LAB 03 - REPORT

February 9, 2018

Vighnesh Reddy | Vamsi Krishna Reddy | Lakshmi Narayana | Yaswanth Kumar

(160050090 | 160050064 | 160050080 | 160050066)

1 Physical Layer

We are using sound to transfer data bits. 1500 Hz — represents 0, 8000 Hz — represents 1 and 4200 Hz — to represent the previous repeating bit (as explained below). Our design is insensitive to low frequency noises (upto 800 Hz). Every beep occurs in nearly 0.3 seconds invertal.

2 Protocol and Encoding

- The message (which is atmost 20 bits) is first resized to 21 bits by appending a 1 and followed by any appropiate 0's needed.
- Now this normalized message is encoded using the **Hamming Coding** (precisely Hamming(27,6)). Details of this standard and effective encoding can be found here: Hamming Coding - Wikipedia
- Also at the client side, when the client either sends a retransmission request (1 — 8000 Hz) or a successful acknowledgement (0 — 1500 Hz)
- So conclusively, our message is effectively encoded into a 27 bit code and is transmitted to the client via. physical layer.
- Also, our code can correct one bit errors and detect 2 bit errors. On detection of 2 bit errors, a retransmission request is sent.

3 Softwares Used and Dependencies

- Python3 with Pyaudio
- LaTeX for Documentation

4 Project Files

- **main.py**: Contains host(), client() functions with provide the main functionality.
- **hamming_coding.py**: Contains the hamming_encode() and hamming_decode() which implement the above encodings.
- **generate_listen.py**: Contains client_generate(), host_generate(), client_listen(), host_listen() functions.
- **audio_io.py**: Contains the basic audio generating and recording functions. (Inspired from code at Github)

It was a great experience working on this project!