邏輯設計實驗 Lab10 結報

104060012 邱怡庭

1 Play the 16 sounds repeatedly based on the sound table. Every sound is played for one second.

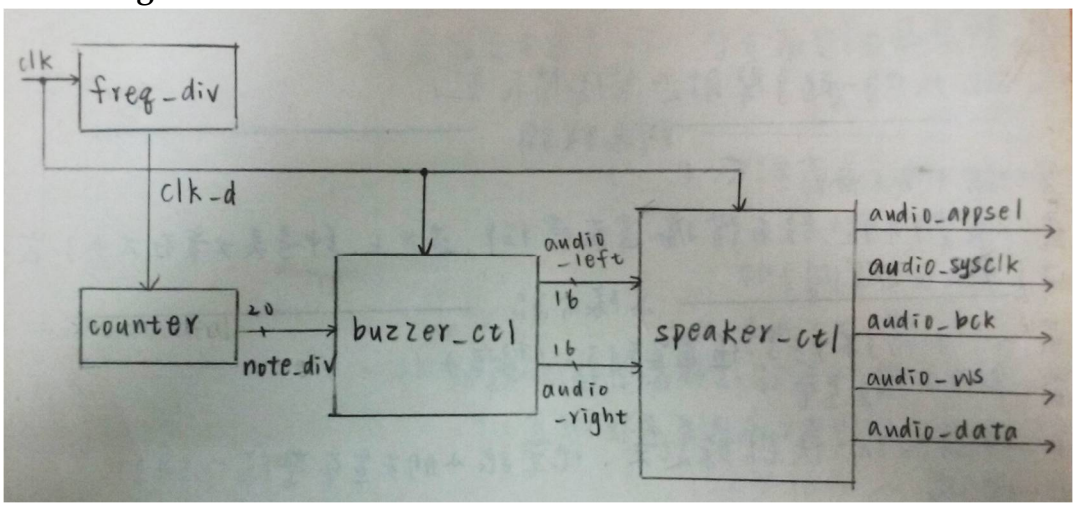
Design Specification

output : audio_appsel, // playing mode selection
audio_sysclk, // control clock for DAC (from crystal)
audio_bck, // bit clock of audio data (5MHz)
audio_ws, // left/right parallel to serial control
audio_data

input : clk, // clock from crystal
rst_n, // active low reset

wire : [15:0] audio_in_left, [15:0]audio_in_right, [19:0] note_div, clk_d;

block diagram:



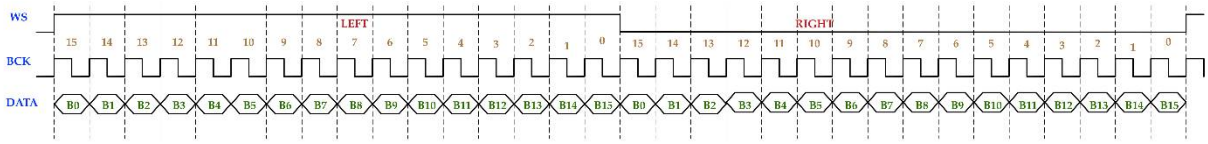
Design Implementation

logic function / logic diagram:

speaker_ctl:

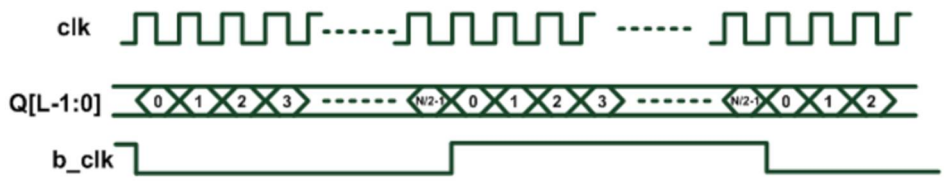
- *Frequency dividers*
  - *audio_bck*
  - *audio_ws*

- *Parallel to serial module*
  - *To re-formulate the audio sequence*
    - *Right first, then left*
    - *MSB first*



buzzer_ctl:

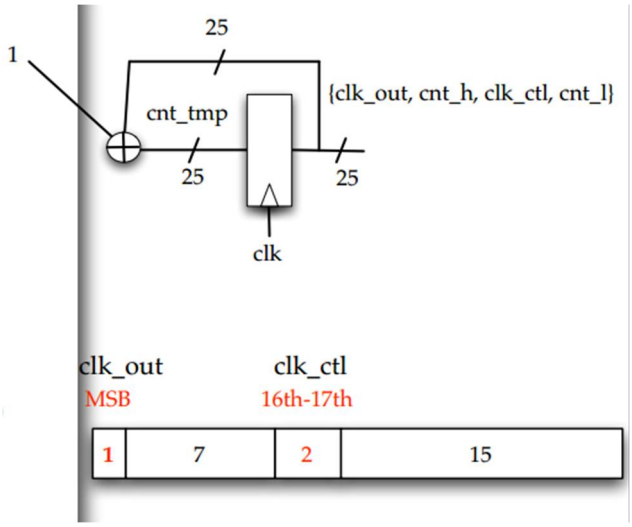
- *The buzzer frequency is obtained by dividing crystal frequency 40MHz by N.*
- *The buzzer clock (b_clk) is periodically inverted for every N/2 clock cycles. (determine the sound)*



Counter:

設置一 4bits 的 Value 從 0 開始數到 15，不同 value 控制不同的頻率，以表現 16 個不同的音。

freq_div:



I/O pin assignment:

```
NET "audio_appsel" LOC = H18;
NET "audio_data" LOC = L16;
NET "audio_sysclk" LOC = H17;
NET "audio_ws" LOC = L15;
NET "clk" LOC = R10;
NET "rst_n" LOC = N3;
NET "audio_bck" LOC = K16;
```

Discussion:

設置一 counter 使其自動改變音高並結合 speaker_ctl 和 buzzer_ctl 產生一段旋律。

2 Electronic Organ

2.1 Integrate the keypad as the keyboard of the electronic organ. Keys A, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, B, C, D, E, F represent the sounds from low to high frequencies.

2.2 Display your playing sound (Do, Re, Mi, Fa, So, La, Si) in the 14-segment LED

Design Specification

output : audio_appsel // playing mode selection
audio_sysclk // control clock for DAC (from crystal)
audio_bck // bit clock of audio data (5MHz)
audio_ws // left/right parallel to serial control
audio_data
[3:0] ftsd_ctl
[14:0] display
Pressed
[3:0] row_n

input : clk // clock from crystal
rst_n // active low reset
[3:0] col_n

wire : clk_d;
[1:0] ftsd_ctl_en;
[3:0] ftsd_in;
[3:0] key;

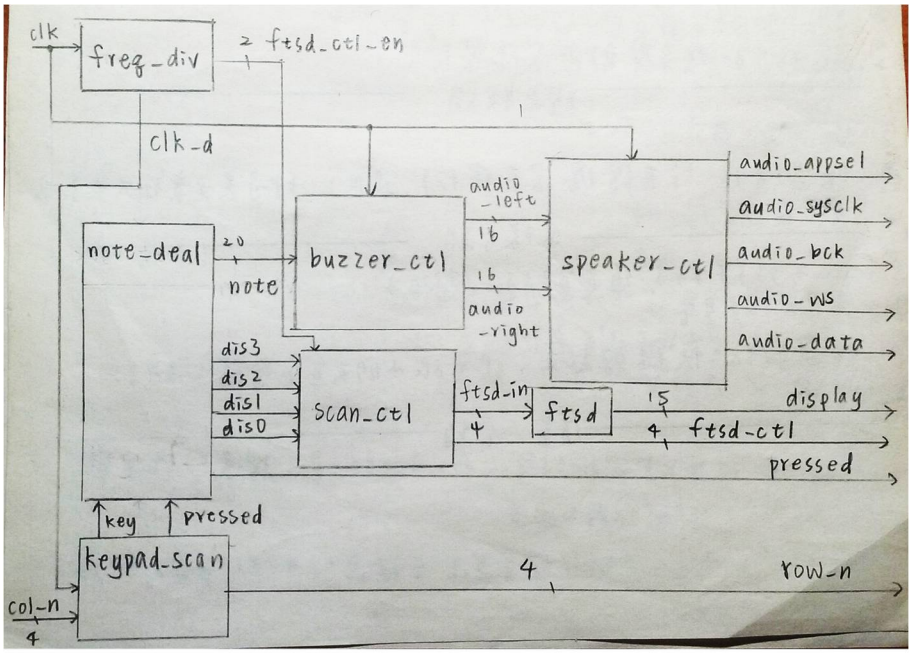
[15:0] audio_in_left;

[15:0] audio_in_right;

[19:0] note;

[3:0] dis3, dis2, dis1, dis0;

block diagram:

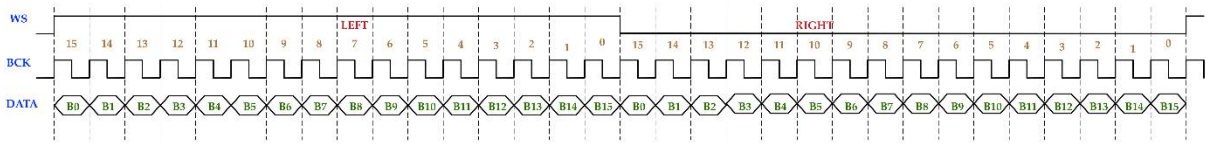


Design Implementation

logic function / logic diagram:

speaker_ctl:

- *Frequency dividers*
  - *audio_bck*
  - *audio_ws*
- *Parallel to serial module*
  - *To re-formulate the audio sequence*
    - *Right first, then left*
    - *MSB first*

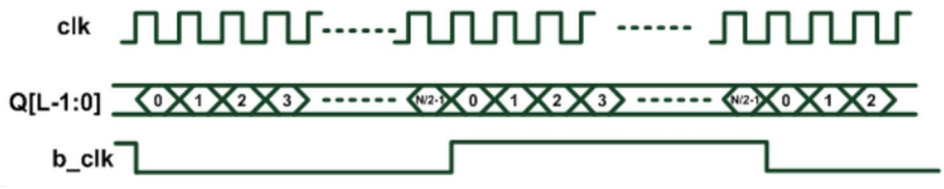


buzzer_ctl:

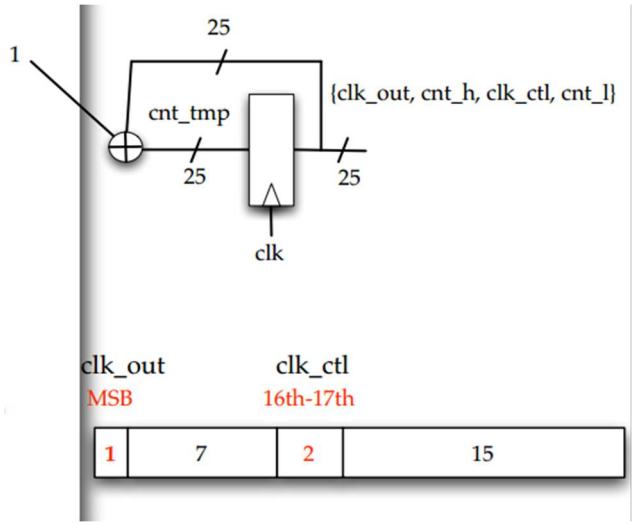
- *The buzzer frequency is obtained by dividing crystal frequency 40MHz by*

N.

• The buzzer clock (b_clk) is periodically inverted for every N/2 clock cycles. (determine the sound)



freq_div:



將 clk_out 改取第 17bit 以加快控制 key_scan 的 clk。

scan_ctl

if ftsd_ctl_en=00

->控制第一個 14 段顯示器，顯示第一種狀況

if ftsd_ctl_en=01

->控制第二個 14 段顯示器，顯示第二種狀況

if ftsd_ctl_en=10

->控制第三個 14 段顯示器，顯示第三種狀況

if ftsd_ctl_en=11

->控制第四個 14 段顯示器，顯示第四種狀況

ftsd

當 level 大於 9，個位數 in1 與十位數 in2 分別控制 ftsd

當 bcd=4'd0: display = 15'b0000_0011_1111_111; //0

當 bcd=4'd1: display = 15'b1111_1111_1011_011; //1

當 bcd=4'd2: display = 15'b0010_0100_1111_111; //2
當 bcd=4'd3: display = 15'b0000_1100_1111_111; //3
當 bcd=4'd4: display = 15'b1001_1000_1111_111; //4
當 bcd=4'd5: display = 15'b0100_1000_1111_111; //5
當 bcd=4'd6: display = 15'b0100_0000_1111_111; //6
當 bcd=4'd7: display = 15'b0001_1111_1111_111; //7
當 bcd=4'd8: display = 15'b0000_0000_1111_111; //8
當 bcd=4'd9: display = 15'b0000_1000_1111_111; //9
default: display = 15'b1111_1111_1111_111; //DEF

keypad_scan
{row_n,col_n}, the mapping:

```
// 1st row              // 3rd row
0111_0111 => F          1101_0111 => A
0111_1011 => E          1101_1011 => 2
0111_1101 => D          1101_1101 => 5
0111_1110 => C          1101_1110 => 8

// 2nd row              // 4th row
1011_0111 => B          1110_0111 => 0
1011_1011 => 3          1110_1011 => 1
1011_1101 => 6          1110_1101 => 4
1011_1110 =>9           1110_1110 =>7
```

當 state 為 pause 且 key_delay 為 15 時，若仍有 pressed_detected，則 next_state 仍為 pause，且持續感應 pressed_detected，以確保長按時仍能發出穩定的音。

note_deal
但按下按鈕時使不同按鈕控制不同頻率的音。

I/O pin assignment:

```
NET "display[0]" LOC = U5;
NET "display[1]" LOC = T7;
NET "display[2]" LOC = R7;
NET "display[3]" LOC = V7;
NET "display[4]" LOC = V4;
NET "display[5]" LOC = T4;
NET "display[6]" LOC = T3;
NET "display[7]" LOC = R5;
NET "display[8]" LOC = N5;
NET "display[9]" LOC = R3;
```

```
NET "display[10]" LOC = U7;
NET "display[11]" LOC = T5;
NET "display[12]" LOC = V5;
NET "display[13]" LOC = N4;
NET "display[14]" LOC = P6;
NET "ftsd_ctl[0]" LOC = V8;
NET "ftsd_ctl[1]" LOC = U8;
NET "ftsd_ctl[3]" LOC = T6;
NET "ftsd_ctl[2]" LOC = V6;
NET "row_n[3]" LOC = L3;
NET "row_n[2]" LOC = L4;
NET "row_n[1]" LOC = K1;
NET "row_n[0]" LOC = K2;
NET "clk" LOC = R10;
NET "rst_n" LOC = N3;
NET "pressed" LOC = H5;
NET "audio_appsel" LOC = H18;
NET "audio_bck" LOC = K16;
NET "audio_data" LOC = L16;
NET "audio_sysclk" LOC = H17;
NET "audio_ws" LOC = L15;
NET "col_n[3]" LOC = H1;
NET "col_n[2]" LOC = H2;
NET "col_n[1]" LOC = J1;
NET "col_n[0]" LOC = J3;
```

Discussion:

設定不同 key 發出不同音，並調整長按時可繼續維持同樣的 state。

Conclusion:

此實驗主要是設定不同音的頻率再結合 counter 或 keypad_scan 做變換!