University of Pennsylvania
Department of Electrical and System Engineering
Digital Audio Basics

ESE150, Spring 2019 | Final | Monday, May 6

- Exam ends at 5:00PM; begin as instructed (target 3:00PM)
- Do not open exam until instructed to begin exam.
- Problems weighted as shown.
- Calculators allowed.
- Closed book = No text or notes allowed.
- Provided reference materials on next to last page.
- Show work for partial credit consideration.
- Unless otherwise noted, answers to two significant figures are sufficient.
- Sign Code of Academic Integrity statement (see last page for code).

____

I certify that I have complied with the University of Pennsylvania's Code of Academic Integrity in completing this exam.

Name:

1						2				3	4					
a	b	c.i	c.ii	d.i	d.ii	a	b	c	d		a	b.i	b.ii	c.i	c.ii	c.iii
1	4	3	2	2	3	1	1	8	5	10	5	2	2	2	2	2

5					6						7					**Total**
a	b	c	d	e	a	b	c	d	e	f	a	b.i	b.ii	b.iiii	b.iv	
3	3	3	3	3	2	2	2	1	5	3	7	2	2	2	2	100

Hypothetical cat auditory critical bands:

Band Number	Low	High
1	45	100
2	100	200
3	200	300
4	300	400
5	400	500
6	500	600
7	600	800
8	800	1200
9	1200	1500
10	1500	2000
11	2000	2500
12	2500	3000
13	3000	4000
14	4000	5000
15	5000	6000
16	7000	8500
17	8500	10000
18	10000	12000
19	12000	15000
20	15000	18000
21	18000	22000
22	22000	25000
23	25000	30000
24	30000	35000
25	35000	42000
26	42000	46000
27	46000	50000
28	50000	56000
29	56000	60000
30	60000	64000

While the cat auditory range to 64,000 Hz is real. This auditory band structure is a synthetic construct generated just for this problem and likely does not represent reality.

1. Continuing our cat audio compression from the midterm, we again consider that a cat can hear up to 64KHz and likely has similar critical band limitations to humans. Consider the hypothetical band structure shown on the facing page, and make the simplifying assumption that we only need to represent the strongest 4 frequencies in each band over a 25 ms time window to 4 Hz resolution. Assume 16b amplitude quantization for each frequency. What bandwidth do we need to continuously send compressed cat audio in real time (send compressed data for 25 ms of sound in 25 ms)?

   (a) Exploiting this structure, what do you store for each 25 ms window and how many bits does this require?

   (b) What raw bandwidth does this require? [state in bits/second]

   (c) Assume we form one TCP/IP over ethernet packet for every 25 ms window. Each packet has a header and checksum that occupies 40 Bytes along with the compressed payload data for one 25 ms window.

      i. What total bandwidth is required including the packet header and checksum? [state in bits/second]

ii. List at least 3 data fields that exist in the packet header (not including the checksum):

(d) Given that the maximum payload size for TCP/IP over ethernet is 1500 Bytes:

i. How many 25 ms frames can you pack into one TCP/IP packet?

ii. What is the total bandwidth requirement to achieve real-time transmission when you pack this many 25 ms frames into one TCP/IP packet? [state in bits/second]

2. Continuing with the compressed cat audio from the previous question, we want to understand the computational requirements for decoding cat audio. At the receiving end, we will need to decode the data. Consider the following code:

```
   // some headers and definitions omitted
#define REC_LENGTH 2
#define FREQUENCY_OFFSET 1
#define AMPLITUDE_OFFSET 0
void decode(uint16_t *freq, uint16_t *pcm25ms)
{
  for (int i=0;i<PCM_SAMPLES_IN_WINDOW;i++) { // contributes 3 instructions
                                               // per loop iteration
    uint16_t v=0;// 1 instruction
    for (int j=0;j<FREQS;j++) { // contributes 3 instructions per loop iteration
      int f=(freq[j*REC_LENGTH+FREQUENCY_OFFSET])<<2;  // 4 instructions
      uint16_t cycles=((int)(f*i<<16)/SAMPLE_INTERVAL)%(1<<16); // 4 instructions
      uint16_t sine_index=2*PI*cycles; // 1 instruction
      int a=freq[j*REC_LENGTH+AMPLITUDE_OFFSET]; // 2 instruction
      v+=a*SINE_TABLE[sine_index]; // 3 instructions
      }
    pcm25ms[i]=v; // 1 instruction
  }
}
```

(a) Assuming we sample at the Nyquist rate to capture frequencies up to 64KHz, what is PC_SAMPLES_IN_WINDOW for the 25 ms window?

(b) From Problem 1, what is FREQS?

(c) Based on the instruction annotations and your answers to (a) and (b), how many instructions are required for one call to `decode` (equivalently for decoding on 25 ms frame of cat audio)?

(d) How fast must a processor run to decode the cat audio in real-time? Specifically, to decode one 25 ms window in 25 ms? [instructions/second]

3. Implement the following truth table using inverters and 2-input AND and OR gates.

a	b	c	d	out
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

4. For the midterm, you sketched a design to translate cat-audible sounds (0 to 64KHz) down to human-audible sounds (0-22KHz).

   (a) Give one reason why the sketch you turned in for the exam would likely not be patentable?

   (b) Assuming you wrote your own code from scratch to implement the translator:

      i. How would that help in potentially patenting a translator?

      ii. Can you copyright your code? (explain why or why not)

   (c) After writing your code as an App for a popular smart phone, you discover that the audio input to the Analog-to-digital converter in the phone has low-pass filters that only allow frequencies below 22 KHz to be seen by the Analog-to-digital converter. Based on this, you decide it may be better to build your own hardware device to perform the cat-audible to human-audible sound conversion.

      i. Why did the smart phone have this low-pass filter?

      ii. How does audio capture for your hardware design need to differ from the smart phone audio capture?

      iii. Compare the patentability of the hardware device converter versus the software-only App converter?

5. An engineer decides to create a c(h)at (a cat-audio version of skype or face time) for computer-to-computer cat audio conferences. Being concerned about bandwidth requirements and quality, she generalizes the cat-audio compression from Problem 1. It now has 3 parameters:

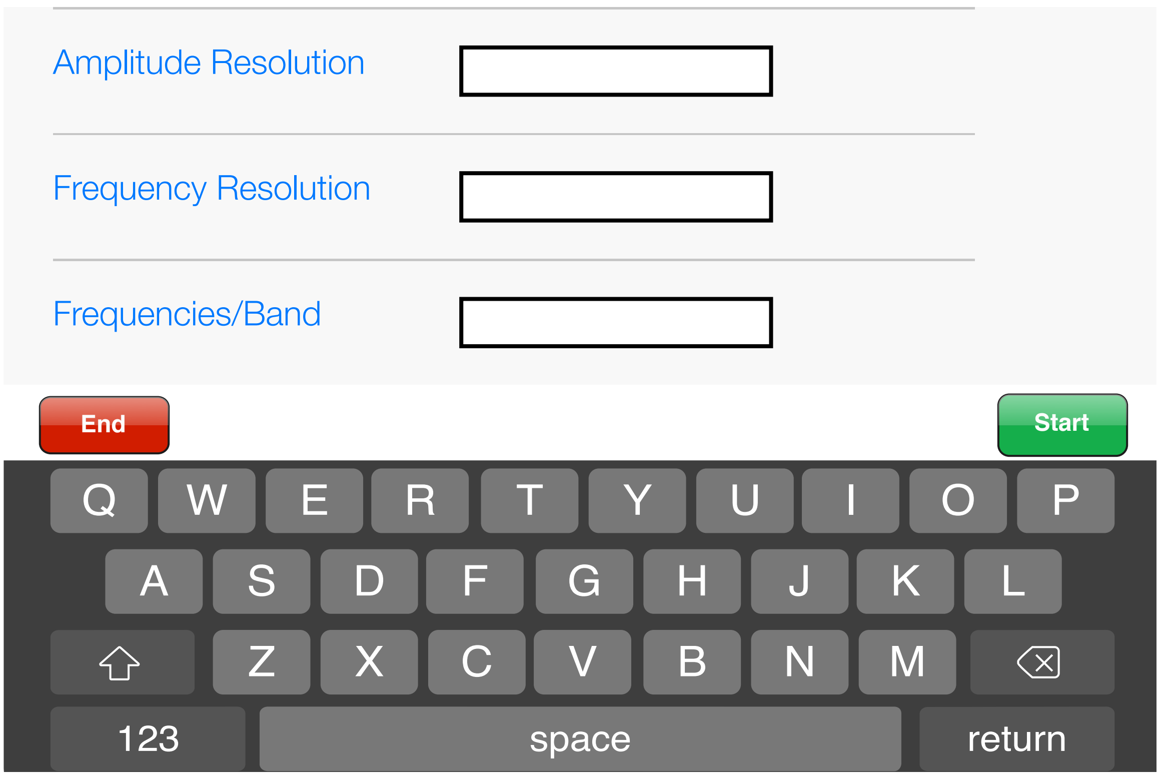
   (a) amplitude quantization

   (b) frequency resolution (quantization)

   (c) number of frequencies to keep per critical band

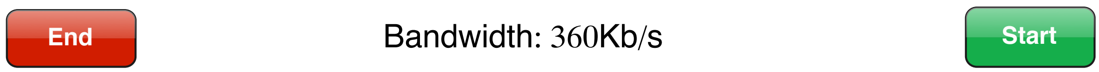
   The c(h)at compressor takes these 3 arguments and compresses sound accordingly, packetizing it for communication over a network (as in Problem 1). The idea is that users will adjust the parameters based on the bandwidth available to them. For each of the following User Interfaces for this task: (a) rank their ease-of-use from (1) easiest to (5) hardest to use; (b) Identify strengths and weaknesses (at least one of each) [hint: general, cat-owning users are unlikely to be familiar with concepts like critical bands and only dimly aware of bandwidth.]:

(a) 3 text boxes that allow you to type any text for the 3 parameters and a start button to indicate you have set the values and are ready to start the c(h)at. When any of the values in the text box are invalid, prints an error message "invalid parameters". When the configuration exceeds the bandwidth available on the link, it drops packets silently (without showing any indication about dropped packets).



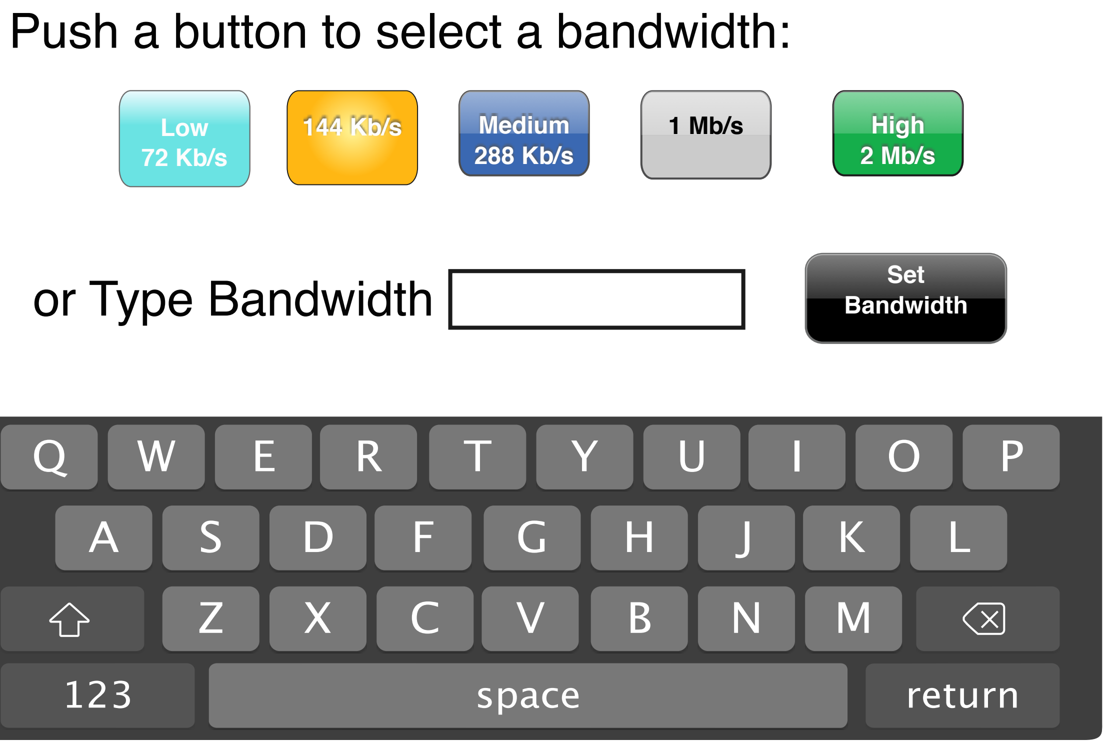
Ease-of-Use Rank	
Strength	
Weakness	

(b) Program has a start button. The program measures the bandwidth on the link, internally determines the values for the 3 parameters to not exceed 80% of the measured link bandwidth, and uses those. When packets are dropped, it remeasures link bandwidth, updates the parameters accordingly, and reports the current bandwidth in use to a status bar at the bottom of the c(h)at window.



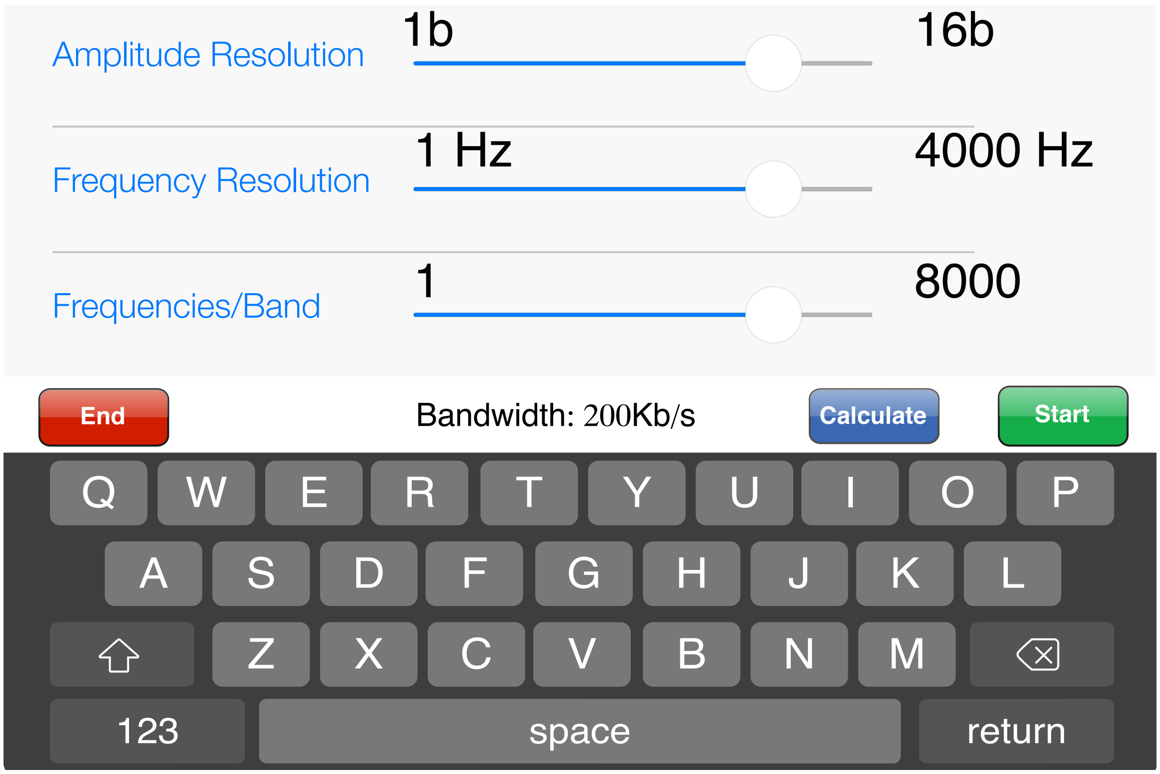
Ease-of-Use Rank	
Strength	
Weakness	

(c) Five buttons to select common bandwidth choices, an optional text box, and a sixth button to set the bandwidth to the value provided in the optional text box. Each of the six buttons start the transfer. When the configuration exceeds the bandwidth available on the link, it: (a) drops to the next smallest common bandwidth, if configured to a common bandwidth, or (b) drops the bandwidth by 10% if the bandwidth was specified in the text box.



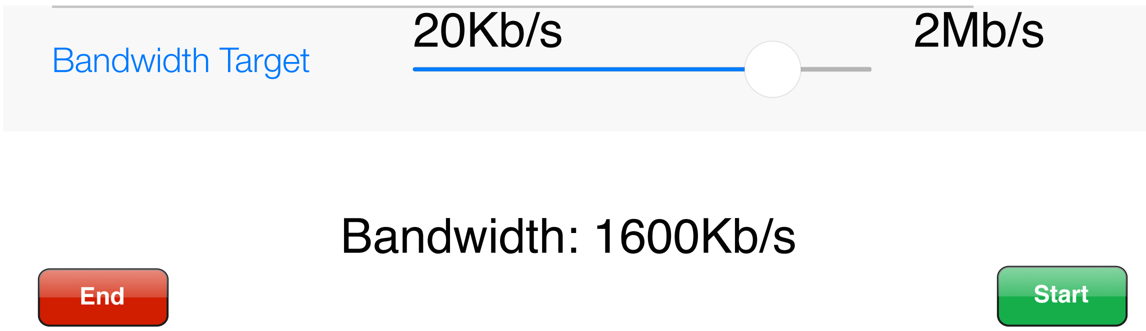
Ease-of-Use Rank	
Strength	
Weakness	

(d) 3 sliders that allow you to select choices within the allowed range of values for each of the 3 parameters, a calculate button, and a start button. When the calculate or start button is selected, the application will calculate and report the bandwidth required to support the specified parameters. When the configuration exceeds the bandwidth available on the link, it drops packets and prints "packets dropped" in a status bar at the bottom of the c(h)at window.



Ease-of-Use Rank	
Strength	
Weakness	

(e) One slider to specify the target bandwidth and a start button. The program internally determines a value for the 3 parameters and uses those. When the configuration exceeds the bandwidth available on the link, it drops packets and reports the packet drop rate on the screen.



Ease-of-Use Rank	
Strength	
Weakness	

6. Consider an Internet-of-Things cat collar with a microphone, an analog-to-digital converter, a small processor, a wireless transmitter, and small battery. We want to use this to record compressed cat audio onto a server. To maximize battery life, where should we perform compression on the data? Assume:

   - Capture 64KHz cat audio.
   - Compress using the scheme from Problem 1.
   - Sending one bit over wireless costs 1 μJ (10^{-6} J).
   - Executing one instruction on the processor costs 1 nJ ($10^{-9} J$) per instruction.
   - The dominant cost in compression is the Fourier Transform to convert to frequencies. Performing an efficient FFT takes 200 instructions per sample point.
   - Detection applies a threshold to the frequencies in the frame. If all frequencies are below an identified threshold, the frame is considered silent and does not need to be stored.
     - Assume detection takes one additional instruction per sample when already doing compression.
     - Detection for uncompressed data requires 5 instructions per sample.
   - On average 95% of frames will be classified as silent.
   - You may ignore the overhead of packet headers for the calculations in this problem.

   (a) How much energy to send each uncompressed 25 ms frame? [Joules]

   (b) How much energy to send each compressed 25 ms frame? [Joules]

   (c) How much energy to compress a 25 ms frame (without detection)? [Joules]

(d) How much energy to detect if a frame has data? [Joules]

i. on uncompressed data?

ii. on compressed data?

(e) What strategy maximizes the battery life? (detail what computation you perform on the collar processor and what data you send out of the collar processor.)

(f) For this strategy, what is the average energy required per hour of operation? [Joules]

7. As discussed in class, a computer with multiple links can help route packets. If it receives a packet that isn't destined for itself, it can send it out along a link that gets the packet closer to the destination. Consider a computer:

- one input link and two output links
- packet length 1000 Bytes (as simplification for exam, assume all packets fixed-size at this length)
- processor spends 100,000 cycles processing each forwarded packet
- the processor on the computer executes 3 Billion (3×10^9) cycles per second.

(a) What fraction of the processor's cycles are spent on routing when supporting 100 Megabit/second network links.

(b) If the computer also kept a record of the IP (IPv4) address of every computer that sent messages through it in a day:

i. maximum number of sources?

ii. Uncompressed space to store those sources?

iii. How much are you likely to be able to compress the list of sources if you Huffman code the list of sources and the computer typically only sees 1000 different sources per hour.

iv. How else might you compress a day's worth of source data? and what compression would you get if you only saw 1000 different sources per hour?

Human auditory critical bands:

Band Number	Low	High
1	20	100
2	100	200
3	200	300
4	300	400
5	400	510
6	510	630
7	630	720
8	720	920
9	920	1080
10	1080	1370
11	1270	1480
12	1480	1720
13	1720	2000
14	2000	2320
15	2320	2700
16	2700	3150
17	3150	3700
18	3700	4400
19	4400	5300
20	5300	6400
21	6400	7700
22	7700	9500
23	9500	12000
24	12000	15500

Code of Academic Integrity

Since the University is an academic community, its fundamental purpose is the pursuit of knowledge. Essential to the success of this educational mission is a commitment to the principles of academic integrity. Every member of the University community is responsible for upholding the highest standards of honesty at all times. Students, as members of the community, are also responsible for adhering to the principles and spirit of the following Code of Academic Integrity.*

Academic Dishonesty Definitions

Activities that have the effect or intention of interfering with education, pursuit of knowledge, or fair evaluation of a student's performance are prohibited. Examples of such activities include but are not limited to the following definitions:

A. Cheating Using or attempting to use unauthorized assistance, material, or study aids in examinations or other academic work or preventing, or attempting to prevent, another from using authorized assistance, material, or study aids. Example: using a cheat sheet in a quiz or exam, altering a graded exam and resubmitting it for a better grade, etc.

B. Plagiarism Using the ideas, data, or language of another without specific or proper acknowledgment. Example: copying another person's paper, article, or computer work and submitting it for an assignment, cloning someone elses ideas without attribution, failing to use quotation marks where appropriate, etc.

C. Fabrication Submitting contrived or altered information in any academic exercise. Example: making up data for an experiment, fudging data, citing nonexistent articles, contriving sources, etc.

D. Multiple Submissions Multiple submissions: submitting, without prior permission, any work submitted to fulfill another academic requirement.

E. Misrepresentation of academic records Misrepresentation of academic records: misrepresenting or tampering with or attempting to tamper with any portion of a student's transcripts or academic record, either before or after coming to the University of Pennsylvania. Example: forging a change of grade slip, tampering with computer records, falsifying academic information on ones resume, etc.

F. Facilitating Academic Dishonesty Knowingly helping or attempting to help another violate any provision of the Code. Example: working together on a take-home exam, etc.

G. Unfair Advantage Attempting to gain unauthorized advantage over fellow students in an academic exercise. Example: gaining or providing unauthorized access to examination materials, obstructing or interfering with another student's efforts in an academic exercise, lying about a need for an extension for an exam or paper, continuing to write even when time is up during an exam, destroying or keeping library materials for ones own use., etc.

* If a student is unsure whether his action(s) constitute a violation of the Code of Academic Integrity, then it is that student's responsibility to consult with the instructor to clarify any ambiguities.