`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////

// Company:

// Engineer:

//

// Create Date: 2019/07/03 13:41:06

// Design Name:

// Module Name: Matrix\_keyboard

// Project Name:

// Target Devices:

// Tool Versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

//////////////////////////////////////////////////////////////////////////////////

module matrixKeyboard\_drive(

input i\_clk, // 时钟信号输入，100MHZ

output reg buzzer\_out, // 蜂鸣器输出

input i\_rst\_n, // 复位信号

input [3:0] row, // 矩阵键盘 行

output reg [3:0] col, // 矩阵键盘 列

output reg [6:0] keyboard\_val // 键盘值

);

//++++++++++++++++++++++++++++++++++++++

// 分频部分 开始

//++++++++++++++++++++++++++++++++++++++

reg [19:0] cnt; // 计数子20位

reg [18:0] cnt\_beep; // 计数子19位

always @ (posedge i\_clk, negedge i\_rst\_n)

if (!i\_rst\_n) // 复位信号为0（低有效），cnt=0

cnt <= 0;

else // 复位信号为1（低有效），时钟信号上升沿到来时，cnt+1

cnt <= cnt + 1'b1;

wire key\_clk = cnt[19]; // T = 2^20/100M = 10.48576ms f = 95.367HZ

always @ (posedge i\_clk) //利用cnt\_beep对buzzer\_out的输出频率进行控制,对应八个音符

begin

cnt\_beep <= cnt\_beep + 1'b1;

if({col\_val, row\_val} == 8'b1110\_1110)//1按下,do,频率：256

begin

if(cnt\_beep <= 19'd190839)

buzzer\_out <= 1'b1;

else if(cnt\_beep <= 19'd381679)

buzzer\_out <= 1'b0;

else

cnt\_beep <= 1'b0;

end

else if({col\_val, row\_val} == 8'b1101\_1110)//2按下,re,频率：288

begin

if(cnt\_beep <= 19'd170068) //

buzzer\_out <= 1'b1;

else if(cnt\_beep <= 19'd340136)

buzzer\_out <= 1'b0;

else

cnt\_beep <= 1'b0;

end

else if({col\_val, row\_val} == 8'b1011\_1110)//3按下,mi,频率：320

begin

if(cnt\_beep <= 19'd151515) //

buzzer\_out <= 1'b1;

else if(cnt\_beep <= 19'd303030)

buzzer\_out <= 1'b0;

else

cnt\_beep <= 1'b0;

end

else if({col\_val, row\_val} == 8'b1110\_1101)//4按下,fa,频率：341

begin

if(cnt\_beep <= 19'd143266) //

buzzer\_out <= 1'b1;

else if(cnt\_beep <= 19'd286532)

buzzer\_out <= 1'b0;

else

cnt\_beep <= 1'b0;

end

else if({col\_val, row\_val} == 8'b1101\_1101)//5按下,sol,频率：384

begin

if(cnt\_beep <= 19'd127551) //

buzzer\_out <= 1'b1;

else if(cnt\_beep <= 19'd255102)

buzzer\_out <= 1'b0;

else

cnt\_beep <= 1'b0;

end

else if({col\_val, row\_val} == 8'b1011\_1101)//6按下,la,频率：426

begin

if(cnt\_beep <= 19'd113636) //

buzzer\_out <= 1'b1;

else if(cnt\_beep <= 19'd227272)

buzzer\_out <= 1'b0;

else

cnt\_beep <= 1'b0;

end

else if({col\_val, row\_val} == 8'b1110\_1011)//7按下,si,频率：480

begin

if(cnt\_beep <= 19'd101214) //

buzzer\_out <= 1'b1;

else if(cnt\_beep <= 19'd202429)

buzzer\_out <= 1'b0;

else

cnt\_beep <= 1'b0;

end

else if({col\_val, row\_val} == 8'b1101\_1011)//8按下,do,频率：512

begin

if(cnt\_beep <= 19'd95602) //

buzzer\_out <= 1'b1;

else if(cnt\_beep <= 19'd191204)

buzzer\_out <= 1'b0;

else

cnt\_beep <= 1'b0;

end

end

//--------------------------------------

// 分频部分 结束

//--------------------------------------

//++++++++++++++++++++++++++++++++++++++

// 状态机部分 开始

//++++++++++++++++++++++++++++++++++++++

// 状态数较少，独热码编码

parameter NO\_KEY\_PRESSED = 6'b000\_001; // 没有按键按下

parameter SCAN\_COL0 = 6'b000\_010; // 扫描第0列

parameter SCAN\_COL1 = 6'b000\_100; // 扫描第1列

parameter SCAN\_COL2 = 6'b001\_000; // 扫描第2列

parameter SCAN\_COL3 = 6'b010\_000; // 扫描第3列

parameter KEY\_PRESSED = 6'b100\_000; // 有按键按下

reg [5:0] current\_state, next\_state; // 现态、次态

always @ (posedge key\_clk, negedge i\_rst\_n)

if (!i\_rst\_n) //复位信号为0（低有效），当前状态设置为NO\_KEY\_PRESSED

current\_state <= NO\_KEY\_PRESSED;

else

begin

current\_state <= next\_state;

end

// 根据条件转移状态

always @ \*

case (current\_state)

NO\_KEY\_PRESSED : // 没有按键按下

if (row != 4'hF)

next\_state = SCAN\_COL0;

else

next\_state = NO\_KEY\_PRESSED;

SCAN\_COL0 : // 扫描第0列

if (row != 4'hF)

next\_state = KEY\_PRESSED;

else

next\_state = SCAN\_COL1;

SCAN\_COL1 : // 扫描第1列

if (row != 4'hF)

next\_state = KEY\_PRESSED;

else

next\_state = SCAN\_COL2;

SCAN\_COL2 : // 扫描第2列

if (row != 4'hF)

next\_state = KEY\_PRESSED;

else

next\_state = SCAN\_COL3;

SCAN\_COL3 : // 扫描第3列

if (row != 4'hF)

next\_state = KEY\_PRESSED;

else

next\_state = NO\_KEY\_PRESSED;

KEY\_PRESSED : // 有按键按下

if (row != 4'hF)

next\_state = KEY\_PRESSED;

else

next\_state = NO\_KEY\_PRESSED;

endcase

reg key\_pressed\_flag; // 键盘按下标志

reg [3:0] col\_val, row\_val; // 列值、行值

// 根据次态，给相应寄存器赋值

always @ (posedge key\_clk, negedge i\_rst\_n)

if (!i\_rst\_n)

begin

col <= 4'h0;

col\_val <= 4'b1111; // 清空列值

row\_val <= 4'b1111; // 清空行值

key\_pressed\_flag <= 0;

end

else

case (next\_state)

NO\_KEY\_PRESSED : // 没有按键按下

begin

col <= 4'h0;

col\_val <= 4'b1111; // 清空列值

row\_val <= 4'b1111; // 清空行值

key\_pressed\_flag <= 0; // 清键盘按下标志

end

SCAN\_COL0 : // 扫描第0列

col <= 4'b1110;

SCAN\_COL1 : // 扫描第1列

col <= 4'b1101;

SCAN\_COL2 : // 扫描第2列

col <= 4'b1011;

SCAN\_COL3 : // 扫描第3列

col <= 4'b0111;

KEY\_PRESSED : // 有按键按下

begin

col\_val <= col; // 锁存列值

row\_val <= row; // 锁存行值

key\_pressed\_flag <= 1; // 置键盘按下标志

end

endcase

//--------------------------------------

// 状态机部分 结束

//--------------------------------------

//++++++++++++++++++++++++++++++++++++++

// 扫描行列值部分 开始

//++++++++++++++++++++++++++++++++++++++

always @ (posedge key\_clk, negedge i\_rst\_n)

if (!i\_rst\_n)

keyboard\_val <= 4'h0;

else

if (key\_pressed\_flag)

case ({col\_val, row\_val})

8'b1110\_1110 : keyboard\_val <= 7'b0000001;

8'b1110\_1101 : keyboard\_val <= 7'b1001100;

8'b1110\_1011 : keyboard\_val <= 7'b0000000;

8'b1110\_0111 : keyboard\_val <= 7'b1110010;

8'b1101\_1110 : keyboard\_val <= 7'b1001111;

8'b1101\_1101 : keyboard\_val <= 7'b0100100;

8'b1101\_1011 : keyboard\_val <= 7'b0001100;

8'b1101\_0111 : keyboard\_val <= 7'b1000010;

8'b1011\_1110 : keyboard\_val <= 7'b0010010;

8'b1011\_1101 : keyboard\_val <= 7'b0100000;

8'b1011\_1011 : keyboard\_val <= 7'b0001000;

8'b1011\_0111 : keyboard\_val <= 7'b0110000;

8'b0111\_1110 : keyboard\_val <= 7'b0000110;

8'b0111\_1101 : keyboard\_val <= 7'b0001111;

8'b0111\_1011 : keyboard\_val <= 7'b1100000;

8'b0111\_0111 : keyboard\_val <= 7'b0111000;

endcase

//--------------------------------------

// 扫描行列值部分 结束

//--------------------------------------

endmodule