Lab 2: Hands-On PHY Layer

Introduction

In this lab, you will form groups of four to implement a physical (PHY) network layer from scratch using audio systems. By leveraging sound as a medium for data transmission, you will receive hands-on experience in encoding, transmitting, and decoding information through audio signals. This lab aims to reinforce theoretical concepts in network communication while enhancing practical skills in signal processing and protocol design.

Plagiarism Warning

There will be **strict** plagiarism checks for this assignment. You are not allowed to copy code verbatim from other groups, or even from groups of your senior batches. An elaborate MOSS check would be used for this plagiarism check.
If your code is taken from a public website or created using generative AI, then you must clearly mention this in your code (as a comment). Moreover, such usage of web/AI must be limited to at most 50% of your code.

Experimental Setup

The system you will build should consist of a Sender and a Receiver. The device(s) comprising each of these is/are your choice. You could, for example, choose a laptop to be the Sender and another laptop to be the Receiver, but the exact choice is yours.
Input to the system will be in the form of messages made of arbitrary number of bits, which you should reliably be able to transfer to the other end. The number of bits in a message would be at most 20. As part of the design, you will need to address many aspects such as bit encoding, identifying the beginning and end of message blocks, handling reliability, etc.
During experimental evaluation (i.e. the demo of your system), we will provide messages which the sender will need to successfully transmit to the receiver. You will be given 2 chances at 2 different points in time. Each chance will comprise of the act of transferring two messages. One of these is going to be a short message and the other is going to be a longer message. The longer message will be carrying more points towards the final grading. Finally, we will be taking the best 1 out of the 2 short messages and best 1 out of the 2 long messages (over the 2 chances) to calculate your score.
In addition to the encoding of the message data (codeword), you are also allowed to send a preamble (either as a separate transmission just before the transmission of the codeword or along with the codeword). You may assume that the preamble would be transmitted error-free. However, the preamble must be **strictly independent** of the erring indices, and **any** dependence of the preamble on the message must only be a **function** of the length of the message.
To simulate real-world scenarios, we would also like you to introduce bit errors during transmission. So, once you prepare the message for transmission i.e. preamble (if applicable) + codeword, you would have to flip one or two bits in this message and then

transmit. Information regarding what bits to flip is provided in the next section. Note that short messages would have a single bit error and long messages would have 2 bit errors.

Example:

Raw Data bits ($n = 20$ bits long): 10110011100001110001
Bits after adding redundancy to handle bit errors i.e. codeword: $a_k; k = 1, 2, \ldots, m$ $a_k \in \{0, 1\}$.
Message prepared for transmission: $b_k; k = 1, 2, \ldots, p + m$ where $b_k \in \{0, 1\}$ and $b_{p+i} = a_i \; \forall i = 1, 2, \ldots, m$ where there are p bits in the preamble of the transmission (either transmitted separately just before the codeword or transmitted along with the codeword)
Flipped bits: 10, 15 (this means b_k for k=10 and k=15 must be flipped before transmission.)
(The above is just an example, it will be made sure that the error bit indices will strictly fall outside the preamble)
Two students would be randomly chosen as the sending team and the other two would be the receiving team. They would be placed in close proximity, however in such a way that the two teams are out of each other's visual field of view. Wifi, Mobile Broadband (4G), etc. must be strictly turned off on both the sender and the receiver laptops. Bluetooth is allowed but only as long as it is used for connecting Bluetooth speakers/microphones for aiding in transmission. If at any point it is found that Bluetooth is being used for **direct** communication between the two terminal devices, you would be awarded 0 for the assignment.
Sometimes the audio reception can be poor (completely possible, does happen at times). In such a case, the human at the receiving end can request the TA to allow the sender to retransmit the same audio string. However, please make sure your mechanism is robust enough to accommodate the receptions. We can allow retransmits once or twice, but not more than that.

Program Description

Everything in this lab needs to be **automated**. There should be a program on the sender side, and a program on the receiver side. Whether these programs are same or distinct is left up to you.

Sender Side: The sender program should take exactly and only as inputs the message bits (provided as a contiguous string of 0s and 1s) on the first line, and on the second line, it should take a space-separated list of two real numbers $\in [0, 1]$ which would be used to calculate the position of bit errors in the codeword as follows:
Let the transmitted message consist of p bits of preamble and m bits of code word. Let the two real numbers given to you be a and b. If $b = 0$, it means that there is a single bit error whose position is $p + \lceil a * m \rceil$. If $b \neq 0$, you must flip the bits at bit positions $p + \lceil a * m \rceil, p + \lceil b * m \rceil$ (where $\lceil \cdot \rceil$ is the ceiling function) in the message prepared for transmission and then transmit this message (or transmit the preamble separately before if you're doing it that way).
A python program will be used to generate random bit-strings and the two real numbers

by the TAs during the demo. You will be asked to download this program in your laptops before coming to the demo so that it would be easy for the TAs to copy and paste the bitstrings into the input stream of your program (along with the real numbers).
You must also display the message prepared for transmission (before introducing bit errors) and the message that is transmitted (post introduction of bit errors) on your screens.
Receiver Side: The person at the receiver end is allowed to manually start and stop the recording/reception. This is the **only non-automated part** allowed at the receiver's end. This program should output the received transmission from the sender, message after performing error correction and the correct input bit-string that was given by the TA.

Program Requirements

Communication across the two ends should occur purely via **audio mechanisms only**. Absolutely, no communication by any other methods, including verbal speech, would be permitted during the duration of the experiment. You could look up suitable python modules to aid you in audio transmission. One such module is `PyAudio` (which works on both Windows and Linux).

Time Measurement Details

The time of the experiment would be measured starting from the moment the "Enter Key" is pressed to send the input on the sender program, and end once the receiver program prints the output on the screen. This would be used to measure your method's throughput which would affect your final demo score i.e. the demo score for higher throughput (with correct transmission) would be $\geq$ the demo score for a lower throughput (with correct transmission).

Design Document

This lab spans two weeks, and will culminate with a demo of your implementation. So as to make sure that you're on track with the lab and haven't postponed everything to the last minute, you are required to submit a design document outlining the implementation idea (keep it concise) i.e. encoding, error detection/correction, reception etc.
Concise does not mean incomplete. Please make sure that your description would allow an intelligent independent reader to be able to replicate your design.

Submission Instructions

At the end of the first week, you should submit a design document pdf named

`<rollno_1>_<rollno_2>_<rollno_3>_<rollno_4>_dd_CS378_lab2.pdf`

Before your demo, you are supposed to submit a tar file containing all the code, documentation you've written for the code along with running instructions for both the sender and the receiver, and the final design document (if you've made any significant changes to the first design document submitted a week earlier). The file must be named

`<rollno_1>_<rollno_2>_<rollno_3>_<rollno_4>_CS378_lab2.tgz`

Note that all the letters in your roll number, if applicable, must be in lower case.

There are going to be **individual** vivas after the demo, so make sure that the entire team knows all aspects of the implementation.

Grading

The score distribution for this lab is as follows:

- Design Document - 25%
- Successful Demo - 40%
- Well-Documented Code - 25%
- Viva - 10%

The viva would be taken individually.
Note that the viva would consist of two components. The first component would check that you actually know how your model works and can explain its working satisfactorily. Success in this component of this viva is crucial to ensure that you are awarded marks for this lab. In particular, if you are not able to satisfactorily explain the model, you shall receive a 0 in this lab.
The second component would comprise of questions about the physical layer to check your understanding of it. This is the component that's worth 10% of the grades.

Words of Caution - PLEASE TAKE THIS VERY SERIOUSLY

Presumably, most of you will be using your laptops at both the sender and receiver ends. It is possible that your laptop speakers get temporarily damaged on frequent transmissions of high frequency signals. However, this temporary damage goes away by allowing the speakers to be idle for a few minutes. To avoid last-minute panic, make sure that you start working on the assignment well in advance so that there is enough cool down time for your laptop speakers to recover in between your several test experiments. Also, it would be a tough day for a lot of you if most of you start working on the assignment in the COMPUTING COMPLEX just one or two days before the deadline, because all of your signals will start interfering with each other :).

Hacking Ideas

We very much appreciate hacks and loopholes in the problem statement and encourage you to post them on Piazza for everyone's amusement. However, do note that such hacks (irrespective of whether they have been mentioned on Piazza or not) which go against the pedagogical spirit of the assignment won't attract marks if applied in practice. If in doubt, please clarify on Piazza (you could choose to send a private message too).

Next Lab's Heads Up

In the very next lab, you would be upgrading your mechanism with a MAC layer to handle collisions. In that lab, there would be senders wanting to simultaneously send messages. Note that on adding the MAC layer, all the nodes must be capable of both receiving and sending, so think about how you might want to structure your code and implementation properly now itself so that adding the MAC layer won't cause too much trouble later. More details will follow in the upcoming weeks.