2. Project 0. Warm Up

Important

Please read the following instructions carefully:

1. This project is to be completed by each group individually.
2. This project contains 2 tasks (3 points).
3. Submit your code through Blackboard. The submission due date is **Sep. 29, 2024**.
4. Each group needs to submit the code once and only once. This project is to be checked together with Project 1.
5. During the performance assessment, any task can only be attempted up to 5 times.

2.1. Overview

The main purpose of Project 0 is to quickly convey the style and potential challenges of the course project to students who plan to dedicate their time to it. Its first part includes self-assessment questions from former students (credit: Chibin Zhang), and the second task is the initial step of Project 1, focusing on environment setup.

2.2. Task 1:(0 point) Self-assessment

The following questions are summarized from students who have taken the course.

- Do you prefer a hands-on coding experience?

  This course is project-oriented. The project accounts for a considerable amount of the total credit. You will have to write a lot of code throughout the semester. If your goal is to understand computer network protocols conceptually without getting your hands dirty, then please consult the instructor for further advice.

- Do you feel comfortable with learning on your own?

  Project specification does not mandate the use of programming languages or libraries. This means that you are on your own to figure out which language or library to use. The course teaching team does not help with implementation issues.

- Do you feel comfortable with multi-thread programming?

  You will need to send and receive packets at the same time, and thus at least 2 threads are needed. Multithreaded code can be difficult to debug due to distributed execution.

- Do you feel comfortable with hardware interfaces?

  Unlike other CS courses where testcases are nicely-formatted and well-defined, this course does not have any testcases. Instead, you will have to interact with the real world in this project, which is cool and exciting. However, you might have to tune your code to fit hardware devices that usually do not guarantee a “perfect” outcome. You will have to face various hardware issues.

2.3. Task 2: (3 points) Understanding Your Tools

We will build a functional computer network from scratch through the four course projects. Conventional network cards transmit data by modulating electrical, e.g., Ethernet, or radio, e.g., Wi-Fi, signals. However, we will use physical/electrical audio signals as the carrier for information transmission. The main purpose of doing this is to simplify the implementation of the physical layer while not overlooking it entirely. Sound cards provide a very flexible and convenient way to control and capture audio signals. Although the data rates achievable using sound cards are much lower than those of typical network cards, they are sufficient for us to understand the essential design elements of the physical layer.

A sound card can convert audio signals between digital and analog (physical) forms. It achieves this function through DAC (Digital-to-Analog Converter) and ADC (Analog-to-Digital Converter) circuits. Sound cards allows us to play any digital waveform to the physical world as sound and capture sound signals from the physical world.

At a higher level, the processing of digital audio signals is similar across different operating systems. That is, playing sound involves copying the discrete samples of the digital waveform into the buffer of the sound card driver. The sound card driver copies the buffer content to the sound card's hardware buffer and then uses the DAC to convert the waveform into electrical signals to drive speakers or headphones. The recording process is the opposite. The sound card stores the power of the sound signals sampled by the ADC, then copies them to the operating system's buffer for use by applications.

But in detail, different operating systems or audio processing architectures handle the aforementioned process, especially the audio buffers, in significantly different ways. Some audio processing architectures reserve large buffers for audio mixing and signal processing purposes, and/or they have to process audio samples through heavy processing stacks, resulting in latencies between digital and analog signals. This means there is a substantial and uncertain time gap between the time when a sample is written into/read from the audio buffer and the time when the audio of that sample is actually played/recorded by the audio card. In normal audio playback and recording situations, these latencies might not be noticeable, but when using audio signals for time-sensitive applications, such as Project 2 tasks, these factors could lead to performance bottlenecks.

Therefore, we recommend using ASIO (Audio Stream Input/Output) to talk with the sound card right from the first step of the project. ASIO differs from the system's default audio processing architecture, such as WindowsAudio, DirectSound, and ALSA. ASIO is at its best to directly interact with the sound card driver and renders a high-performance buffer management interface. This can facilitate the performance optimization of your implementation.

ASIO has several driver implementations. ASIO4ALL working in Windows is verified by the teaching team. For programming in JAVA and C++, you can refer to this page for step-by-step setup instructions. For programming in other languages, you can either access the ASIO native APIs directly or find other wrappers on your own. Linux and MAC OS users can try WineASIO, but the teaching team has not tested it.

This task will familiarize you with using the computer's audio interface. To play and record sounds simultaneously using the sound card, you may want to implement two

threads. If you are interested in the performance of different audio architectures, you may want to leverage RTL Utility. It measures latency by continuously playing and capturing sound pulses. With appropriate buffer sizes, the latency of ASIO driver should be below 20 ms.

Tip

- Stereo audio has two channels, i.e., left and right. You only need one channel to complete the course project.
- Sound card can be configured to different sampling rates. 48 kHz is recommended for both playback and recording, and succeeding tasks.
- Be aware of audio enhancement features that come with the sound card driver. If you are unsure about their functions, disable them.

Performance Assessment

The group provides one device: `NODE1`.

- Objective 1 (1.5 points): `NODE1` should record the TA's voice for 10 seconds and accurately replay the recorded sound.
- Objective 2 (1.5 points): `NODE1` must simultaneously play a predefined sound wave (e.g., a song) and record the playing sound. The TA may speak during the recording. After 10 seconds, the playback and recording should stop. Then, `NODE1` must accurately replay the recorded sound.