Project A Electronic piano learning machine

Your team is expected to implement a mini electronic organ to help play music, this circuit supports mode selection and playing functions, and can use VGA as an output device.

Attention::

1. Due to the need for demonstration, the duration of each note can be extended compared to the actual application.
2. The following electronic organ learning methods are for reference only, if you have a better implementation method (reasonable and user-friendly), you can adjust the implementation accordingly.
3. The playing rhythm is designed by your team, and the music can be roughly identified.
4. It is recommended to optimize your learning machine in terms of flexibility, fluency, visualization and fun.

1 Task requirement

When the electronic piano learning machine starts up and works normally, there are three modes available: free mode Auto play mode Learning mode

1.1 1. Basic Functions

1.1.1 Free mode

Press the keys freely, the electronic organ plays the corresponding notes of the keys, such as, press a few buttons, the learning machine plays do, re, mi

1.1.2 Autoplay Mode

The learning machine automatically plays the song, for example, enter the auto play mode, automatically play "Little Star"

1.1.3 Learning mode

Lights up according to note order and duration to guide users to play correctly, for example, according to the order of "Little star", the key under the light should be pressed, and the light will be extinguished after the user presses it, and another light will be turned on

1.1.4 Modular design

Reasonably divided modules handle user input, music storage, control buzzer, led, seven-segment nixie and other outputs.

1.2 Optimization function

1.2.1 Free mode :

On the basis of freely pressing the key, the electronic organ plays the corresponding notes of the key.

- Add an octave adjustment to the note :

  Higher octave: Press a specific key, you can pop out notes that are higher than the original octave, and the same goes for lower octaves

1.2.2 Autoplay Mode :

The learning machine automatically plays songs on the basis of increasing the music library.

- Display track numbers on seven nixie tubes

  After entering the automatic mode, press the button to realize the back and forth pages of the library, select three or more songs, and seven digital tubes display the contents of the library (at least two songs can be played).

- When playing automatically, use lights to indicate the user's playing position and duration

  Press the button to confirm the track, and the learning machine starts to play automatically, starting with a light above the note when it appears, and turning off the light after the note is played until the song is finished.

1.2.3 Learning mode :

- Increased rating for the entire play based on guidance
  For each note, the playing interval is determined, and according to the number of times the user correctly plays, the user's rating is displayed on seven sections of digital tubes or LED lights

1.3 Bonus

User operation is more convenient, display is more intuitive, and performance is smoother.

Including but not limited to the following items:

- Users can adjust the key positions according to their usage habits:
  When the learning device enters adjustment mode, the device plays notes in order like do re mi. After selecting the note corresponding to 'do', press the confirm button, and the machine will play the next note, waiting for the user to choose and confirm again. After all confirmations are complete, the notes corresponding to the keys during user practice will be consistent with the adjustment mode.

- For the user's playing records, you can choose whether to save them in the music library:

Enter the recording mode, record the order and position of the keys pressed by the user, stop recording when the user presses the end key, and then automatically replay it. After the user presses the confirm key, save it in the machine, and it can be replayed through the automatic playback mode later. (Options for delete and overwrite operations are available)

- VGA displays more information about music to assist in guiding the performance and adding interest:
  For example, use some graphics to prompt the timing of the note playing, and display the name of the song instead of the number.

- More refined and reasonable real-time updates for scoring:
  In learning mode, give a real-time changing score based on the difference between the timing of the user pressing the key and the standard. The method of scoring will serve as the basis for evaluation.

- More variation in the rhythm of the music:
  In automatic mode and learning mode, the duration of each note can be inconsistent.

2 Some help

2.1 EGO1 User Manual (required pin description)

EGO1 user manual

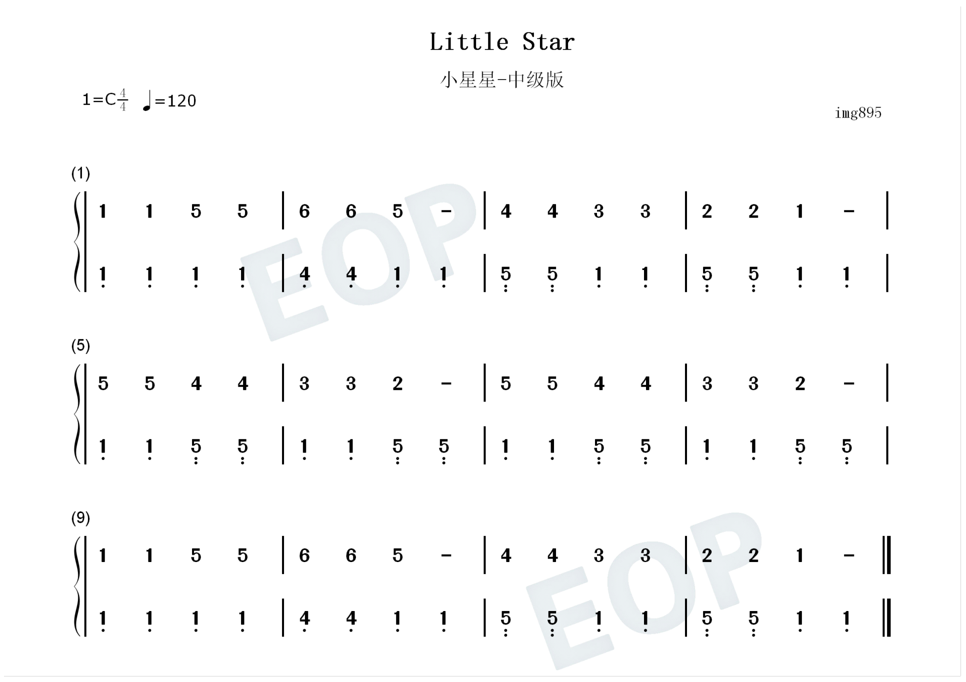


图 1: Little star

2.2 Hardware

The buzzer provided in the course is a passive buzzer. Passive buzzers can be piezoelectric or magnetic buzzers. The way they work is relatively simple. Their pitch is determined by their inherent vibrational frequency. In order for the buzzer to sound correctly, the output frequency needs to be adjusted according to the clock of the development board and the frequency of 'do' in C major.

You may be a little worried about your lack of musical talent, but don't worry! We have prepared the code for you according to the EGO1 clock frequency, in the later code block, as long as you enter the corresponding number, such as input 1, it will output the frequency of the do sound.

When in use, pay attention to the connection of the buzzer line. Through the test, it was found that a long time of power without proper vibration would cause the wire to burn out. If the confirmation code is correct but the buzzer does not sound, the wire may have burned out, although it is not visible, or the interface is incorrectly connected.

The correct interface should connect one end to the positive terminal and the other end to a pin that outputs a specific frequency. Refer to the user manual for specific pins.

If you do have the misfortune to burn out your buzzer wire, don't panic, it will continue to work. Solution: Tear the rubber packaging on the wire, so that the metal part of the wire contact the original connection position. (From Pioneer Explorer)

If you're really confused and you're sure it's not your code, pins, or some other problem that's causing the vocalization problem, you can try the following methods to make sure it still works.

Method to verify that the buzzer is working normally: The left IO column from the top number of the first and second I/O port (the second is positive 3V), will generate current when connected, this time if repeatedly inserted and removed, will force intermittent current. The buzzer gives off a "bared" sound. (From Pioneer experimenter)

The buzzer may not be as loud as you think, so pay attention to the faint sound it makes as it flashes.

2.3 code

Listing 1: Buzzer

```
module Buzzer(
input wire clk,          // Clock signal
input wire[3:0] note,    // Note (Input 1 outputs a signal for 'do,' 2 for 're,' 3 for 'mi,' 4, and so on)
output wire speaker      // Buzzer output signal
);

wire[31:0] notes[7:0];
reg [31:0] counter;
reg pwm;
// Frequencies of do, re, mi, fa, so, la, si
// Obtain the ratio of how long the buzzer should be active in one second
assign notes[1]=381680;
assign notes[2]=340136;
assign notes[3]=303030;
assign notes[4]=285714;
```

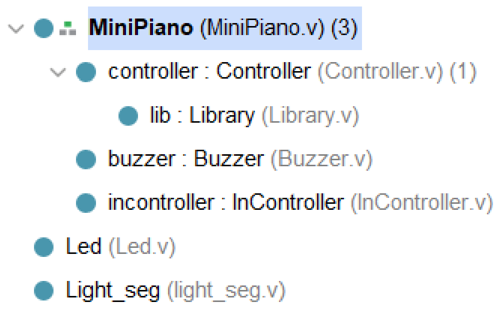
```verilog
assign notes[5]=255102;
assign notes[6]=227273;
assign notes[7]=202429;
initial
begin
pwm=0;
end
always @(posedge clk) begin
if (counter < notes[note]||note==1'b0) begin
counter <= counter + 1'b1;
end else begin
pwm=~pwm;
counter <= 0;
end
end

assign speaker = pwm;  // Output a PWM signal to the
    buzzer
```

You may also wonder how such a frequency is calculated, the do frequency is 261.6Hz, the clock frequency is 100MHz, 100M/261.6=381680. So, the adjustment octaves in the next mission are up to you to explore. Also, when using two-dimensional arrays, note that wire[31:0] notes[7:0], [31:0] is the bit width, [7:0] indicates that the array size is 8 (from 0 to 7), the two meanings are different. (from Pioneer Explorer)

Listing 2: "for", it might help you reduce some of your workload

```verilog
integer i;
for ( i = 0;  i < 8;  i = i + 1)
begin
if (key[i] == inputs )
begin
note <= i;
light<=key[i];
end
end
```



MiniPiano (MiniPiano.v) (3)
    controller : Controller (Controller.v) (1)
        lib : Library (Library.v)
    buzzer : Buzzer (Buzzer.v)
    incontroller : InController (InController.v)
Led (Led.v)
Light_seg (light_seg.v)

图 2: a module example

2.4 Your code module might look like this

We prepared a Bit stream file for testing and recorded a video of it in use. The three petite white switches (R3, T3, T5) on the far right can control its state :011 free mode, 010 Autoplay mode, 101 Adjust button mode.

I hope it can be helpful to you. I wish the project a good success