

选课时间段: 周五 6-8 节

序号 (座位号): \_\_\_\_\_

# 杭州电子科技大学

## 实 验 报 告

课程名称: EDA 技术

实验名称: 乐曲硬件演奏电路设计

指导老师: 岳克强

学生姓名: 黄继升

学生学号: 16041321

学生班级: 16040313

所学专业: 电子信息工程

实验日期: 2018.1.5

## 一.实验目的

学习设计硬件乐曲演奏电路以及相关的控制电路

## 二.实验仪器设备或关键器材

1. Quartus II 软件
2. EDA 实验箱上的 FPGA 开发板

## 三.实验原理

硬件乐曲演奏电路的顶层模块见实验内容中的原理电路图设计。

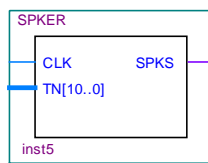
- (1) 音符的频率由图中的 SPKER 获得，这是一个用作分频器的可预置计数器；
- (2) 音符的持续时间根据乐曲的速度及每个音符的节拍数来确定；模块 F\_CODE 为模块 SPKER（11 位分频器）提供决定所发音符的分频预置数，并且是乐曲简谱码对应的分频预置数查表电路。
- (3) 模块 MUSIC 是一个 LPM\_ROM，存储“梁祝”乐曲的音符数据。
- (4) 模块 CNT138T 是一个 8 位二进制计数器，内部设置计数最大值为 139，作为音符数据 ROM 的地址发生器；
- (5) 锁相环 PLL20 将 22KHz 的高频时钟信号进行分频为 2KHz 的输出频率；
- (6) 分频模块 FDIV 继续分频为 CNT138T 和 ROM 模块 MUSIC 提供 4Hz 的输入频率；

#### 四.实验内容以及操作：

1. 用作分频器的可预置计数器设计：获得音符的频率  
SPKER 代码：

```
1 module SPKER(CLK,TN,SPKS);
2   input CLK;
3   input[10:0] TN;
4   output SPKS;
5   reg SPKS;reg[10:0] CNT11;
6   always@(posedge CLK) begin: CNT11B_LOAD
7     if(CNT11==11'h7FF) begin CNT11=TN; SPKS<=1'b1; end
8     else begin CNT11=CNT11+1; SPKS<=1'b0; end
9   end
10 endmodule
```

生成元件：

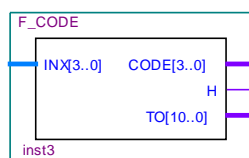


2. 制作一个模块 F\_CODE，首先为模块 SPKER（11 位分频器）提供决定所发音符的分频预置数，而此数在 SPKER 输入口停留的时间即为此音符的节拍周期。然后根据电子琴音阶基频对照图设置“梁祝”乐曲全部音符所对应的分频预置数，共 14 个。

F\_CODE 代码：

```
1 module F_CODE(INX,CODE,H,TO);
2   input[3:0] INX; output[3:0] CODE; output H; output[10:0] TO;
3   reg[10:0] TO; reg[3:0] CODE; reg H;
4   always@(INX) begin
5     case(INX)
6       0: begin TO<=11'h7FF; CODE<=0; H<=0; end
7       1: begin TO<=11'h305; CODE<=1; H<=0; end
8       2: begin TO<=11'h390; CODE<=2; H<=0; end
9       3: begin TO<=11'h40C; CODE<=3; H<=0; end
10      4: begin TO<=11'h45C; CODE<=4; H<=0; end
11      5: begin TO<=11'h4AD; CODE<=5; H<=0; end
12      6: begin TO<=11'h50A; CODE<=6; H<=0; end
13      7: begin TO<=11'h55C; CODE<=7; H<=0; end
14      8: begin TO<=11'h582; CODE<=1; H<=1; end
15      9: begin TO<=11'h5C8; CODE<=2; H<=1; end
16      10: begin TO<=11'h606; CODE<=3; H<=1; end
17      11: begin TO<=11'h640; CODE<=4; H<=1; end
18      12: begin TO<=11'h656; CODE<=5; H<=1; end
19      13: begin TO<=11'h684; CODE<=6; H<=1; end
20      14: begin TO<=11'h69A; CODE<=7; H<=1; end
21      15: begin TO<=11'h6C0; CODE<=1; H<=1; end
22    default : begin TO<=11'h6C0; CODE<=1;H<=1;
23    end
24  endcase end
25 endmodule
```

生成元件：



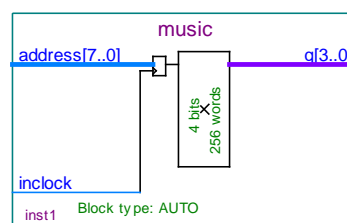
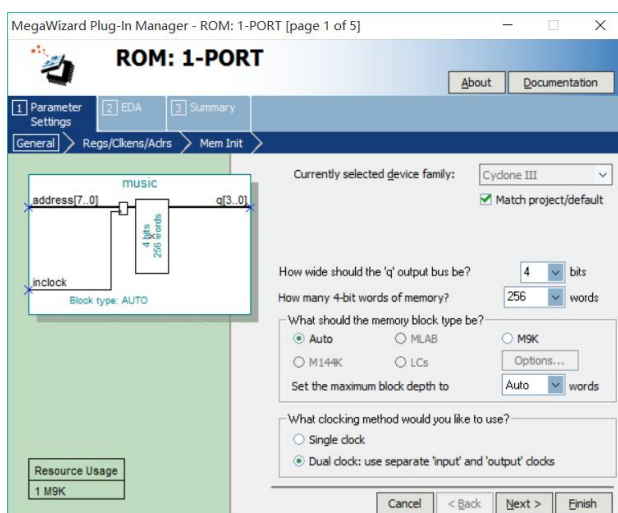
### 3. 定制音符数据 “liangzhu.mif”

Addr	+0	+1	+2	+3	+4	+5	+6	+7
0	3	3	3	3	5	5	5	6
8	8	8	8	9	6	8	5	5
16	12	12	12	15	13	12	10	12
24	9	9	9	9	9	9	9	0
32	9	9	9	10	7	7	6	6
40	5	5	5	6	8	8	9	9
48	3	3	8	8	6	5	6	8
56	5	5	5	5	5	5	5	5
64	10	10	10	12	7	7	9	9
72	6	8	5	5	5	5	5	5
80	3	5	3	3	5	6	7	9
88	6	6	6	6	6	6	5	6
96	8	8	8	9	12	12	12	10
104	9	9	10	9	8	8	6	5
112	3	3	3	3	8	8	8	8
120	6	8	6	5	3	5	6	8
128	5	5	5	5	5	5	5	5

4. 每一个音符的停留时间则由音乐节拍和音调发生查表模块 MUSIC 中简谱码和工作时钟 incock 的频率决定，在此为 4HZ。模块 MUSIC 是一个 LPM\_ROM，存储“梁祝”乐曲的音符数据。

模块 MUSIC 设置：

输出 q 位宽定义为 4 位，数据深度为 256(即 8 位数据线)。对应 CycloneIII，存储器构建方式选择 M9K，再选择双时钟方式，点击 Next，去掉选项“q output port”前的钩，即选择时钟只控制锁存输入信号。点击 Next，导入制作好的“liangzhu.mif”音符数据文件，不断点击 Next 最后完成设置，生成一个 LPM\_ROM 元件 MUSIC。



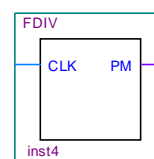
### 5. 4Hz 的频率来自分频模块 FDIV。

模块 FDIV 代码：

```

1 module FDIV (CLK, PM);
2 input CLK;
3 output PM; reg[8:0] Q1; reg FULL; wire RST;
4 always@(posedge CLK or posedge RST)
5 begin
6 if(RST) begin Q1<=0; FULL<=1;
7 end
8 else begin Q1<=Q1+1; FULL<=0;
9 end end
10 assign RST=(Q1==499);
11 assign PM=FULL;
12 assign DOUT=Q1;
13 endmodule

```



6. 设置一个 8 位二进制计数器模块 CNT138T，内部设置计数最大值为 139，作为音符数据 ROM 的地址发生器。

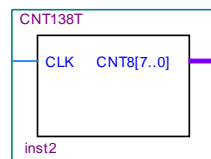
CNT138T 代码：

```

1 module CNT138T(CLK,CNT8);
2   input CLK;
3   output[7:0] CNT8;
4   reg[7:0] CNT;
5   wire LD;
6   always@(posedge CLK or posedge LD)
7   begin
8     if(LD) CNT<=8'b00000000;
9     else CNT<=CNT+1;
10  end
11  assign CNT8=CNT;
12  assign LD= (CNT==138);
13 endmodule

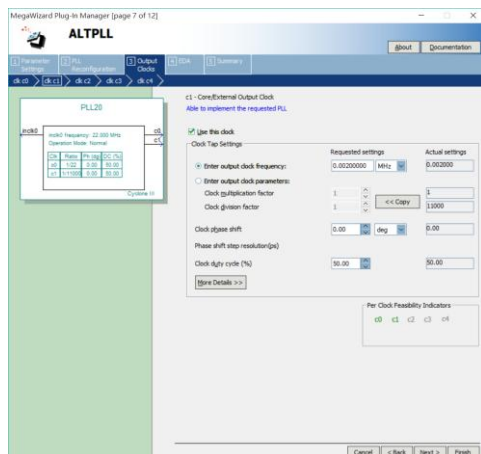
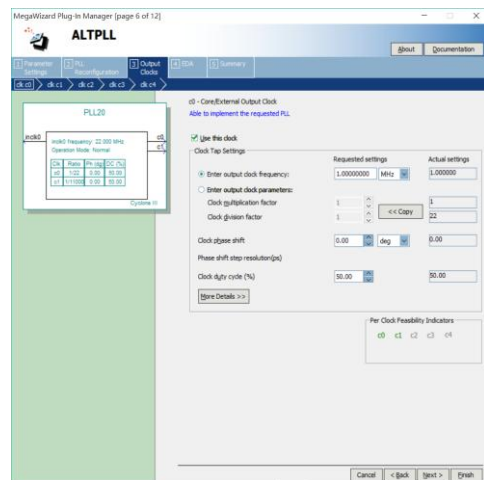
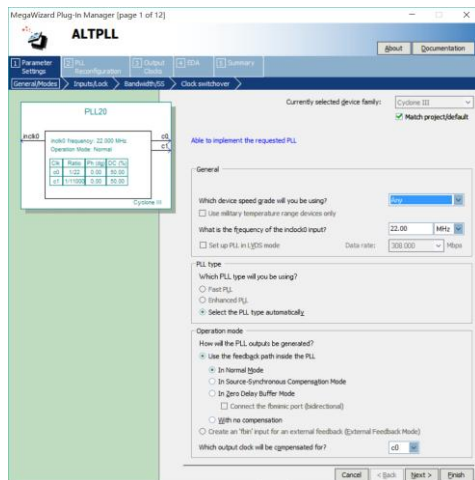
```

生成元件：

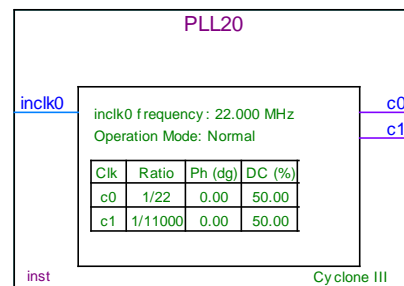


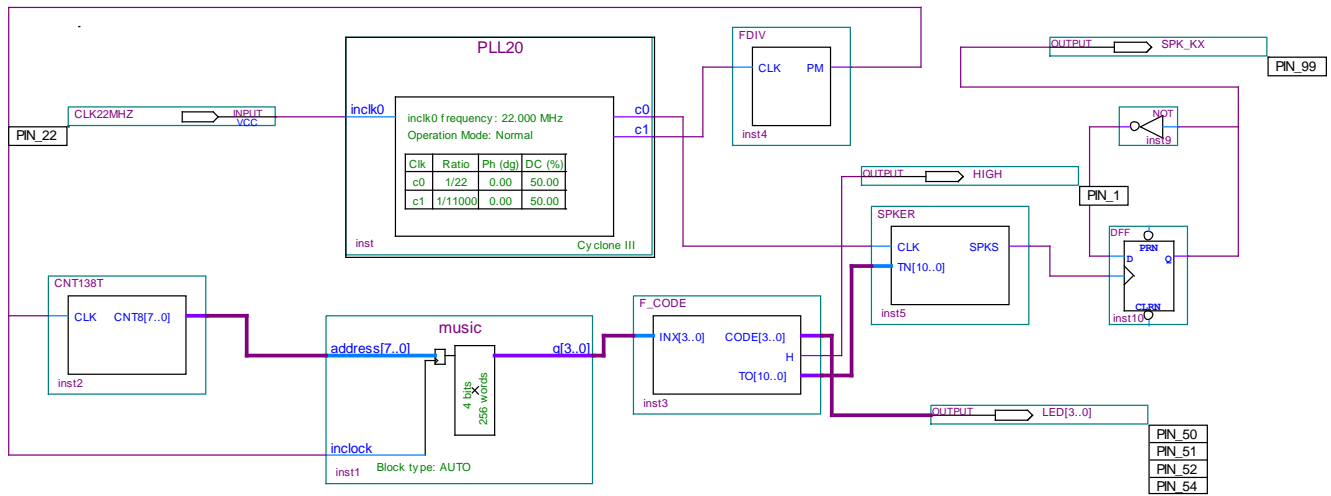
7. 提供 2kHz 输入频率的锁相环 PLL20 的设置：








原始输入频率设置为与 FPGA 对应的时钟频率 22MHz，输出时钟信号设置两个 C0 和 C1。其中 C0 频率定义为 1MHz，即分频为 1/22，C1 频率定义为 0.002MHz，即分频为 1/11000。



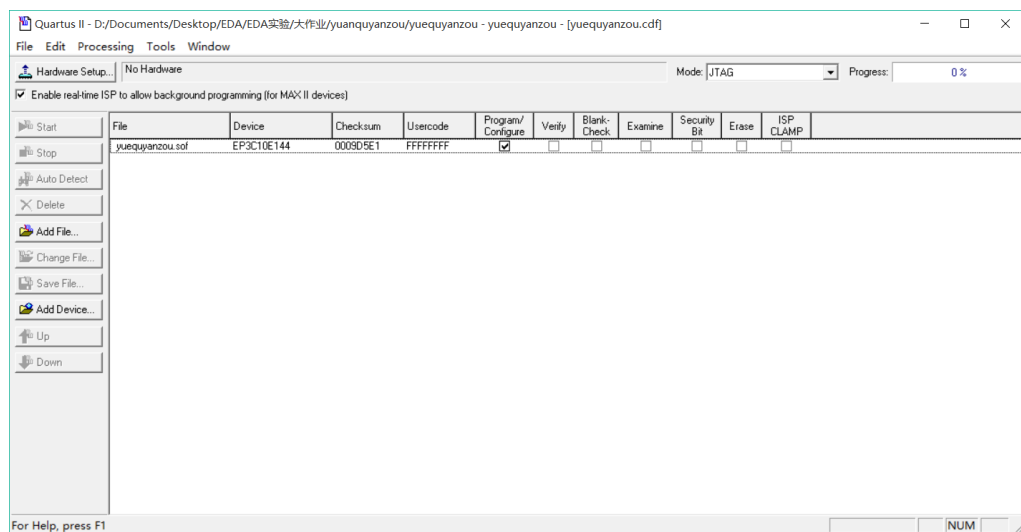
生成元件：





	Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard
1	 CLK22MHZ	Input	PIN_22	1	B1_N0	2.5 V (default)
2	 HIGH	Output	PIN_1	1	B1_N0	2.5 V (default)
3	 LED[3]	Output	PIN_50	3	B3_N0	2.5 V (default)
4	 LED[2]	Output	PIN_51	3	B3_N0	2.5 V (default)
5	 LED[1]	Output	PIN_52	3	B3_N0	2.5 V (default)
6	 LED[0]	Output	PIN_54	4	B4_N0	2.5 V (default)
7	 SPK_KX	Output	PIN_99	6	B6_N0	2.5 V (default)

SPK\_KX 对应管脚 PIN\_99



**硬件测试现象：**

蜂鸣器发出清晰的《梁祝》乐曲, HIGH 为高八度音指示, 可由发光管指示。

**实验感想：**

这次期末大作业是最后一次 EDA 实验, 算是对本学期 EDA 实验的总结吧。整个实验其实很轻松, 因为课本已经将最难的原理图设计部分直接给你了, 只要你懂得了整个乐曲演奏电路的原理, 那么各个模块元件的设置并不是什么难事。这次乐曲硬件演奏电路的设计涉及到了本学期 EDA 实验操作的许多方面, 包括对分频器代码的编写, 对计数器代码的编写, 对蜂鸣器代码的编写, 以及设置锁相环对高频时钟信号进行分频输出, 还有 LPM\_ROM 模块的设置来存储音符数据。通过这次实验, 我也对之前的实验的模块设置理解更深, 也运用得更为熟练。希望我从本学期 EDA 实验学到的东西能够帮助我得到自身能力的更大提升, 同时对我将来掌握更深的电子设计和软件设计立下坚实的基础。