Lab 7: Piano Helper

Submission Due Dates:

Demo: 2023/11/28 17:20
Source Code: 2023/11/28 18:30
Report: 2023/12/03 23:59

Objective

Getting familiar with the control of the audio peripheral Pmod I2S2.

Description

Charlie dreams of becoming a skilled pianist but lacks the funds for lessons. His friend Snoopy, an NTHU CS student, surprises him with a birthday gift: a special piano designed to aid Charlie's practice. The piano features a DEMONSTRATE mode and a PLAY mode: the DEMONSTRATE mode can plays pre-written songs to acquaint Charlie with melodies, and the PLAY mode can generate notes as keys are pressed. It also includes a unique "Helper" game.

I/O signal specification

BtnC	Reset	Asynchronous positive reset
BtnU	Volume Up	
BtnD	Volume Down	
SW 0	Play / Pause	1: Play; 0: Pause
SW 1	Start / Exit	1: Start; 0: Exit
SW 14	Mute / Normal	1: Mute; 0: Normal
SW 15	Mode	1: DEMONSTRATE mode; 0:PLAY mode
LED 0 ~ 4	Volume Indicator	
LED 9 ~ 15	Key indicator	
Pmod JB 1~6	Pmod I2S	
7-Segment	Displaying Note & score	

1. DEMONSTRATE mode (40%):

- In the DEMONSTRATE mode, the piano will **repeatedly** play a demonstration song (starting from the beginning again when reaching the end) and will NOT react to any key pressed on the keyboard. (Pick a song from the sample sheets below!)

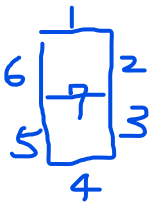
- The piano should support two tracks: one track plays the melody part, and the other plays the accompaniment part.

- By switching the **Play/Pause** switch (SW 0) to “pause,” the piano should pause the demonstration song immediately and should play no tone. The piano should start/resume to play the song from where it paused by switching to “play” mode.

- Upon the **Reset** (BtnC), the volume should go back to its default level 3, and the demonstration song should start from the beginning.

- Support the volume control of 5 levels and the “mute” function.

- **Display the melody pitch** (C, D, E, F, G, A, and B) on the 7-segment display.
  For example:
  - ✧ You should display '- - G 5' when the first note of *Lightly row* is being played.
  - ✧ You should display '- - - -' when the demonstration song is paused.



2. PLAY mode – Piano (30%):

- In the PLAY mode, the piano should make no sound unless you press a key. The following table shows the key-note mapping. Note that the LED labels are for the Helper game.

	LED 15	LED 14	LED 13	LED 12	LED 11	LED 10	LED 9
Key	q	w	e	r	t	y	u
Note	C5 (Do)	D5 (Re)	E5 (Mi)	F5 (Fa)	G5 (Sol)	A5 (La)	B5 (Ti)
Key	a	s	d	f	g	h	j
Note	C4 (Do)	D4 (Re)	E4 (Mi)	F4 (Fa)	G4 (Sol)	A4 (La)	B4 (Ti)
Key	z	x	c	v	b	n	m
Note	C3 (Do)	D3 (Re)	E3 (Mi)	F3 (Fa)	G3 (Sol)	A3 (La)	B3 (Ti)

  If you think these notes are not enough to play a good song, feel free to add more notes with additional keys.

- Raising a note an octave higher means doubling the note's frequency
  (e.g., A4 with the frequency of 440 Hz -> A5 with the frequency of 880 Hz)

- The piano should play a note when you press a corresponding key and stop when you release it. **Only one note will be played at a time**. You may define the behavior if multiple keys are pressed at once.

- Support the volume control of 5 levels and the "mute" function.

- Upon the **Reset** (BtnC), the volume should go back to its default level 3.

- **Display the melody pitch** as the description in the DEMONSTRATE mode**.**

3. PLAY mode – Helper (30%):

- When the **Start/Exit** switch (SW 1) is set to "start," the game initiates. The LEDs will indicate the designated key to be pressed, synchronizing with the melody segment of a preset song.
  For instance, if the note "Do" (C5, C4, or C3) needs to be played, LED15 will illuminate.
  (follow the previous key-note table)
  - Same as the "Piano" function, the design will generate sounds corresponding to the pressed key.
  - If the player correctly presses the key that matches the illuminated LED (and also the 7-segment display), they will earn one point, with the score updating immediately.
  - If a player holds down a key during a sustained note (such as a whole note), the score will continue to increase. The frequency of continuous score increments is determined by your design (it's up to you to decide), as long as it remains reasonable and the score increases by one point at a time when displayed.

    Note: This score increment feature generally makes the implementation easier. We will not test it during the demo. Therefore, make sure your design satisfies the previous

requirement, which is good enough. It is okay if you decide not to implement the score increment feature.
  - The maximum score is 99.

- When the preset song reaches the end, the game will end, and all LEDs [15:9] will illuminate. Players can switch SW1 to 0 to exit the game. (Unlike the DEMONSTRATE mode, the song will not repeat during the game.)
  Note: After the game ends, pressing piano keys will not produce sound, and the score will remain unchanged.

- Upon the **Reset** (BtnC), the volume should go back to its default level 3 and restart the game. The player's score will be reset to zero, and both the LED indicators and the 7-segment display will synchronize with the melody of the preset song, starting from the beginning.

- Support the volume control of 5 levels and the "mute" function.

- Show the notes that should be played on the 7-segment display.
  For example:
  - '0 0 C 4': the last two digits indicate key 'a' should be pressed.
  - '0 1 C 4': the first two digits show the scores.
  - '0 0 - -' : initial state of the game start.
  - '1 6 C 5': 16 is the final score; C5 is the last note of the preset song.

Other settings in detail:

- **Volume:** The piano should support 5 distinguishable levels of volume, controlled by Volume Up (BtnU) and Volume Down (BtnD). The default level of volume is 3. At the lowest level, the sound should still be able to be heard. Pressing Volume Up at level 5 or pressing Volume Down at level 1 takes no effect.
  LED 0~4 indicates the current volume level:

Volume Level	LED 4	LED 3	LED 2	LED 1	LED 0
Muted	○	○	○	○	○
Level 1	○	○	○	○	●
Level 2	○	○	○	●	●
Level 3	○	○	●	●	●
Level 4	○	●	●	●	●
Level 5 (the loudest)	●	●	●	●	●

  Volume control should take effect on both modes with both tracks, also during the game play.

- By switching the **Mute** switch (SW 14) to 1, the player can mute the piano. When muted, the key in the PLAY mode can still be pressed and the demonstration song in the DEMONSTRATE mode will keep playing. But you will hear no sound.

- By switching the **Mode** switch (SW 15), the player can change the piano mode: 0 for the PLAY mode and 1 for the DEMONSTRATE mode. If it is the first time switching to the DEMONSTRATE mode, the song should start from the beginning. If it is NOT the first time switching to the DEMONSTRATE mode, the song should start from where it is paused (or switched to the PLAY mode) last time.

Demo video

I. DEMONSTRATE mode: https://reurl.cc/K0Z7W9
II. PLAY mode – Piano: https://reurl.cc/lD8m1A
III. PLAY mode – Helper: https://reurl.cc/o7lknD

Questions and Discussion

Please answer the following questions in your report and use the provided report template.

A. If we want to provide two songs in the DEMONSTRATE mode by using an additional switch to select the song, how would you modify your design of Lab 7?

B. What happens when we change

```
next_ibeat = (ibeat + 1 < LEN) ? (ibeat + 1) : LEN-1;
```

to

```
next_ibeat = (ibeat + 2 < LEN) ? (ibeat + 2) : LEN-1;
```

?

music 速度 ×2

Guidelines for the report

Note that the grading for this lab is adjusted as follows (please refer to the new report template for further details):

Grading policy: Part (A): **45%**; Part (B): **40%**; Part (C): **10%**; (D): **5%**

A. **Lab Implementation**
   1. Block diagram of the design with an explanation
   2. Partial code screenshot with an explanation:
      - Volume control
      - DEMONSTATE mode
      - PLAY mode – Piano
      - PLAY mode – Helper
      - Score mechanism of the Helper

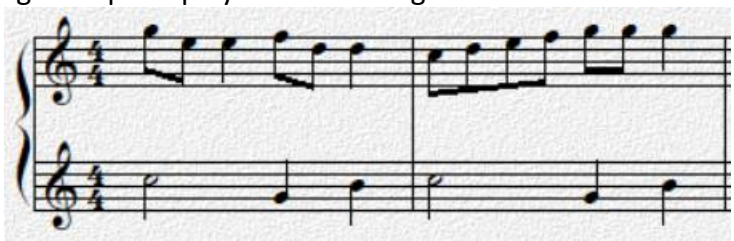
B. **Questions and Discussions**
C. **Problem Encountered**
D. **Suggestions**

Attention

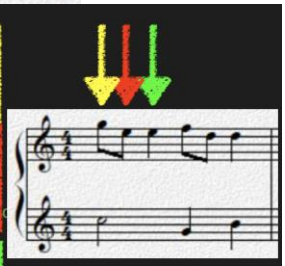
1. You should hand in only one Verilog file, lab7.v.
2. Finish the modules from the template, and integrate them in lab7.v. You don't need to put debounce, onepulse, keyboard_decoder, and speaker_control in the file you hand in. Please do not integrate them in lab7.v.
3. You should also hand in your report as **lab7_report_StudentID.pdf** (e.g.**lab7_report_111456789.pdf**).
4. Please do not hand in any compressed files, which will be considered as an incorrect format.
5. You should be able to answer the questions of this lab from TA during the demo.
6. You need to generate the bitstream before the demo.
7. If you have any questions about the spec, feel free to ask on the EECLASS forum.

Pick a Song

The design template plays the following music.



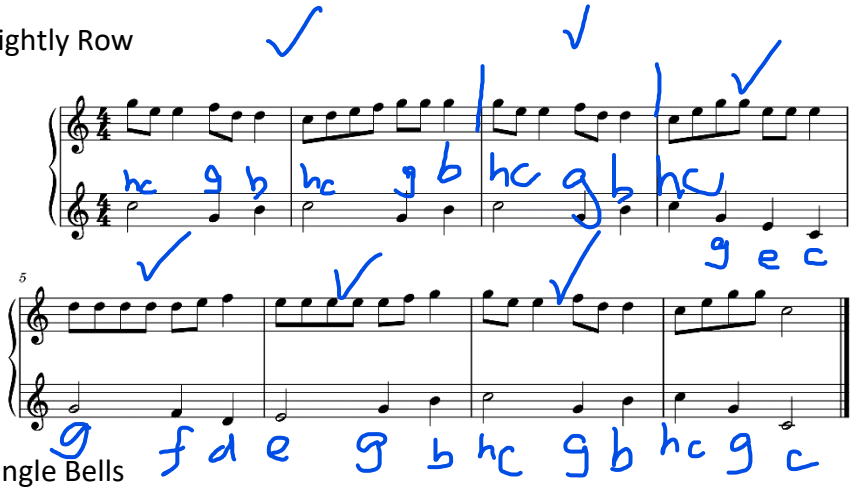
```
// --- Measure 1 ---
12'd0: toneR = `hg;      12'd1: toneR = `hg; // HG (half-beat)
12'd2: toneR = `hg;      12'd3: toneR = `hg;
12'd4: toneR = `hg;      12'd5: toneR = `hg;
12'd6: toneR = `hg;      12'd7: toneR = `hg;
12'd8: toneR = `he;      12'd9: toneR = `he; // HE (half-beat)
12'd10: toneR = `he;     12'd11: toneR = `he;
12'd12: toneR = `he;     12'd13: toneR = `he;
12'd14: toneR = `he;     12'd15: toneR = `sil; // (Short break
12'd16: toneR = `he;     12'd17: toneR = `he; // HE (one-beat)
12'd18: toneR = `he;     12'd19: toneR = `he;
12'd20: toneR = `he;     12'd21: toneR = `he;
12'd22: toneR = `he;     12'd23: toneR = `he;
12'd24: toneR = `he;     12'd25: toneR = `he;
12'd26: toneR = `he;     12'd27: toneR = `he;
12'd28: toneR = `he;     12'd29: toneR = `he;
12'd30: toneR = `he;     12'd31: toneR = `he;
```



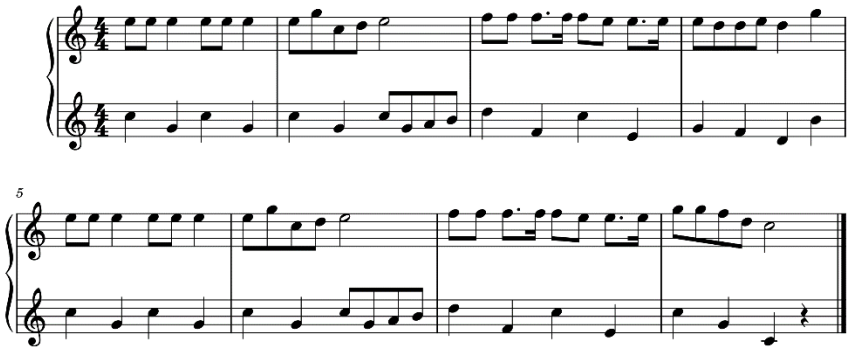
You can pick any song of the sample songs listed below.

They are all 8 measures (小節) in length.

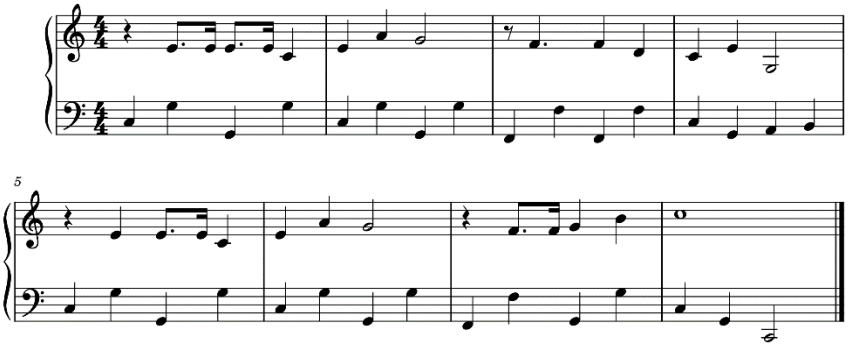
1. Lightly Row



2. Jingle Bells

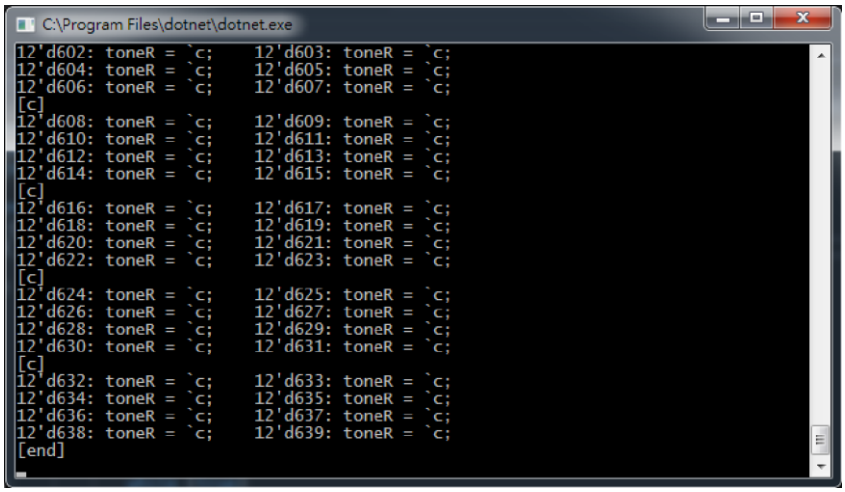


3. 貓咪大戰爭六周年廣告 (Note the bass clef 𝄢)



Hints

- Trace code first. This lab won't be so hard if you realize how the template works.
- Remember that the buzzer/speaker uses 2's complement numbers for audio signals. When designing the volume level, choose the peak value to be some certain *val* and *-val*.
- If two consecutive notes have the same frequency, it may not be possible to tell when the second note starts. Therefore, in the template, one Quarter Note ♩ is divided into 16 beats further, for the sophisticated note arrangement. You may use a short rest (silence) so that the 2 same-frequency notes can be separated by a short break. (Refer to music_example.v).
- A note-to-frequency table is in the appendix for your reference.
- You can use multiple music modules or player modules if that helps.
- You can add or modify some modules in the template.
- We use square waves. So, the music may not sound smooth.
- Since a quarter note is 16 beats, it may be tedious to enter all the notes one by one (8 measures consist of 8*4*16 = 512 beats in total). In that case, you may want to write an aid program like:

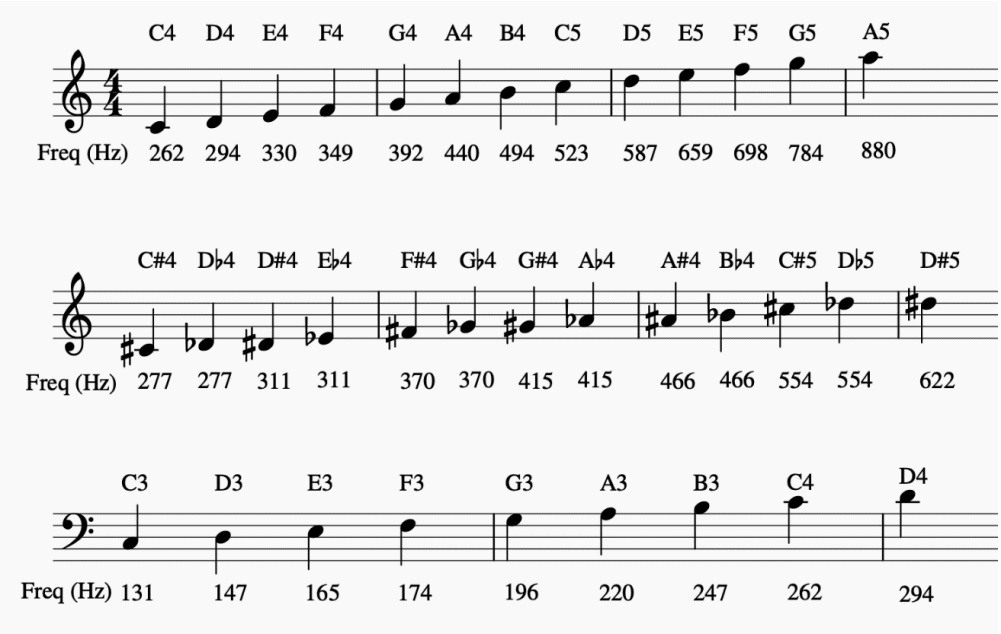


So that you can copy-n-paste them, saving a tremendous amount of time.

Appendix

- **Pitch-to-Frequency Table**

  The number after the notation indicates how high the pitch is.



  You can use these frequencies in your code like that in music_example.v.

  Or you can refer to this page.

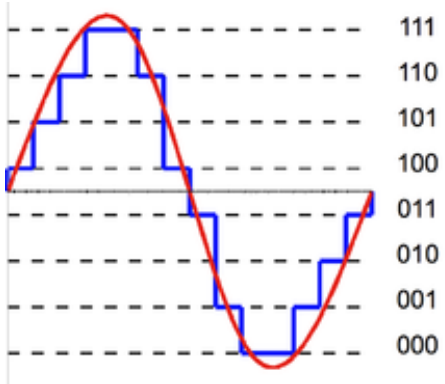
The following discussions are for your reference if you plan to go for an audio-related final project.

- **Square Waves and Sine Waves**

  We use square waves to control the speaker, which inevitably results in a buzzing sound.

  Triangle waves will do, but sine waves are usually the most natural.

  Sine waves can be emulated in Verilog with a look-up table. But it results in some **quantization noises** (if no interpolation is involved) that make the sound terrible. (like the picture below, from Wikipedia)



  However, there is an algorithm known as **CORDIC**, or **Volder's algorithm**, a simple and efficient way to calculate trigonometric functions, even when using the FPGA. Refer to Wikipedia for more

information. Vivado also provides the CORDIC IP in the IP Catalog. So, you may find it handy in this lab if you really cannot stand the square wave sound or if you are going to improve the audio effect in your final project. (You can also generate audio data in your PC and store it in the block memory of Basys3. But it will consume a lot of memory.)

- **Different Timbres** (音色)

  Refer to this video to gain an insight of how timbres are made of. Actually, the frequency change (vibrations on violins), amplitude change (like the fading sound of pianos), and how the sound waves start can determine how we perceive the timbres. Even the ratio of overtones can differ on the same musical instrument when it plays different frequencies.