chất lượng âm thanh đạt mức nguyên bản, việc đọc ghi đơn giản và không tốn tài nguyên CPU để giải nén.
- **Nhược điểm:** Dung lượng file rất lớn, được tính trực tiếp từ các tham số số hóa gốc theo công thức:

$$S = \frac{f_s \cdot n \cdot n_{\text{ch}} \cdot t}{8} \quad \text{(byte)} \tag{3.6}$$

Ví dụ áp dụng: Tính dung lượng một tệp nhạc Stereo 16-bit, tần số lấy mẫu fs = 44.100 Hz, thời lượng dài 60 giây (1 phút):

Dung lượng = 44.100 × (16 / 8) × 2 × 60 = 44.100 × 2 × 2 × 60 = 10.584.000 Bytes ≈ 10,09 MB (chia cho 1.048.576).

3.4.2 Nhóm nén không tổn hao: FLAC, ALAC

Định dạng nén không tổn hao sử dụng các thuật toán nén dữ liệu entropy chuyên dụng cho tín hiệu âm thanh (tương tự nguyên lý nén không mất mát của file ZIP, RAR). Chuẩn FLAC (mã nguồn mở toàn cầu) [12] và ALAC (sử dụng trên hệ sinh thái Apple) là hai đại diện tiêu biểu nhất.

- **Nguyên lý:** Sử dụng thuật toán dự đoán tuyến tính LPC để tìm ra quy luật biến thiên giữa các mẫu, rồi chỉ lưu trữ sai số thặng dư và nén entropy chuỗi số này.
- **Đặc điểm:** Dung lượng file giảm được từ 40% đến 60% so với file WAV gốc, trong khi giải nén khôi phục chính xác từng bit 0 và 1 ban đầu nên chất lượng âm thanh được bảo toàn nguyên vẹn 100%.

3.4.3 Nhóm nén có tổn hao: MP3, AAC, Opus

Để đạt tỷ lệ nén cao (giảm dung lượng file tới hơn 90%), nhóm nén có tổn hao chấp nhận loại bỏ vĩnh viễn những thành phần thông tin âm thanh có năng lượng yếu, vượt quá khả năng cảm nhận sinh lý của thính giác con người.

- **Nguyên lý mô hình tâm lý thính giác (Psychoacoustic Model):** Codec thực hiện phân tích biến đổi phổ Fourier, áp dụng trực tiếp quy luật đường cong đẳng âm Fletcher-Munson và hiện tượng che lấp âm (Auditory Masking). Thành phần âm thanh bị che khuất bởi các âm to lân cận sẽ bị thuật toán loại bỏ hoàn toàn và không được phân bổ bit để mã hóa.
- **MP3 (MPEG-1 Audio Layer 3) [11]:** Ra đời năm 1987, là chuẩn nén phổ biến lâu đời nhất, với nhược điểm là ở bitrate thấp tiếng bị méo rõ ở vùng tần số cao.
- **AAC (Advanced Audio Coding):** Hiệu quả nén vượt trội MP3, hỗ trợ tới 48 kênh tần số cao, và được sử dụng mặc định trên Apple iTunes cũng như các nền tảng streaming hiện đại.
- **Opus:** Chuẩn nén hiện đại nhất (IETF, 2012) [5], kết hợp công nghệ SILK (nén giọng nói) và CELT (nén nhạc), đạt độ trễ rất thấp và hiệu quả nén cao ở mọi dải bitrate.

Bitrate (đo bằng kbps - kilobit/giây) là số lượng bit dữ liệu dùng để biểu diễn một giây âm thanh: bitrate càng cao thì thông tin bị loại bỏ càng ít và chất lượng âm thanh càng tiệm cận file gốc CD.

Bảng 3.4 Bảng tổng hợp đặc tính các nhóm định dạng âm thanh

Nhóm định dạng	Định dạng tiêu biểu	Có tổn hao chất lượng?	Cơ chế hoạt động cốt lõi
Không nén	WAV, AIFF	Không	Lưu trực tiếp mảng số PCM thô, dung lượng cực kỳ lớn
Nén không tổn hao	FLAC, ALAC	Không	Nén entropy toán học, giải nén khôi phục 100% bit gốc
Nén có tổn hao	MP3, AAC, Opus	Có	Loại bỏ dữ liệu theo mô hình che lấp thính giác (masking)

3.4.4 Nhận diện tệp giả danh chất lượng cao

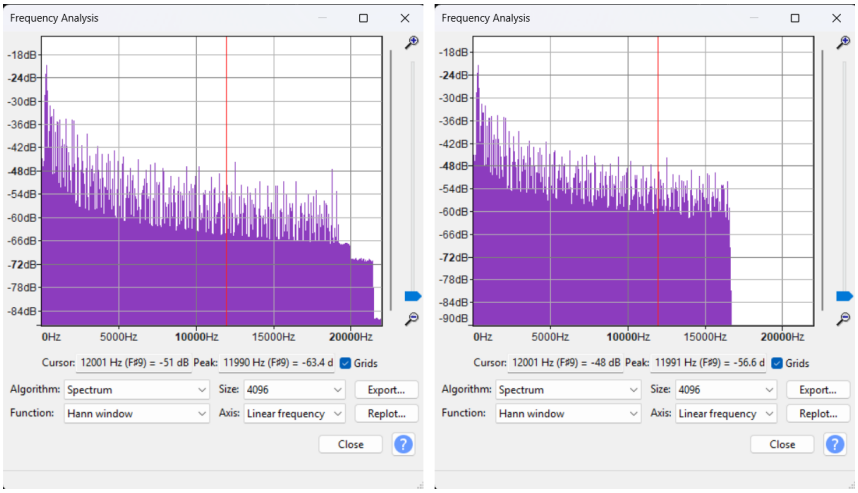
Một tệp có đuôi mở rộng là `.wav` hay `.flac` chưa chắc đã mang chất lượng nguyên bản. Nếu một tệp âm thanh nén có tổn hao chất lượng thấp (như MP3 128 kbps) được đổi đuôi hoặc chuyển đổi định dạng (transcode) ngược lên WAV/FLAC, dữ liệu âm thanh dải cao đã bị codec MP3 xóa bỏ vĩnh viễn từ trước sẽ không thể khôi phục lại.

Sản phẩm thu được chỉ là một tệp 'Fake Lossless': dung lượng tệp tăng rất lớn nhưng chất lượng thực tế bên trong vẫn là chất lượng kém của file lossy ban đầu.

Dấu vết của hiện tượng này nằm ở vạch cắt phổ tần số cao dứt khoát trên đồ thị phổ spectrogram. Codec MP3 ở bitrate 128 kbps cắt bỏ hoàn toàn các dải

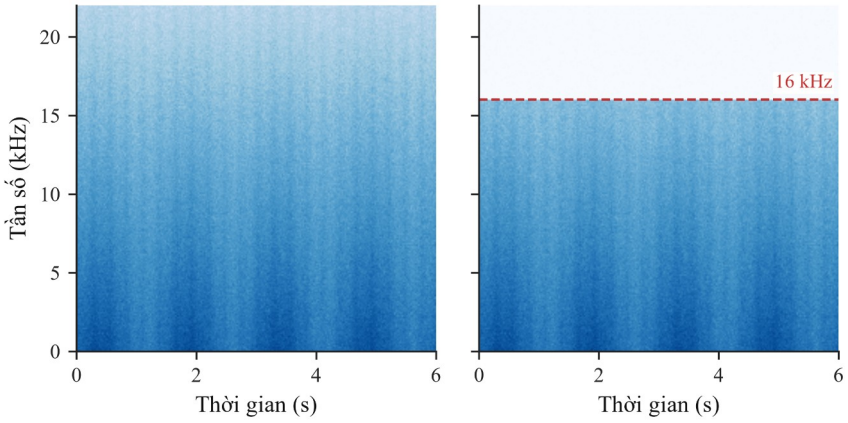
tần số trên 16 kHz, tạo thành một đường ranh giới ngang phẳng và sắc nét (gọi là vạch cắt brick-wall cutoff). Khi mở tệp âm thanh bằng Spek hoặc công cụ Spectrogram của Audacity, lập trình viên có thể phát hiện ngay vạch cắt này để loại các tệp giả danh chất lượng cao khỏi hệ thống xử lý chuyên nghiệp.

Hình 3.11 so sánh phổ của cùng một đoạn nhạc ở dạng WAV 16 bit (trái) và sau khi nén MP3 128 kbps (phải): phổ MP3 bị cắt dứt khoát ở khoảng 16 kHz, vùng phía trên không còn năng lượng.



Hình 3.11 Phổ của cùng đoạn nhạc ở WAV 16 bit (trái) và MP3 128 kbps (phải), trục tần số tuyến tính, cửa sổ Hann, FFT 4096 (Audacity)

Nguồn: ảnh chụp màn hình do người soạn thực hiện (Audacity 3.7.7), 07/11/2026.



(a) tệp WAV/FLAC thật (b) tệp giả danh từ MP3 128 kbit/s

Hình 3.12 Dùng phổ đồ để nhận diện tệp "Fake Lossless": vạch cắt brick-wall là dấu vết không thể chối cãi của nén có tổn hao

3.5 Cập nhật công nghệ: mã hóa âm thanh nơ-ron

Các bộ mã hóa nén truyền thống (MP3, AAC, Opus) đã đạt đến giới hạn phát triển do phụ thuộc hoàn toàn vào các quy tắc do con người thiết lập sẵn (mô hình tâm lý thính giác tĩnh). Từ năm 2021, trí tuệ nhân tạo đã mở ra một hướng phát triển nén hoàn toàn mới là codec âm thanh mạng nơ-ron (Neural Audio Codec) [7], công nghệ tự động học cách nén và giải nén tín hiệu trực tiếp từ kho dữ liệu lớn bằng mạng nơ-ron sâu.

3.5.1 Nguyên lý biến âm thanh thành chuỗi token

Kiến trúc của một Neural Audio Codec bao gồm ba khối chức năng liên hoàn tuần tự:

1. **Bộ mã hóa nơ-ron (Encoder):** Sử dụng các mạng nơ-ron tích chập 1D để nén mảng sóng thô PCM ban đầu thành một biểu diễn đặc trưng ẩn (latent representation) có kích thước rất cô đọng.
2. **Bộ lượng tử hóa vector thặng dư (Residual Vector Quantization - RVQ):** Chia biểu diễn ẩn liên tục thành nhiều cấp độ thặng dư, đối chiếu và làm tròn chúng về các chỉ số (mã số) nằm trong một từ điển mã (Codebook) được học sẵn, nhờ đó đầu ra là một chuỗi các token rời rạc (discrete tokens) tương đương với các mã số nguyên gọn nhẹ.
3. **Bộ giải mã nơ-ron (Decoder):** Nhận chuỗi token rời rạc, đối chiếu từ điển và tổng hợp tái tạo lại dạng sóng âm thanh nguyên thủy bằng mạng nơ-ron tạo sinh đối nghịch (GAN).

Ưu thế nổi bật: Neural codec đạt chất lượng âm thanh tự nhiên ở mức bitrate rất thấp (dưới 3 kbps, tức là giảm 10 lần so với Opus và 100 lần so với WAV thô) bằng cách thay các mức lượng tử hóa biên độ cố định bằng từ điển token tự học.

3.5.2 Các mốc phát triển công nghệ nổi bật

Các mô hình Neural Audio Codec dẫn dắt công nghệ âm thanh AI toàn cầu hiện nay gồm:

Bảng 3.5 Các cột mốc Neural Audio Codec tiên tiến

Mô hình và Hãng phát triển	Năm công bố	Đặc tính kỹ thuật và đóng góp cốt lõi
SoundStream (Google) [7]	2021	Đặt nền móng kiến trúc RVQ đầu tiên; hỗ trợ nén và giải nén thời gian thực
EnCodec (Meta AI) [4]	2022	Mở rộng xuất sắc lên định dạng Stereo 48 kHz dải rộng cho chất lượng âm nhạc gốc
DAC (Descript Audio Codec) [8]	2023	Cải thiện tỷ lệ nén vượt bậc và triệt tiêu hoàn toàn các hiện tượng nhiễu tạo tác
Mimi (Kyutai) [9]	2024	Tối ưu hóa sâu sắc cho mô hình ngôn ngữ âm thanh thời gian thực độ trễ siêu thấp

CẦU NỐI AI 3.1 — CHUỖI TOKEN CHÍNH LÀ NGÔN NGỮ CỦA AI TẠO SINH NHẠC

Chuỗi token rời rạc do RVQ sinh ra không chỉ có vai trò nén dữ liệu mà còn biến sóng âm thanh liên tục thành các 'từ vựng' số nguyên rời rạc.

Nhờ đó, chuỗi token này có thể được đưa vào làm dữ liệu đầu vào cho các mô hình ngôn ngữ lớn (Transformer LLM) tương tự như văn bản. Khi AI tạo sinh nhạc (như Suno, Udio) hoặc AI sinh giọng nói tự nhiên (như ElevenLabs) 'sáng tác' nhạc, thực chất mô hình đang thực hiện thuật toán dự đoán token kế tiếp trong từ điển mã.

Neural Audio Codec vì vậy là cầu nối kỹ thuật liên kết thế giới số hóa cổ điển của bài học này với kỷ nguyên âm thanh AI tạo sinh, nội dung sẽ được lập trình ở các chương về ứng dụng trí tuệ nhân tạo.



Hình 3.13 Kiến trúc bộ mã hóa âm thanh nơ-ron: encoder – lượng tử RVQ – chuỗi token – decoder

3.6 Hướng dẫn thực hành

3.6.1 Tái hiện và quan sát hiện tượng chồng phổ bằng Python

Mục tiêu: Người đọc tự lập trình một script Python để kiểm chứng bằng mắt và tai hiện tượng aliasing. Sóng sin tần số cao 5.000 Hz khi lấy mẫu ở tần số thấp 8.000 Hz (Nyquist 4.000 Hz) sẽ bị dội phổ ngược lại và nghe giống hệt sóng sin chuẩn 3.000 Hz.

Liệt kê 3.1 src/aliasing_demo.py — Tái hiện hiện tượng chồng phổ khi lấy mẫu thấp

```
# -*- coding: utf-8 -*-
import numpy as np
import soundfile as sf

# 1. Thiết lập tham số số hóa dải Nyquist thấp
fs = 8000            # Tần số lấy mẫu thấp (Nyquist = 4000 Hz)
duration = 1.0       # Thời lượng 1 giây
t = np.linspace(0, duration, int(fs * duration), endpoint=False)

# 2. Sinh sóng sin 3000 Hz (Nằm dưới dải Nyquist -> Ghi đúng bản chất)
f_ok = 3000
y_ok = 0.4 * np.sin(2 * np.pi * f_ok * t)
sf.write('ok_3000.wav', y_ok, fs)

# 3. Sinh sóng sin 5000 Hz (Vượt quá Nyquist 4000 Hz -> Bị Aliasing dội ngược)
f_high = 5000
y_alias = 0.4 * np.sin(2 * np.pi * f_high * t)
sf.write('alias_5000.wav', y_alias, fs)

print("--- THỰC THI THÀNH CÔNG DEMO ALIASING ---")
print(f"Tần số lấy mẫu fs = {fs} Hz | Tần số Nyquist = {fs//2} Hz")
```

```
print(f"Sóng 3000 Hz ghi ra file: ok_3000.wav")
print(f"Sóng 5000 Hz ghi ra file: alias_5000.wav (Sẽ bị nghe thành {abs(fs -
f_high)} Hz do Aliasing!)")
```

Thao tác thực hiện: Chạy script trên, mở hai file WAV sinh ra và nghe thử trên tai nghe. Hai tệp có cao độ giống hệt nhau dù một tệp lẽ ra phải cao hơn hẳn. Sau đó mở cả hai file bằng Audacity ở chế độ phổ Spectrogram để quan sát vạch tần số dội ngược nằm tại 3.000 Hz.

3.6.2 Nhận diện dấu vết nén có tổn hao

Mục tiêu: Thực hành kỹ năng phân tích phổ để kiểm chứng chất lượng nguồn âm thanh thực tế.

Các bước thực hiện:

Bước 1: Chuẩn bị một file nhạc WAV gốc chất lượng cao chuẩn CD (44.1 kHz, 16-bit), sau đó dùng Audacity hoặc công cụ FFmpeg để xuất file này sang hai định dạng: MP3 bitrate 320 kbps (chất lượng cao của lossy) và MP3 bitrate 128 kbps (chất lượng thấp phổ thông).

Bước 2: Mở lần lượt ba tệp âm thanh WAV gốc, MP3 320k và MP3 128k bằng phần mềm phân tích phổ Spek.

Bước 3: Quan sát và so sánh đặc tuyến:

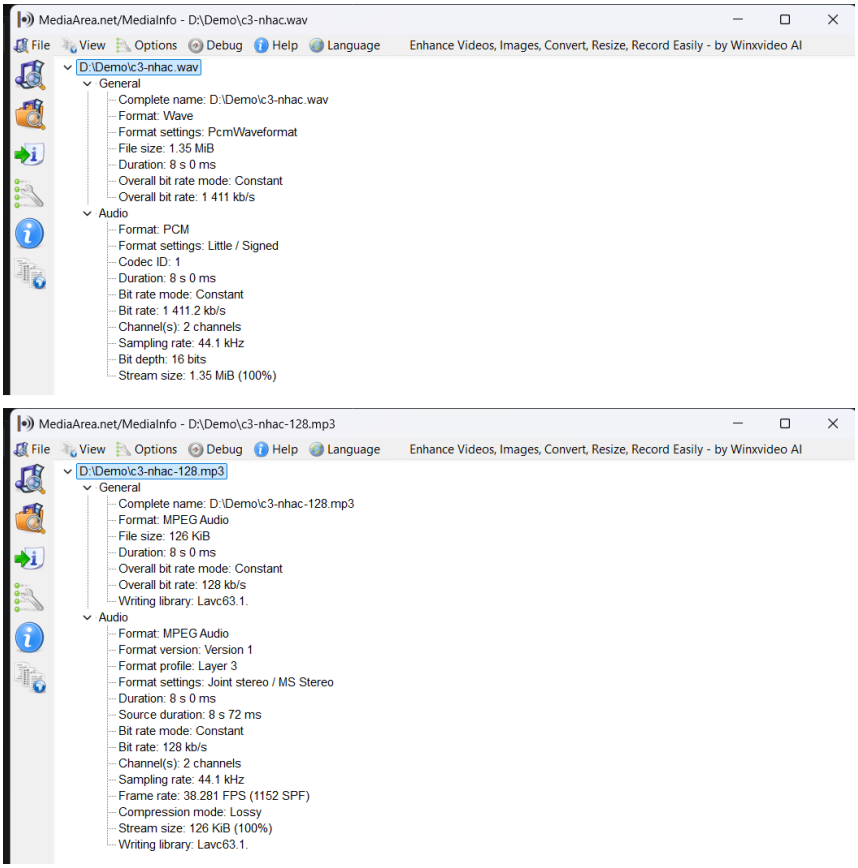
- **WAV gốc:** Năng lượng phổ trải đều và liên tục lên sát dải biên giới thính giác 20 kHz - 22 kHz.
- **MP3 320 kbps:** Năng lượng được bảo toàn tốt, chỉ bị cắt nhẹ và bo tròn ở sát vùng đỉnh 20 kHz.
- **MP3 128 kbps:** Thấy rõ một vạch cắt phẳng và dứt khoát (brick-wall cutoff) ngay tại tần số 16 kHz, và toàn bộ vùng trên 16 kHz tối đen hoàn toàn do bị loại bỏ triệt để.

Bước 4: Sử dụng phần mềm chuyển đổi (transcode) file MP3 128k ngược trở lại định dạng WAV không nén, rồi mở file WAV mới này bằng Spek. Vạch cắt 16 kHz vẫn hiển thị rõ dù đuôi file đã là .wav và dung lượng phình to gấp 10 lần, đây chính là bằng chứng của tệp 'Fake Lossless' đánh lừa người dùng.

3.6.3 Khảo sát thông số và chuyển đổi định dạng bằng Python

Mục tiêu: Nắm vững các tham số số hóa ở mức lập trình, đọc thông tin siêu dữ liệu của tệp, tính toán ước lượng dung lượng và ghi ra các định dạng với các độ sâu bit khác nhau.

Hình 3.14 cho thấy thông số của hai tệp âm thanh được đọc bằng MediaInfo ở chế độ Tree: tệp WAV là PCM 16 bit, 44,1 kHz, 2 kênh, tốc độ bit 1 411 kb/s; tệp MP3 là MPEG Audio Layer 3, nén có tổn hao ở 128 kb/s.



Hình 3.14 Thông số kỹ thuật của tệp WAV (trên) và tệp MP3 128 kbps (dưới) trong MediaInfo, chế độ Tree

Nguồn: ảnh chụp màn hình do người soạn thực hiện (MediaInfo 26.10), 07/11/2026.

Liệt kê 3.2 src/format_tool.py — Khảo sát thông số tệp và chuyển đổi độ sâu bit

```
# -*- coding: utf-8 -*-
import numpy as np
import soundfile as sf

# 1. Đọc tệp âm thanh đầu vào và trích xuất siêu dữ liệu
filename = 'input.wav'
try:
    x, fs = sf.read(filename)
    info = sf.info(filename)
except Exception as e:
    print(f"Lỗi: Hãy chuẩn bị một file {filename} trong thư mục chạy script!")
    # Tự sinh file mẫu để chạy test
    fs = 44100
    t = np.linspace(0, 3.0, fs * 3, endpoint=False)
    x = 0.5 * np.sin(2 * np.pi * 440.0 * t)
    sf.write(filename, x, fs, subtype='PCM_16')
    x, fs = sf.read(filename)
    info = sf.info(filename)

num_channels = 1 if x.ndim == 1 else x.shape[1]
duration = len(x) / fs

print("--- THÔNG SỐ SỐ HÓA CỦA FILE ---")
print(f"Tần số lấy mẫu fs : {fs} Hz")
print(f"Độ sâu bit gốc    : {info.subtype}")  # v.d. PCM_16, PCM_24, FLOAT
print(f"Số kênh âm thanh  : {num_channels} (Mono)" if num_channels==1 else
f"Số kênh: {num_channels} (Stereo)")
```

```
print(f"Thời lượng tệp    : {duration:.2f} giây")
print(f"Tần số Nyquist    : {fs // 2} Hz")

# 2. Ước lượng dung lượng tệp WAV 16-bit lý thuyết theo công thức
bits_calc = 16
size_bytes = fs * (bits_calc // 8) * num_channels * duration
print(f"Dung lượng WAV 16-bit lý thuyết tính toán: {size_bytes / 1e6:.2f} MB")

# 3. Chuẩn hóa về float32 và thực hiện chuyển đổi độ sâu bit lượng tử hóa
if x.dtype != np.float32:
    x_float = x.astype(np.float32)
else:
    x_float = x

# Ghi ra các file với cấu hình lượng tử hóa khác nhau
sf.write('out_PCM16.wav', x_float, fs, subtype='PCM_16') # Lượng tử hóa 16-bit
sf.write('out_PCM24.wav', x_float, fs, subtype='PCM_24') # Lượng tử hóa 24-bit
sf.write('out_compressed.flac', x_float, fs)            # Nén không tổn hao
FLAC

print("\nĐã xuất bản thành công ba file: out_PCM16.wav, out_PCM24.wav và
out_compressed.flac")
print("So sánh dung lượng thực tế trên ổ cứng để thấy rõ mối tương quan.")
```

TÓM TẮT CHƯƠNG 3

1. **Bản chất số hóa:** Gồm hai phép rời rạc hóa độc lập nối tiếp nhau: rời rạc hóa trục thời gian (lấy mẫu) và rời rạc hóa trục biên độ (lượng tử hóa).
2. **Định lý Nyquist-Shannon:** Tần số lấy mẫu fs phải không nhỏ hơn hai lần tần số cực đại của tín hiệu ($fs \geq 2 \cdot f_{max}$). Nếu điều kiện này bị vi phạm, hiện tượng răng cưa aliasing xảy ra và làm các tần số cao bị gập ngược về vùng tần số thấp dưới dạng thành phần giả.
3. **Bộ lọc chống răng cưa (Anti-aliasing filter):** Là bộ lọc thông thấp analog đặt trước ADC để cắt bỏ mọi tần số vượt quá fs/2; khi aliasing đã xuất hiện trong dữ liệu lấy mẫu thì không thể khắc phục bằng phần mềm.
4. **Độ sâu bit n:** Quyết định số lượng mức biên độ biểu diễn được (2^n) và dải động của định dạng số theo công thức dải động $\approx 6{,}02 \cdot n$ dB. Chuẩn 16-bit cho dải động ~96 dB; chuẩn 24-bit cho dải động ~144 dB.
5. **Kỹ thuật dither:** Là việc thêm một lượng nhiễu ngẫu nhiên rất nhỏ trước khi thực hiện phép làm tròn hạ bit (ví dụ từ 24-bit sang 16-bit), nhờ đó méo hài có hại được chuyển thành nhiễu nền ít gây khó chịu đối với thính giác.
6. **Thang đo dBFS:** Thang đo decibel trong máy tính số với mức 0 dBFS là mức trần cực đại. Nguyên tắc lập trình cơ bản là xử lý toàn bộ trên số thực float32 trong dải [-1.0, 1.0] và chỉ hạ bit về số nguyên int ở khâu xuất tệp cuối cùng.
7. **Định dạng tệp:** Chia làm ba nhóm: không nén PCM (WAV, AIFF), nén không tổn hao (FLAC, ALAC) và nén có tổn hao (MP3, AAC, Opus) dựa trên mô hình che lấp thính giác người. Tệp 'fake lossless' có thể bị phát hiện qua vạch cắt phổ brick-wall trên đồ thị spectrogram.

8. **Neural Audio Codec:** Công nghệ nén âm thanh thế hệ mới dựa trên AI (SoundStream, EnCodec, DAC, Mimi), nén dạng sóng thành chuỗi token rời rạc thông qua từ điển mã tự học, và là cầu nối quan trọng tới âm thanh tạo sinh dựa trên mô hình ngôn ngữ lớn (LLM).

3.7 Hệ thống bài tập và câu hỏi ôn tập

3.7.1 Câu hỏi ôn tập lý thuyết

Câu 1: Quá trình số hóa âm thanh tương tự gồm hai phép rời rạc hóa độc lập nào, và mỗi phép tác động lên trục tọa độ nào (thời gian hay biên độ)?

Câu 2: Phát biểu định lý lấy mẫu Nyquist–Shannon và giải thích bằng trực giác hình học vì sao một sóng sin cần tối thiểu 2 điểm mẫu để có thể khôi phục.

Câu 3: Hiện tượng aliasing (chồng phổ) là gì? Giải thích vì sao bộ lọc chống răng cưa phải đặt trước khối ADC mà không thể thay bằng thuật toán phần mềm xử lý sau khi số hóa.

Câu 4: Trình bày mối quan hệ toán học giữa độ sâu bit lượng tử hóa và dải động, đồng thời giải thích vì sao trong sản xuất âm thanh chuyên nghiệp, kỹ sư thường thu âm ở chuẩn 24-bit thay vì 16-bit.

Câu 5: Kỹ thuật dither là gì? Khi nào cần sử dụng dither trong mã nguồn xử lý âm thanh?

Câu 6: Định nghĩa hiện tượng 'Fake Lossless' và mô tả quy trình sử dụng phần mềm Spek cùng dấu hiệu phổ đặc trưng để phát hiện hiện tượng này.

3.7.2 Bài tập vận dụng có lời giải chi tiết

Hệ thống bài tập định lượng giúp người đọc củng cố lý thuyết toán học và ứng dụng thực tế trong lập trình hệ thống âm thanh số:

Bài tập 1: Tính toán dung lượng tệp âm thanh không nén (WAV)

Đề bài: Một bản thu âm nhạc Stereo (2 kênh), tần số lấy mẫu fs = 48.000 Hz, độ sâu bit lượng tử hóa 24-bit, thời lượng đúng 3 phút. Tính dung lượng lý thuyết của tệp âm thanh WAV thu được theo đơn vị Megabyte (MB), rồi so sánh với dung lượng khi nén FLAC (tỷ lệ nén 50%) và nén MP3 320 kbps.

BÀI TẬP 3.1 — HƯỚNG DẪN GIẢI

Bước 1: Quy đổi các tham số về đơn vị chuẩn:

• Độ sâu 24-bit = 24 / 8 = 3 Bytes/mẫu.

• Thời lượng 3 phút = 3 × 60 = 180 giây.

Bước 2: Áp dụng công thức tính dung lượng WAV thô:

Dung lượng = fs × (Số bit / 8) × Số kênh × Thời gian

Dung lượng = 48.000 × 3 × 2 × 180 = 51.840.000 Bytes.

Bước 3: Quy đổi sang đơn vị Megabyte (chia cho 1.048.576):

Dung lượng (MB) = 51.840.000 / 1.048.576 ≈ 49,44 MB.

Bước 4: So sánh các định dạng nén:

• **Nếu nén FLAC (lossless - giảm 50%):** Dung lượng tệp còn khoảng ~24,7 MB.

• Nếu nén MP3 320 kbps (bitrate 320.000 bit/giây): Dung lượng một giây là 320.000 / 8 = 40.000 Bytes.

Tổng dung lượng MP3 = 40.000 × 180 = 7.200.000 Bytes ≈ 6,87 MB.

Nhận xét: Nén lossy giảm kích thước tệp hơn 7 lần so với WAV không nén, trong khi chất lượng nghe vẫn ở mức tốt.

Bài tập 2: Lựa chọn độ sâu bit lượng tử hóa dựa trên dải động thiết bị

Đề bài: (a) Tính dải động lý thuyết tối đa của một định dạng âm thanh số hóa có độ sâu bit n = 20-bit. (b) Một hệ thống thiết bị phần cứng thu âm thực tế có dải động vật lý (sàn nhiễu nội tại của mạch điện) là 100 dB. Kỹ sư lập trình hệ thống nên lựa chọn độ sâu bit lượng tử hóa là 16-bit hay 24-bit cho phần mềm thu âm, và nguyên lý lựa chọn là gì?

BÀI TẬP 3.2 — HƯỚNG DẪN GIẢI

a) Áp dụng công thức thực nghiệm liên hệ giữa độ sâu bit và dải động:

Dải động ≈ 6,02 × n = 6,02 × 20 = 120,4 dB.

Vậy định dạng 20-bit có dải động lý thuyết tối đa là 120,4 dB.

b) Phân tích dải động các chuẩn:

• **Nếu chọn 16-bit:** Dải động tối đa chỉ đạt ~96 dB. Do 96 dB < 100 dB (dải động phần cứng), định dạng số hóa 16-bit trở thành điểm nghẽn (bottleneck), làm hẹp dải động thực tế của phần cứng và suy giảm chất lượng âm thanh thu được.

• **Nếu chọn 24-bit:** Dải động tối đa đạt ~144 dB. Do 144 dB > 100 dB, dải động của định dạng số bao trùm toàn bộ sàn nhiễu vật lý của thiết bị, bảo toàn trọn vẹn 100% chất lượng dải động thực của phần cứng.

Kết luận: Kỹ sư lập trình nên lựa chọn độ sâu bit 24-bit cho phần mềm hệ thống.

Bài tập 3: Xác định hiện tượng aliasing theo tần số

Đề bài: Một tín hiệu tương tự chứa đồng thời ba thành phần tần số đơn âm: f_1 = 1.000 Hz, f_2 = 9.000 Hz, và f_3 = 15.000 Hz. Tín hiệu này được đưa vào một bộ lấy mẫu số hóa với tần số lấy mẫu fs = 16.000 Hz (16 kHz).

a) Xác định tần số Nyquist của hệ thống số hóa này.

b) Thành phần tần số nào an toàn và thành phần nào bị aliasing? Tính các giá trị tần số thu được trong mảng số sau khi lấy mẫu.

BÀI TẬP 3.3 — HƯỚNG DẪN GIẢI

a) Tần số giới hạn Nyquist của hệ thống là: $f_N = fs / 2 = 16.000 / 2 = 8.000$ Hz (8 kHz).

b) Đánh giá và tính toán từng thành phần:

• Thành phần $f_1 = 1.000$ Hz: Do 1.000 Hz < 8.000 Hz (nằm dưới giới hạn Nyquist), thành phần này hoàn toàn an toàn, được lấy mẫu chính xác và khôi phục đúng tần số f_out1 = 1.000 Hz.

• Thành phần $f_2 = 9.000$ Hz: Do 9.000 Hz > 8.000 Hz, hiện tượng aliasing xảy ra và tần số này bị gập về một tần số giả trong dải thấp có giá trị:

f_out2 = | fs - f_2 | = | 16.000 - 9.000 | = 7.000 Hz.

• Thành phần $f_3 = 15.000$ Hz: Do 15.000 Hz > 8.000 Hz, aliasing xảy ra và tần số này bị gập về giá trị:

f_out3 = | fs - f_3 | = | 16.000 - 15.000 | = 1.000 Hz.

Nguy cơ kỹ thuật: Sóng cao tần 15 kHz bị gập thành sóng giả 1.000 Hz, trùng với sóng thật $f_1 = 1.000$ Hz, gây méo tiếng và làm nhiễu loạn thông tin.

Kết luận: Cần đặt một bộ lọc chống răng cưa trước ADC để cắt bỏ hoàn toàn hai dải tần 9 kHz và 15 kHz trước khi lấy mẫu.

Bài tập 4: Quy trình kiểm chứng và giám định 'Fake Lossless'

Đề bài: Người đọc nhận được một tệp âm thanh có tên là `master\_piece.flac` từ một đối tác gửi đến để làm nhạc nền cho phim; đối tác khẳng định đây là tệp chất lượng cao gốc CD. Trình bày quy trình khoa học sử dụng các công cụ phần mềm để xác định đây là tệp chất lượng gốc thật hay là tệp giả danh ('fake lossless') được transcode từ nguồn MP3 128 kbps.

BÀI TẬP 3.4 — HƯỚNG DẪN GIẢI

Bước 1: Nạp tệp `master_piece.flac` vào phần mềm phân tích phổ Spek hoặc chế độ hiển thị Spectrogram của Audacity.

Bước 2: Quan sát dải năng lượng phổ theo trục tần số (trục tung từ 0 đến 22 kHz):

• **Nếu tệp là lossless thật:** Các vệt năng lượng phổ màu sáng (biểu thị cường độ âm thanh) trải đều, biến thiên tự nhiên và kéo dài tới sát vùng giới hạn 20 kHz - 22 kHz.

• **Nếu tệp là giả danh 'Fake Lossless':** Xuất hiện một đường ranh giới ngang, phẳng và rất sắc nét tại tần số 16.000 Hz (16 kHz). Vùng phía trên từ 16 kHz đến 22 kHz tối đen, tức là không có năng lượng.

Bước 3: Nguyên lý kết luận: Đường cắt sắc nét 'brick-wall cutoff' tại 16 kHz là dấu vết không thể xóa bỏ của codec MP3 128 kbps (mô hình che lấp đã chủ động cắt bỏ dải siêu cao để nén tệp). Mặc dù tệp được đóng gói dưới định dạng FLAC chất lượng cao, dữ liệu đã mất ở dải cao không thể khôi phục; do đó có thể kết luận đây là tệp Fake Lossless và cần từ chối đưa vào chuỗi sản xuất.

3.8 Thuật ngữ chuyên ngành Anh – Việt

Bảng tra cứu thuật ngữ chuyên sâu phục vụ học tập, đọc tài liệu khoa học và viết mã nguồn lập trình hệ thống âm thanh quốc tế:

Bảng 3.6 Bảng tra cứu thuật ngữ Anh — Việt số hóa âm thanh

Thuật ngữ tiếng Anh	Thuật ngữ tiếng Việt	Định nghĩa / Ngữ cảnh sử dụng chuyên ngành
Digitization	Số hóa	Quá trình chuyển đổi tín hiệu tương tự sang tín hiệu số rời rạc
Sampling	Lấy mẫu	Quá trình rời rạc hóa tín hiệu liên tục theo trục thời gian
Sampling rate (fs)	Tần số lấy mẫu	Số lượng mẫu đo được trong thời gian một giây (Hz)
Nyquist Frequency	Tần số Nyquist	Giới hạn tần số cực đại bằng fs/2 biểu diễn trung thực được
Aliasing	Hiện tượng chồng phổ	Méo răng cưa phổ khi tần số vượt Nyquist bị dội về tần số thấp giả
Anti-aliasing Filter	Bộ lọc chống răng cưa	Bộ lọc thông thấp cắt bỏ dải tần vượt Nyquist trước khối ADC
Quantization	Lượng tử hóa	Quá trình rời rạc hóa và làm tròn biên độ mẫu theo trục dọc
Bit Depth	Độ sâu bit	Số lượng bit nhị phân dùng để mã hóa biên độ của mỗi điểm mẫu
Quantization Noise	Nhiễu lượng tử	Sai số biên độ sinh ra giữa giá trị thực và giá trị làm tròn
Dither	Nhiễu làm mịn dither	Nhiễu ngẫu nhiên nhỏ thêm vào trước khi hạ bit để ngẫu nhiên hóa sai số
dBFS	Decibel toàn dải số	Thang đo decibel trong máy tính số với mốc cực đại trần là 0 dBFS
Uncompressed Format	Định dạng không nén	Định dạng lưu trữ PCM nguyên bản thô sơ (WAV, AIFF)
Lossless Compression	Nén không tổn hao	Thuật toán nén entropy bảo toàn 100% dữ liệu gốc (FLAC, ALAC)
Lossy Compression	Nén có tổn hao	Thuật toán nén loại bỏ dữ liệu theo che lấp thính giác (MP3, AAC, Opus)
Fake Lossless	Giả danh không tổn hao	Hiện tượng file transcode ngược từ nguồn nén lossy chất lượng thấp

TÀI LIỆU THAM KHẢO CHƯƠNG 3

[1] Oppenheim, A. V., & Schafer, R. W. (2010). *Discrete-Time Signal Processing* (3rd ed.). Upper Saddle River, NJ: Prentice Hall.

[2] Pohlmann, K. C. (2010). *Principles of Digital Audio* (6th ed.). New York: McGraw-Hill.

[3] Smith, J. O. (2007). *Mathematics of the Discrete Fourier Transform (DFT) with Audio Applications* (2nd ed.). Stanford University: W3K Publishing.

[4] Défossez, A., Copet, J., Synnaeve, G., & Adi, Y. (2022). *High Fidelity Neural Audio Compression* (EnCodec). arXiv:2210.13438. Meta AI Research.

[5] Valin, J.-M., Vos, K., & Terriberry, T. (2012). *Definition of the Opus Audio Codec* (RFC 6716). Internet Engineering Task Force.

[6] Nguyễn An Sỹ. (2018). *Giáo trình Kỹ thuật âm thanh số và Truyền hình*. Hà Nội: Nhà xuất bản Khoa học và Kỹ thuật.

[7] Zeghidour, N., Luebs, A., Omran, A., Skoglund, J., & Tagliasacchi, M. (2022). SoundStream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30, 495–507.

[8] Kumar, R., Seetharaman, P., Luebs, A., Kumar, I., & Kumar, K. (2023). High-fidelity audio compression with improved RVQGAN. *Advances in Neural Information Processing Systems (NeurIPS)*, 36.

[9] Défossez, A., Mazaré, L., Orsini, M., et al. (2024). *Moshi: A speech-text foundation model for real-time dialogue*. arXiv:2410.00037.

[10] Lipshitz, S. P., Wannamaker, R. A., & Vanderkooy, J. (1992). Quantization and dither: A theoretical survey. *Journal of the Audio Engineering Society*, 40(5), 355–375.

[11] ISO/IEC. (1993). *ISO/IEC 11172-3:1993 — Information technology — Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s — Part 3: Audio*. Geneva: ISO/IEC.

[12] Internet Engineering Task Force (IETF). (2024). *RFC 9639: Free Lossless Audio Codec (FLAC)*. IETF.