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

ESE150, Spring 2020 — Final Solution — Monday, May 11

- Exam is a 2 hour timed quiz on Canvas.
  - We recommend you draft your answers in a text file (and save often) so that you don't lose any data if Canvas or your browser were to crash.
- You may take the exam any time during the 24 hour period of Monday, May 11 EST. We will use Canvas scheduling to control availability.
- Calculators and computers allowed. (MATLAB, spreadsheets, etc.)
- Open book = Text and notes allowed.
- Internet is discouraged but not prohibited.
  - We won't give you extra time or other considerations if you have a period of Internet outage during the exam.
  - Our strong recommendation is that you do not plan on using it.
    (e.g., download everything you think you might need before starting the exam.)
  - On the off chance that an Internet outage prevents you from submitting the exam to Canvas, capture (e.g., print to PDF, screen capture, answers in text file) and submit your answers via email (andre@seas.upenn.edu).
- The following are prohibited:
  - Getting help from anyone else (in the class or not)
  - Discussing exam with anyone (other than course staff) during the 24 hour exam period
    - even commiserating (e.g., problem X was hard)
    - even if you believe someone has finished the exam
- Questions during exam (e.g., if you suspect a typo or error in the exam) should go to a private piazza post (preferred so that anyone on course staff can answer) or email.
  - Of course, we will try hard not to make errors, but history suggests there's a chance they remain.
  - We won't promise to be responsive throughout the entire 24 hour period.
  - We will take a poll of when you expect to take it so we can try to be available.
  - As necessary, state the assumptions you make along with your answers.
- Show work for partial credit consideration.
- Unless otherwise noted, answers to two significant figures are sufficient.
- Remember Code of Academic Integrity statement (included next page).

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 else's 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 one's 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.

We've all been using Zoom and Google Meetings for the last half of the term. Like a mobile phone, Zoom uses digital audio for sound and digital video for pictures. Zoom adds the ability to support sound and video from a group of people. It builds on the same technologies we've been understanding during the term. In this final, we'll look at how it puts together the building blocks we've already seen and what optimization issues may come up in multi-person, online meetings.

1. **Multi-source Audio**: One key difference is that Zoom combines live audio from multiple sources. An MP3 player only produces one (already combined) audio source. A cell phone is only sending one audio stream each way.

   (a) Using what you learned in the course about human psychoacoustics, explain why the output stream can be the same size as any of the input streams and provide the same quality. (Quiz 2, 5pts)

   MP3 is designed to provide good audio quality for human perception. For constant bitrate, the quality is roughly comparable. An output at the same bitrate as the inputs should have the same subjective quality. Due to masking effects, we only need to capture the dominant frequencies in each critical bands. As we combine the input streams, we keep the dominant frequencies and drop the masked ones which the human will not perceive. Many details in the input streams will no longer exist, but these are exactly the ones the human will not hear.

   (b) Using what you know about MP3s and human psychoacoustics, briefly describe how you would take multiple 128Kb/s MP3 streams and combine them into a **single** 128Kb/s output MP3 stream to send to a single participant. (Quiz 3, 5pts)

   i. process streams critical-band by critical-band

      - within a band look at all the present frequency components across the input streams and identify which to keep and which are masked; drop the mask frequencies

   ii. reconstruct the MP3 stream with the remaining frequencies within each band

2. **Unique Audioscapes** Continuing on with multi-source audio—in practice, we might like each participant to get a unique combination of input signals. In particular, at least, we don't want to include the participant's own sound in the combined output in order to prevent echos. We might like to customize it further, perhaps allowing each participant to include a unique subset of sounds. That raises questions of where we should do the combining. Continue to assume 128Kb/s MP3 audio streams. Assume adding two streams requires 3 Million instructions per second.

   - What are the worst-case bandwidth and compute requirements for a 15 person meeting in the following cases:

     (a) Server receives all 15 input streams from participants, creates 15 unique combinations, and produces 15 output streams, one for each participant. (Quiz 4, 6pts)

Server	input bandwidth	15× 128Kb/s = 1.9 Mb/s
	output bandwidth	1.9 Mb/s
	computational cycles/s	15 × 14 × 3 =630M cycles/s or 15 × 13 × 3 =585M cycles/s
Participant Computer	input bandwidth	128 Kb/s
	output bandwidth	128 Kb/s
	computational cycles/s	0 cycles/s

Combining N streams requires $N-1$ pairwise combines. So, if you don't mix in the participant's own source, at most mixing 14 streams, so need 13 combining operations.

     (b) Server receives all 15 input streams from participants and sends 14 streams (omit self stream) to each participant. Each participant's computer performs the combination locally. (Quiz 5, 6pts)

Server	input bandwidth	1.9 Mb/s
	output bandwidth	15×14×128 Kb/s = 26 Mb/s
	computational cycles/s	0
Participant Computer	input bandwidth	14×128Kb/s = 1.8 Mb/s
	output bandwidth	128 Kb/s
	computational cycles/s	14 × 3=42 M cycles/s or 13 × 3=39 M cycles/s

3. **Multi-source Video** Zoom provides both video and audio. As we've seen from time-to-time, video compression exploits similar concepts. Furthermore, each individual is able configure their videoscape as well (configuration and size of individual video streams, including shared desktop stream). Assume for simplicity that each video stream (cameras from participants, desktop view from presenter, video output to be displayed for each participant) contains 1024×1024 32b pixels.

   (a) The limitation here is the computer display window (or maybe our bandwidth budget). Describe how this limit constrainting the output video for a participant is an analog to limits on human sound perception. (Quiz 6, 5pts)

   As the human can only perceive limited number of frequencies per critical band, the screen can only display a certain number of pixels. In some way, the composite input pixels must be reduced to fit on the limited pixels available on the screen.

   Alternately: The limited screen size does mean limited resolution or limited spatial frequency. So, there's also an analogy to humans not perceiving high audio frequencies. The display cannot represent spatial frequencies that are too high. Fully connecting this means observing that the resolution available for each of the video streams when sharing the screen will be lower than when a single video stream has the entire screen.

   Talking only about human perception and not relating it to the limited representation of the screen – 3 pts

   Describing in terms of masking – 2pts

   (b) Considering only single image frames at a time from the video streams, in order to pack the composite image into a single image (e.g. pack 4 1024×1024 input images into an array of 2×2 (was given as 4×4) images on a 1024×1024 composite image), some form of compression is required. Identify the form of compression (or explain how to perform the compression) and classify it as lossy or lossless. (Quiz 7, 5 pts)

   Reducing a 1024×1024 image to a 512×512 image (or a 256×256 if it was 4×4 images) is an example of coarser-grained sampling. We are reducing the spatial sampling rate by a factor of two (or 4). This is a form of lossy compression.

   3 pts if only say lossless, but not describe how compress (form of compression).

   2 pts if try to use a lossless encoding

4. **User Interface for Multi-source Video and Audio** As noted, above (and demonstrated in Zoom), we may want to allow each participant to customize the audio and video streams they see and hear. To contain the scope, let's focus the audioscape and allow:

   - each participant to select which of the participants to hear (listen-to)
   - each participant can select which participants they will allow to hear them
   - each participant can alter their choices during a session (e.g., choosing to whom they will talk, choosing to whom to listen)
   - participants may join and leave the session

   Assume sound combining for a single participant is handled in a process that receives all the input streams. This could be a process on the server or on the participants local machine.

   (a) What configuration is needed in each such participant process to control the combination process? (Quiz 8, 2pts)

   A bit-vector capturing the set of audio streams to combine for the participant.

   (b) Score each of the following UIs according to the evaluation metrics given (similar to what we used in Lab 12):

   - User Time
   - Cognitive Load
   - Error Prone
   - Self Describing

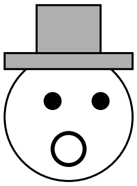
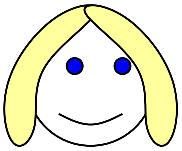
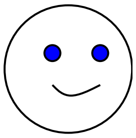
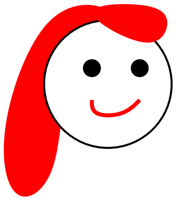
   Rate each 1–5 (1-bad (high time, load, error rate, not self-describing); 5-good (low time, load, error rate, compeltely self-describing). Give a short comment on what makes good/bad.

   i. Conference has a text control console. As participants join and leave, a line is printed on the console. Each user has two commands, "listen" and "allow" that each take a binary string, where each bit in the string represents a participant that they either want to listen to (listen command) or allow to listen to them (allow command). Ordering of bits in string is alphabetical based on last name of current participants. By default new participants are classified as neither allow or listen. (Quiz 9, 3pts)

```
Notice: Conference started.
Notice: Welcome Mike Jones.
Notice: Participant Sally Watkins joined.
Notice: Participant John Smith joined.
Notice: Participant Mary Baker joined.
Action> listen 1011
Action> speak 0011
Notice: Participant Ted Abrams joined.
Action> listen 11011
```

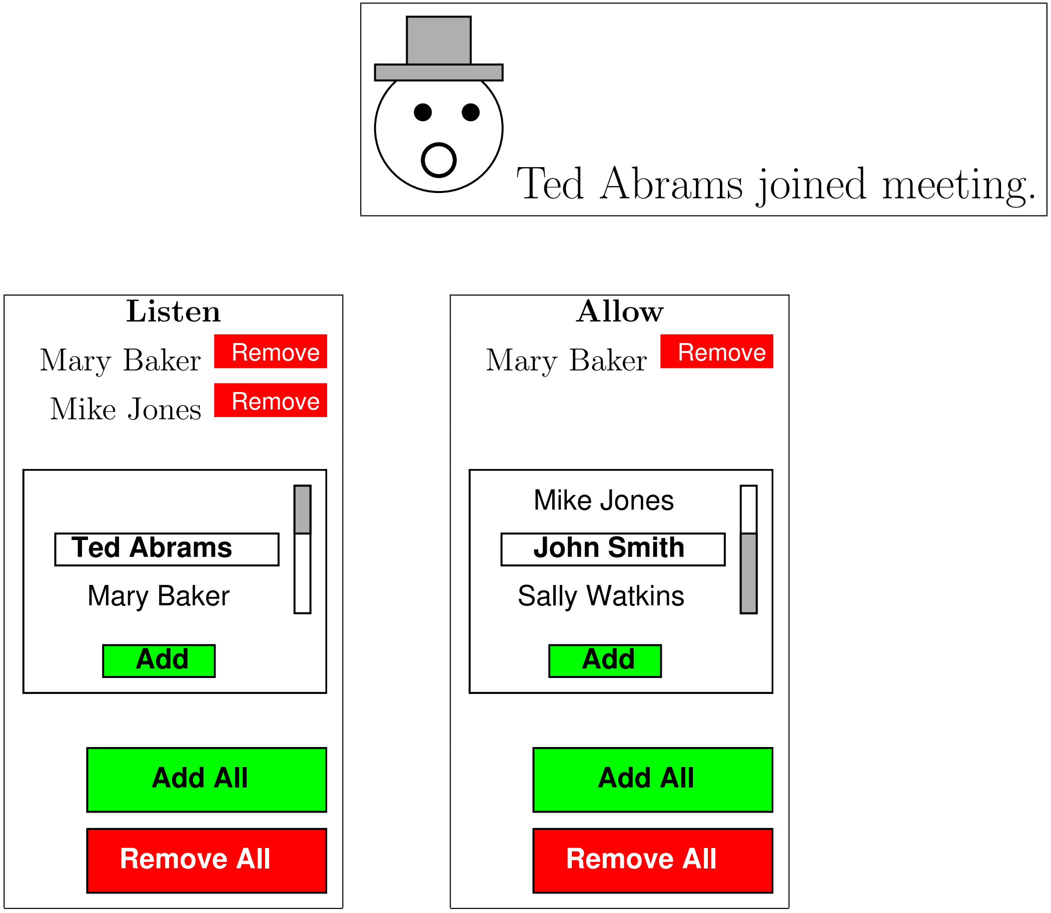
User Time	3	Easy to input, but must input at least a bit for every participant.
Cognitive Load	1	Users must keep track of participants and sort the list of current particpants in their head. This is an extreme example of putting a high burden on the user.
Error Prone	1	If you sorted wrong or missed someone, the bit vector will be wrong.
Self Describing	1	Nothing here tells the user what to do.

ii. Each user has a set of pictures/names for other participants and two associated buttons – one to allow the participant to hear and one to listen to the particpant. Each picture/name displays the current allow and listen state. Particpants that have joined are flagged until the user makes a choice on them. (Quiz 10, 3pts)

	Who	**Listen To**		**Allow Hear**		
	Ted Abrams	Turn On	Turn Off	Turn On	Turn Off	Just Joined
	Mary Baker	ON	Turn Off	Turn On	OFF	
	Mike Jones	ON	Turn Off	Turn On	OFF	
	John Smith	ON	Turn Off	Turn On	OFF	
	Sally Watkins	ON	Turn Off	Turn On	OFF	

User Time	3	Must classify everyone in meeting.
Cognitive Load	5	Shows state, so user can just think about how to correct.
Error Prone	4	Actions are direct, so easy perform and see effect; also easy to see if something went wrong and how to correct.
Self Describing	5	Easy to see current state and buttons that change state.

iii. Each user has a list of allowed and listen-to users. The user can remove a participant from one of the lists with a remove button for the user. A user can add someone to the list by a scroll selection, that also allows the user to type the prefix of the participants last time (or last name, first name) to jump forward in the scroll. When new user's join, a window pops up for 15s to announce their joining. There are also special buttons to add or remove all users to each of the lists. (Quiz 11, 3pts)



User Time	4	Bulk classification means work may be small for common cases.
Cognitive Load	4	Shows state, so user can just think about how to correct. Higher effort to keep track of new participants that may need to be added.
Error Prone	3	Actions are direct, so easy perform and see effect; also easy to see if something went wrong and how to correct. If user accidentally turns all off or on, could be work to recover intended state.
Self Describing	5	Easy to see current state and buttons that change state.

(c) Which user interface is likely to be easiest for the users? Include a short rationale for your selection.

  i. for a small group of 5 participants (Quiz 12, 3pts)
  middle – (Q10) – can easily see all participants on screen and configure with only 5 button selections.

  ii. for a large group of 1000+ participants, such as a large lecture course (Quiz 12, 3pts)
  last – (Q11) – explicitly setting all 1000 participants will be unmangeable.

(d) Based on your selected UI for Q12 (c.i), how do you extract the information that each participant output stream process needs (as identified in Problem **??**)? (Quiz 14, 2pts)

- Each participant already has its own listen-to list. Extract that bit vector.
- Collect the allow-hear vectors from each participant into a matrix with one column per participant
- Extract rows from that matrix to represent what each participant is allowed to hear as the allow vector for the respective participant
- bitwise-AND the listen-to and allow vector for a participant to get the bit vector of input streams to combine

(e) Suggest at least one way to improve your chosen UI or describe a UI that is likely superior. Explain why, include relation to evaluation metrics. (Quiz 15, 3pts)

For middle/Q10, provide a configurable default setting for new participants, so the user doesn't always have to set.

For middle/Q10, provide option to turn on or remove all.

For last/Q11, provide information about unclassified participants; provide options (default, group turn-on/off) for unclassified participants and options to quickly identify unclassified individuals.

5. **Processor and Network handling multiple sessions**

One processor, with a single network connection, can potentially handle multiple, multi-participant (e.g., Zoom) sessions. Assume a system with:

- Single processor running 2 billion instructions per second (2×10^9 instructions/s).
- Single network port that can handle 1 billion bits per second input and 1 billion bits per second output.
- Each Audio+Video stream (input or output) requires 1MB/s (10^6 Bytes/second)
- A meeting session with P participants will require $3 \times 10^6 P^2$ instructions per second. (This assumes server is generating the per-participant output streams.)

(a) Start by considering a processor handling a single, multi-participant session. How could the computer know which packets come from which participant input audio+video stream? (Quiz 16, 3pts)

Have each send to a different port. Separate port streams distinguish separate participants.

Alternately, could look at source IP. Probably still need to look at source port to handle the case where multiple participants are using the same computer.

(b) Moving to multiple, multi-participant sessions (meetings), how can the processor keep track of multiple sessions? (Quiz 17, 3pts)

Use a different process for each session.

(c) How many total input streams can the single network port handle? (Quiz 18, 3pts)

$(10^9 \text{ bits } / \text{ s}) \ / \ (10^6 \text{ B/s} \times 8\text{b/B}) = 125$

(d) Assuming a set of 10 person meetings, how many meetings can the processor computation support? (Quiz 19, 3pts)

$2 \times 10^9 \text{ instructions/s} > M \times 3 \times 10^6 (10)^2 \text{ instructions/s}$

$\frac{2\times10^9}{3\times10^8} = 6.7 > M$

M=6 meetings.

6. **Processing requirements for "wake" words** Smart audio digital assistants (e.g., Alexa, Siri) often process voice recognition on network servers. However, to save bandwidth and computation they will typically not send audio data for remote processing unless they first hear a "wake" word that tells them to pay attention to your commands. The "wake" words are processed locally by simple, low-energy processors that are always processing sounds. Let's understand how these wake-computations may be simpler than arbitrary sound processing.

   First, we know the "wake" words can be perceived by humans. That suggest certain limitations on what it will take to recognize them. Furthermore, they are understood widely by almost all humans, so they will not require particularly high frequencies. Let's assume they can be recognized with no frequency components above 4KHz.

   Assume we use a dot-product, Fourier Transform as in lecture and Lab 8 to extract frequencies (there are more efficient ways, but they are beyond what we covered in class, so we will stick with this simplification). As a result, for a sample window of N samples, it will require $2N$ multiplications and N additions for each frequency component computed. Assume it takes an addition $7N$ instructions for control and loop handling per sample, for a total of $10N$ instructions per frequency component.

   (a) Assuming the assistant must handle MP3-quality audio for some tasks, at what rate must it be able to sample audio input? (Quiz 20, 3pts)

   44KHz – handling sounds up to human hearing around 22Khz

   (b) During idle periods while the assistant is waiting for a "wake" word, at what rate must it sample in order to recognize "wake" words? (Quiz 21, 3pts)?

   8KHz, the Nyquist sampling frequency for the maximum 4KHz frequencies that must be captured for wake-word recognition.

   (c) Assuming the Fourier Transform on 25 ms windows is the dominant computation required to recognize wake words, how many instructions per second must the local processor on the digital assistant support (Quiz 22, 6pts)?
   (Hint: how many samples in the 25 ms window? How many frequencies can you extract from that window?)

   $8000/\text{s} \times 0.025\ \text{s} = 200$ samples and 200 frequencies

   10×200 instructions/frequency $\times$ 200 freququencies $\times$ 40 windows per second

   16 M instruction

   (d) Continuing to assume 25 ms windows, how much more processing would be required if you performed the processing on the full MP3 frequency range? [State assumptions as necessary.] (Quiz 23, 4pts)?

   $44000/\text{s} \times 0.025\ \text{s} = 1100$ samples and 1100 frequencies

   10×1100 instructions/frequency $\times$ 1100 freququencies $\times$ 40 windows per second

484 M instruction

Alternately, we might say that we don't need to produce all the frequencies, only the ones under 4 KHz. So only $1100 \times (4/22) = 200$.

10 × 1100 instructions/frequency × 200 freququencies × 40 windows per second
88 M instruction

7. **Logic to Count Streams to Mix** Show how to compute the number of streams you will need to mix to generate the output stream for a single participant from a 4-input allow (to hear) and is-listening vector as developed in Problem **??**. The inputs will be the vectors relevant to a single output.

   - listen-to[i] = This participant is listening to participant *i*.
   - allow[i] = Participant *i* is allowing this participant to hear them.

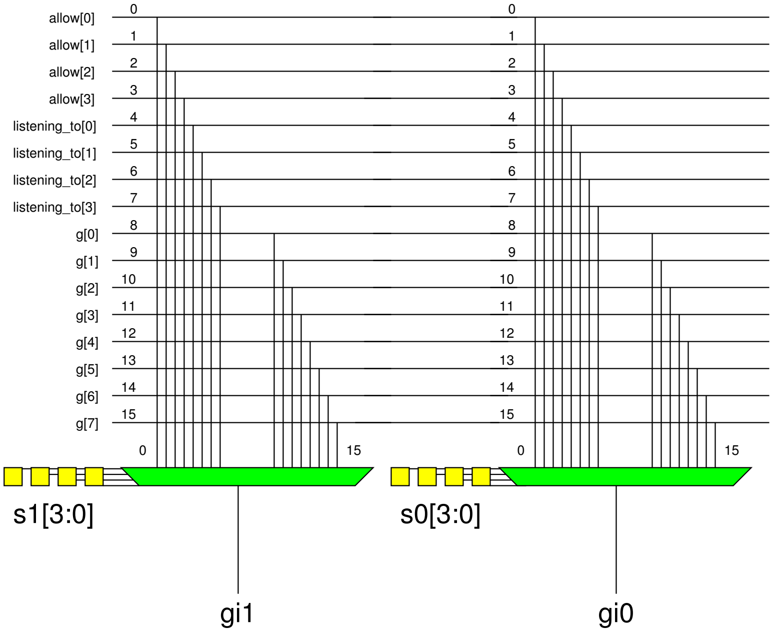
   (a) Design logic to compute the count using primitive logic gates (1-input NOT gate, 2- or 3-input AND, NAND, or OR gates) and 3-input, 2-output Full Adder (FA) gates. (Quiz 24, 5pts)?
   (Hint: given the 4-input bit-vectors, how many bits do you need to hold the result?)
   You may write as functional equations, e.g.

```
t1=AND2(a,b);
t2=AND2(b,c);
t3=AND2(a,b);
carry=OR3(t1,t2,t3);
```

   Need 3b to represent 0 through 4

```
b[0]=AND2(allow[0],listen-to[0]);
b[1]=AND2(allow[1],listen-to[1]);
b[2]=AND2(allow[2],listen-to[2]);
b[3]=AND2(allow[3],listen-to[3]);
(c0,s0)=FA(b[0],b[1],b[2]);
(c1,sum[0])=FA(b[3],s0,0);
(sum[2],sum[1])=FA(c1,c0,0);
Result is sum[2:0]
```

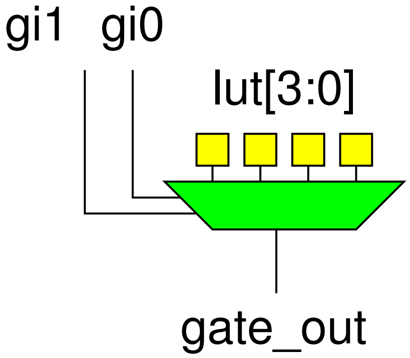
(b) How should you program the interconnect multiplexers shown to select gate inputs (gi1, gi0) as allow[0] and listen-to[0] ? (Quiz 25, 8pts)?



Provide the binary values that need to be stored in s1[3:0] and s0[3:0] to perform the intended selection:

s1[3:0]	0000
s0[3:0]	0100

(c) Program this mux-based programmable gate to perform the AND function on the two inputs gi0 and gi1 (where gi0 and gi1 may be the outputs produced from Question 7b (25)). (Quiz 26, 4pts)?



Provide the binary values that need to be stored in lut[3:0] so that it performs the intended AND function:

1000

3 pts if got that there was only one 1, but put it in the wrong place.

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