数字电路与数字系统实验

实验十

音频输出

计算机科学与技术系

191220008 陈南曈

924690736@qq.com

2020.

一、实验目的

声音是一种重要的人机交互手段。音频信号可以通过扬声器或者耳机输出，由麦克风输入计算机。在输入和输出过程中一般需要对信号进行数字/模拟或模拟/数字转换。

本实验的主要目的是学习音频信号的输出方式以及如何将数字信号转换为模拟信号的基本原理。

实验内容：

请将之前实验实现的键盘与本实验的音频输出结合，实现一个简单的键盘电子琴功能。钢琴上的不同音高对应着不同的频率，如下表所示。我们可以根据按下的键的键值，决定播放的正弦的频率，从而实现电子琴的功能。

音名/八度	3	4	5	6
C	130.81Hz	261.63 Hz	523.25 Hz	1046.5 Hz
C#	138.59 Hz	277.18 Hz	554.37 Hz	1108.7 Hz
D	146.83 Hz	293.66 Hz	587.33 Hz	1174.7 Hz
D#	155.56 Hz	311.13 Hz	622.25 Hz	1244.5 Hz
E	164.81 Hz	329.63 Hz	659.26 Hz	1318.5 Hz
F	174.61 Hz	349.23 Hz	698.46 Hz	1396.9 Hz
F#	185.00 Hz	369.99 Hz	739.99 Hz	1480.0 Hz
G	196.00 Hz	392.00 Hz	783.99 Hz	1568.0 Hz
G#	207.65 Hz	415.30 Hz	830.61 Hz	1661.2 Hz
A	220.00 Hz	440.00 Hz	880.00 Hz	1760.0 Hz
A#	233.08 Hz	466.16 Hz	932.33 Hz	1864.7 Hz
B	246.94 Hz	493.88 Hz	987.77 Hz	1975.5 Hz

基本要求：

①实现至少 8 个音符，建议可以实现 C5，D5，...，B5，C6 这些音符。

②只需要支持每次按下单个按键。按键按下后开始发音，按键期间持续发音，松开后停止发音。按无关键不发音。

③无杂音和爆破音等。

可选扩展要求：

①可调节音量。

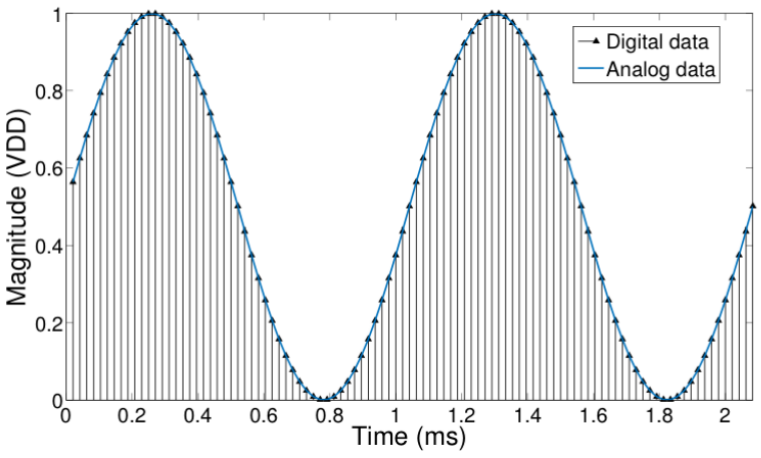
②支持多个键同时按下的和声。例如，同时按下 C5，E5，G5 时发出大三和弦，即对应三个音相加的结果，可以只支持同时发两个音。

二、实验原理（知识背景）

1、音频输出原理

人耳可以听到的声音的频率范围是 20-20kHz。音频设备如扬声器或耳机等所接收的音频信号一般是模拟信号，即时间上连续的信号。但是，由于数字器件只能以固定的时间间隔产生数字输出，我们需要通过数字/模拟转换将数字信号转换成模拟信号输出。根据采样定律，数字信号的采样率（每秒钟产生的数字样本数量）应不低于信号频率的两倍。所以，数字音频一般采用 44.1kHz(CD 音频) 或 48kHz 的采样率，以保证 20kHz 的信号不会失真。

下图显示了数字信号到模拟信号转换的基本原理。在 48kHz 的采样率下，我们每秒输出 48000 次，即每间隔 1/48000 秒（1/48 毫秒）的时间产生一个数字输出样本点，如图中黑色小三角所示。该输出经过平滑后，会产生一个对应的模拟信号。



我们需要在合适的时间点上设置（或输出）合适的数字值来形成正弦波形。对于一个正弦波信号 s(t)，其数学表达式是：

$$s(t) = \sin(2\pi f t) \qquad (10-1)$$

其中 f 为频率，t 为时间。在数字信号中，我们用整数 n 来标记各个数字样本，样本

编号顺序依次为 n = 0, 1, 2, 3, 4, . . .。当采样率是 48kHz 时，每两个点之间的间隔是 1/48 毫秒。此时，我们可以将 t 改写成 t = n/48000 秒。这样式 (10-1) 就变成：

$$s(n) = sin\left(\frac{2\pi fn}{48000}\right) \qquad (10-2)$$

实际信号输出时，我们一般不采用浮点数而选用整数值来表示每个样本点的大小。这个过程称为量化 (Quantization)。假设我们用带符号的 16bit 整数 （补码）来表示单个样本点，此时 32767 即对应输出的最大值（例如 +1V 电压），-32768 即对应输出的最小值（例如-1V 电压）

实际应用中，我们如果要产生不同频率的正弦波，就不能采用简单计数直接查表的方式，而需要先按频率计算出样本点对应的相位，然后查三角函数表获取对应幅度值的方式。

首先我们需要存储器中存储一张 1024 点的 sin 函数表。即存储器中以地址 k =0. . .1023 存储了 1024 个三角函数值（以 16bit 补码整数表示），地址为 k 的数值设置为

$$round\left(\sin\left(\frac{2\pi k}{1024}\right)\times 32767\right) \qquad (10-3)$$

感兴趣的同学可以用高级语言生成该函数表，我们也提供了一张 1024 点 sin 函数表的 mif 文件。

对于任意频率为 f 的正弦波，我们在第 n 个样本点需要输出的值为

$$round\left(\sin\left(\frac{2\pi nf}{48000}\right)\times 32767\right) \qquad (10-4)$$

比较式 (10-3) 和 (10-4)，我们得到 $\frac{k}{1024}=\frac{nf}{48000}$。因此，在我们的函数表中最接近这个值的表项应该是 $k = round\left(nf\times\frac{1024}{48000}\right) mod\ 1024$。函数表中的项目越多， 查找到的函数值越精确，但一般情况下人的耳朵往往可以容忍 1%-5% 的误差， 所以我们使用 1024 点的函数表已经基本够用。 但是，在 FPGA 中要计算乘除法及取整操作耗费资源较多，我们实际应用中采取累加的方法。我们观察到 n 每增加 1，对应的 k 会增加 $f\times\frac{1024}{48000}$ 。这样，我们可以通过每个样本点将 k 递增 $f\times\frac{1024}{48000}$ 来避免乘除法。例如，f = 960Hz 时， 我们可以每个样本点对 k 递增 $960\times\frac{1024}{48000}=20.48$。这样，50 个点后正好 k 增加了 1024，完成一个周期。但是，我们这里的 k 是整数，如何能够每次递增一个小数值呢？这里，我们可以采用定点小数的方式，即用 16bit 来表示 k。其中前 10bit 是整数部分，用来查三角函数表，后面 6bit 是小数部分，用来提高精度。这时，如果将此 16bit 数看成是一个无符号整数的话，每个周期是 65536， 而对应前 10 比特循环的周期是 1024 点。因此，n 每增加 1，我们需要 k 递增 $960\times\frac{65536}{48000}=1311$，对应的小数值是 $\frac{1311}{64}=20.4843$。从整

数的角度来看，n 变化 50 个样本点时 k 递增了 $1311 \times 50 = 65550$，这个值略大于 65536 一些，会对周期带来一些小的误差，但是这样的误差对于人耳来说是可以容忍的。

因此，生成频率为 f 的正弦波的过程如下

1. 根据频率 f 计算递增值 $d = f \times \frac{65536}{48000}$ 。
2. 在系统中维持一个 16bit 无符号整数计数器，每个样本点递增 d 。
3. 根据 16 位无符号整数计数器的高 10 位来获取查表地址 k，并查找 1024 点的正弦函数表。
4. 使用查表结果作为当前的数字输出。

2、音频接口

DE10-Standard 开发板上集成了一块 WM8731 音频编解码芯片。该音频编解码芯片提供 24bit 的音频接口，支持 8kHz 到 96kHz 的采样率。在我们的实验中仅考虑 48kHz 采样率，每个样本点 16 比特的情况。

FPGA 通过音频数字模拟转换（DA）和模拟数字转换（AD）接口来与音频编解码芯片通信。FPGA 输出的数字样本信号被转换成模拟信号（DA 转换），通过绿色的 Line Out 接口输出。在实验中请使用耳机接入 Line Out 接口。同时，系统还支持通过 Mic In 或 Line In 接口接入麦克风或模拟信号，将其转换为数字信号（AD 转换）然后发送给 FPGA。

FPGA 与音频编解码芯片的接口包括两大部分。一部分是控制接口，该接口是利用通用 I²C 总线实现的。该接口的功能主要是在音频编解码芯片的控制寄存器内写入配置信息，控制音频编解码芯片的工作方式。另一部分是音频信号接口，主要是通过 I²S 音频协议实现的，包括 DAC 和 ADC 两个方向。在本实验中只使用 DAC 输出音频信号的方向。

三、实验环境/器材等

1) 软件环境:

Quartus (Quartus Prime 17.1) Lite Edition

2) 硬件环境:

DE10-Standard 开发板
FPGA 部分:
Intel Cyclone V SE 5CSXFC6D6 F31C6N

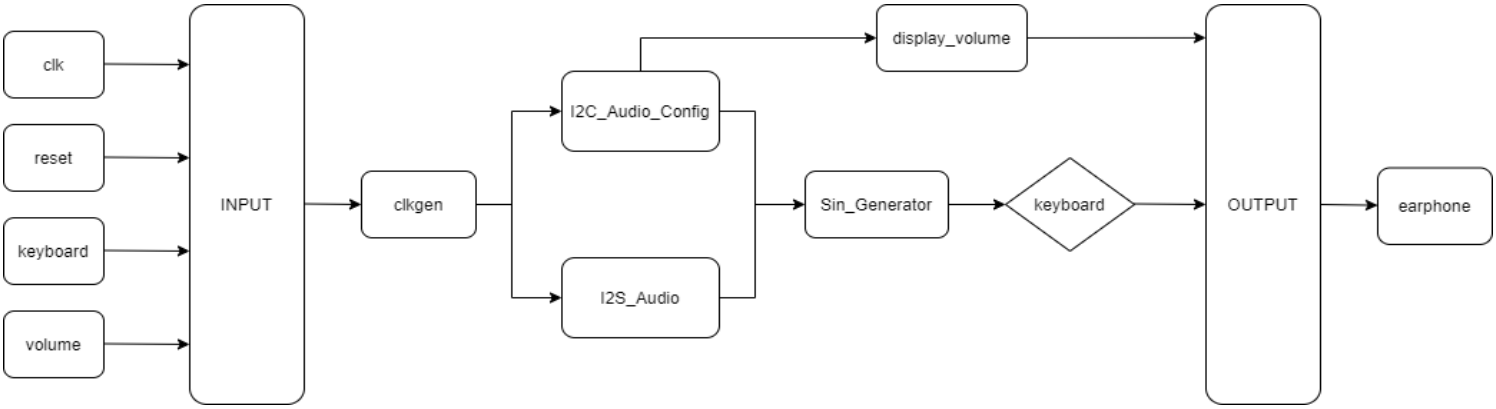
- 110K 逻辑单元
- 5,761Kbit RAM

HPS 部分:

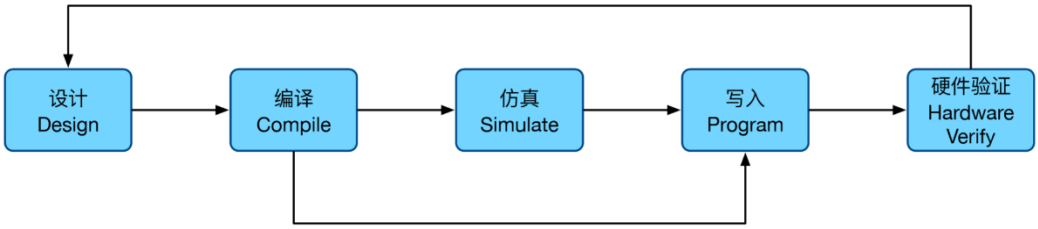
Dual-core ARM Cortex A9

- 925MHz
- 1GB DDR

四、程序代码或流程图



五、实验步骤/过程

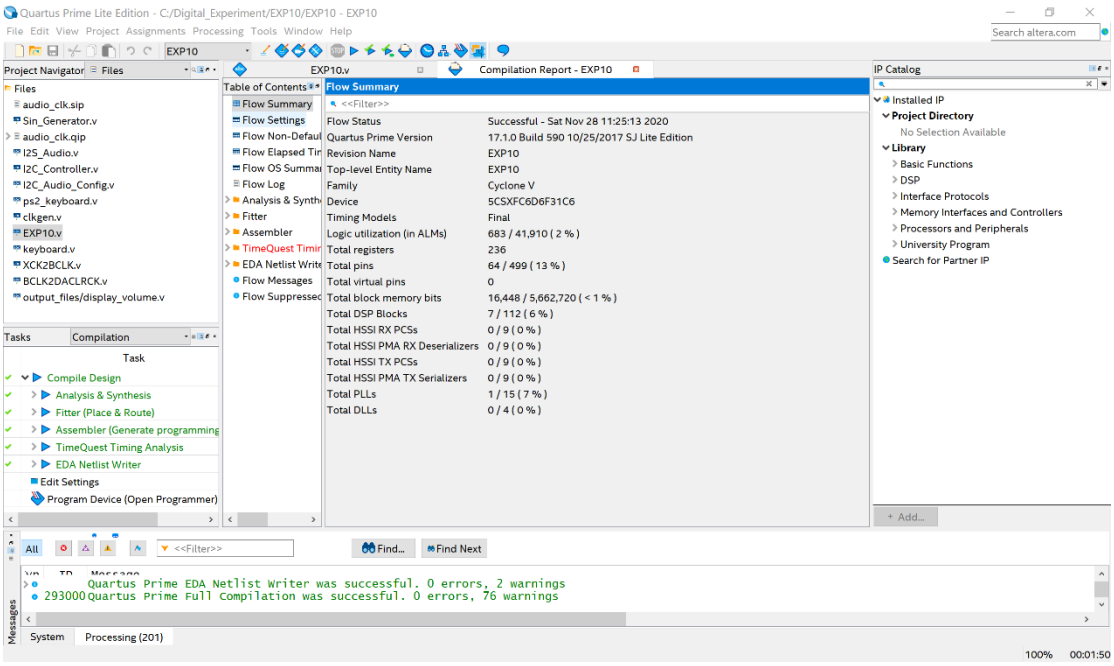


设计：

```
audio_clk my_aud_clk(CLOCK_50, reset, AUD_XCK, LEDR[9]);
//XCK2BCLK my_bclk(AUD_XLK, AUD_BCLK);
//BCLK2DACLRCK my_daclrck(AUD_XLK, AUD_BCLK);
//I2C part
clkgen #(10000) my_i2c_clk(CLOCK_50, reset, 1'b1, clk_i2c);  //10k I2C clock
I2C_Audio_Config my_i2c(clk_i2c, ~reset, I2C_SCLK, I2C_SDAT, VOLUME, vol);
I2S_Audio my_i2s(AUD_XCK, ~reset, AUD_BCLK, AUD_DACDAT, AUD_DACLRCK, audio_data);
Sin_Generator my_sin(AUD_DACLRCK, ~reset, freq, audio_data);
keyboard my_keyboard(CLOCK_50, PS2_CLK, PS2_DAT, freq);
display_volume my_vol(vol, HEX0, HEX1);
```

测试：无

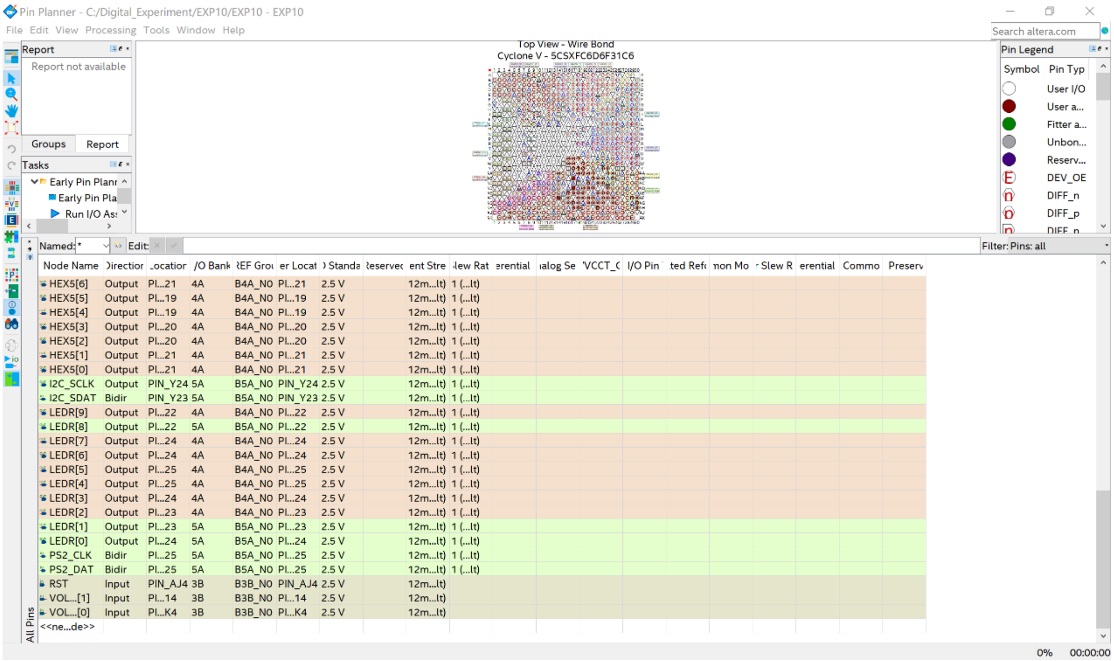
编译：



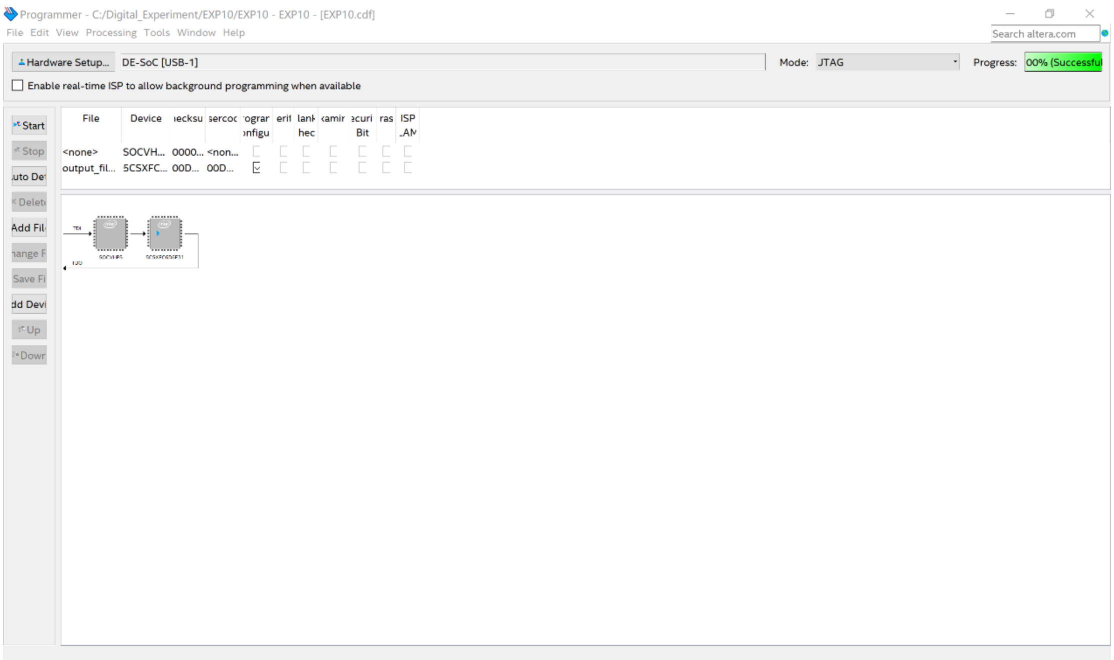
测试：无

编译：

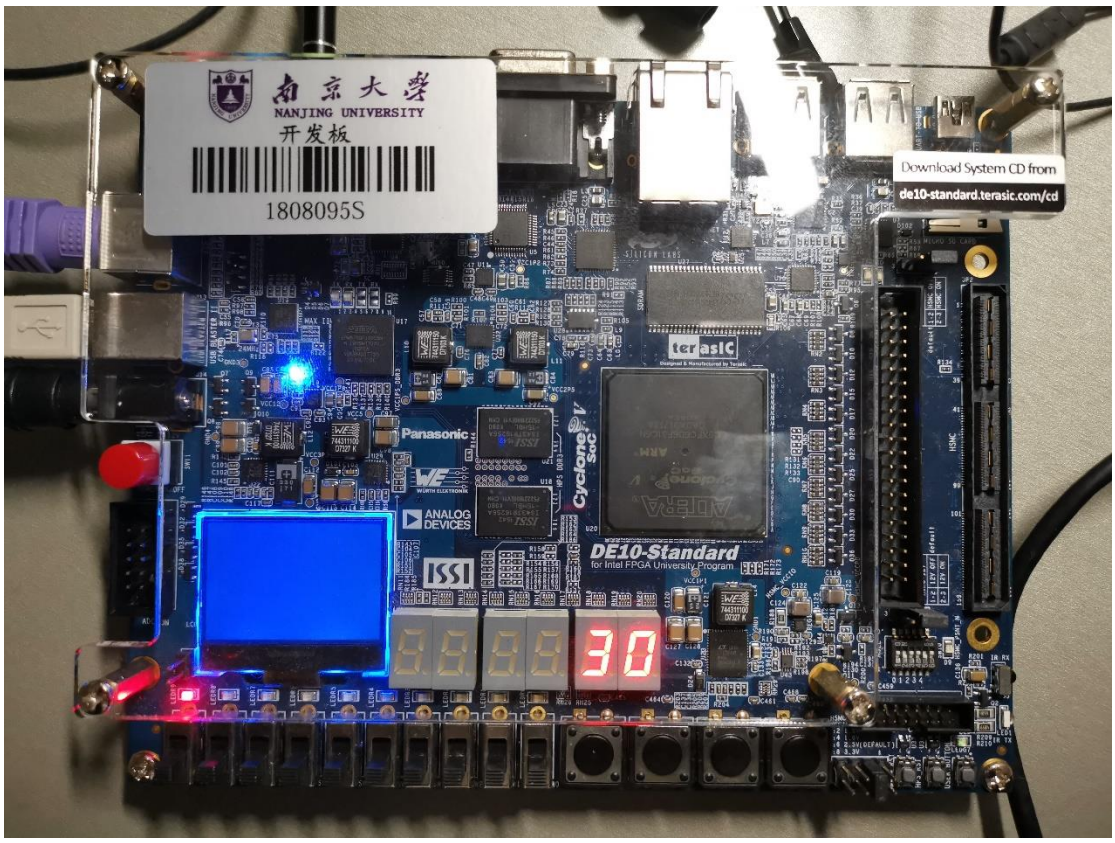
引脚分配：

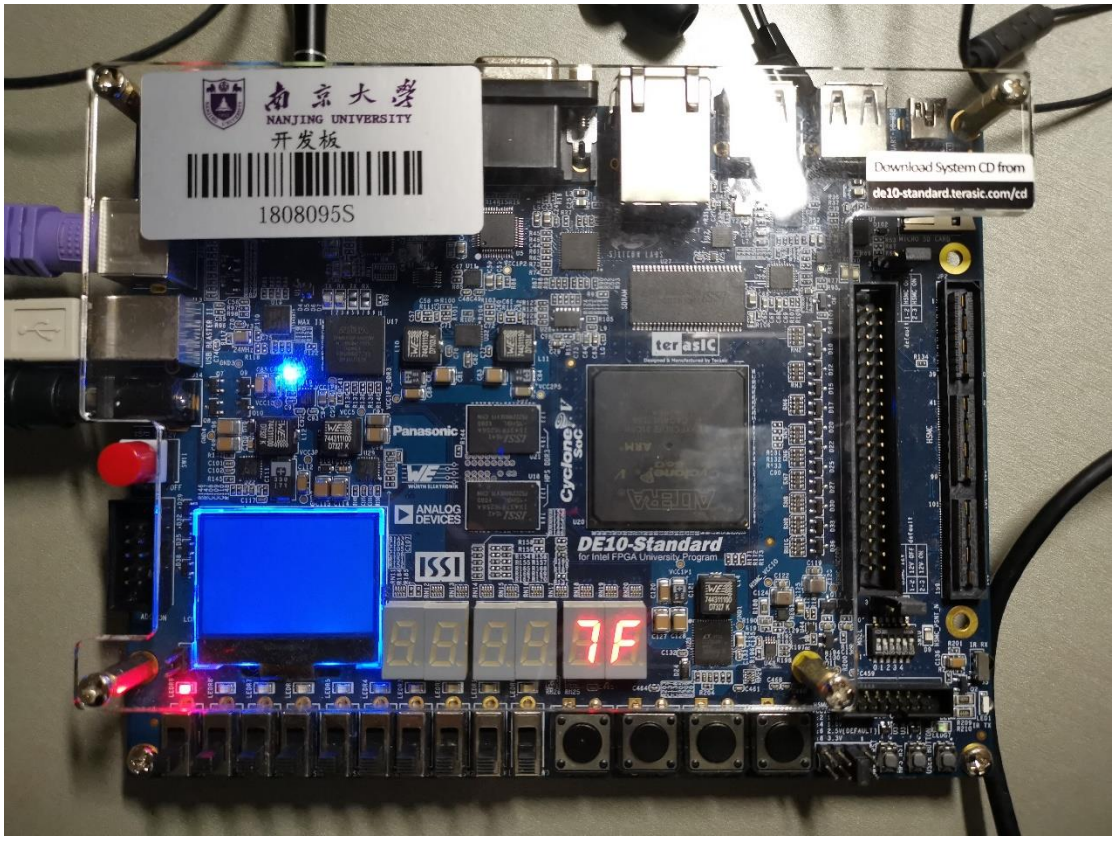


写入：



硬件验证:


南京大学
NANJING UNIVERSITY
开发板
1808095S
Download System CD from
de10-standard.terasic.com/cd
terasIC
Panasonic
ANALOG DEVICES
ISSI
DE10-Standard
for Intel FPGA University Program
30


南京大学
NANJING UNIVERSITY
开发板
1808095S
Download System CD from
de10-standard.terasic.com/cd
terasIC
Panasonic
ANALOG DEVICES
ISSI
DE10-Standard
for Intel FPGA University Program
7F

六、测试方法

该实验无法使用 Test Bench 进行仿真。故采用直接通过耳机反馈的声音音高、音调和音色，来判断可能存在的 bug。

七、实验结果

经过较多次的调试后，实际结果完全符合预期效果。

通过在键盘上按键，耳机能够发出正常的音阶，按键按下后开始发音，按键期间持续发音，松开后停止发音。按无关键不发音。无杂音和爆破音。

此外，按下多个按键时（任意个数），还能够发出正常的和声。

通过 KEY 按钮，能够调高音量和调低音量，且能够重置，并通过数码管显示音量。

八、实验中遇到的问题及解决办法

1、键盘按下后耳机无法发出声音。

解决办法：先检查开发板接口连接处是否有松动，发现连接正常后，更换了不同的耳机，均无法发出声音。因此，可以推断是代码的逻辑错误。代码整体逻辑框架已有，故猜测可能是阻塞与非阻塞的使用不当导致的时序错误，进而无法发出声音。仔细检查后，与猜测原因相符合。

2、耳机发出的声音不够清脆，有杂音或金属音。

解决办法：推测出现杂音可能是部分时刻出现了未定义的状态，出现金属音可能是输出频率的错误。在时序逻辑修改后，声音恢复正常，但无法确定上述两种现象的根本原因。

3、编写好音量控制的功能后，音量大小却无法变化，但数码管显示变化正常。

解决办法：这种情况很难理解，数码管显示的音量正常变化，但实际音量没有变化。经过长时间的排查，发现是 I2C 中音量改变后，未将 cmd_count、mi2c_state 和 mi2c_go 重置为 0，导致尽管 audio_cmd[3]和 audio_cmd[4]中控制音量的部分被修改，但无法在此次周期发出

改变后的声音，且此后的每个周期都如此，故呈现出声音大小无法改变的现象。

4、在编写音量控制功能前，声音正常发出，但在编写音量控制功能后，持续按键时声音会出现间断、渐强渐弱和少量杂音的情况。

解决办法：在编写音量控功能前，声音能够正常发出，说明 bug 出现的地方在于音量改变的部分。声音出现间断，猜测可能是未按照正常时序进行，出现了某些时刻未定义的状态。经过严密的排查，发现无论是否正在改变音量，均会进入改变音量部分的代码，且因该部分将 cmd_count、mi2c_state 和 mi2c_go 全部重置为 0，导致原来的声音无法输出，在下一个周期才能输出，故出现了间断，由于此周期的未定义状态，故有少量杂音，且声音有此前的强转为弱，在下一个周期到来时，声音又由弱变强。最后控制音量改变时才执行该部分代码，声音便恢复正常。

九、实验得到的启示

1、为实现电子琴的功能，不仅需要理解有关音频输出的内容，还要将键盘实验相关的内容理解充分，才能在本实验运用得当，不然极容易出现时序上的错误，导致无法输出声音或声音持续输出无法停止等现象。

2、当讲义太长的时候，可以先快速浏览一遍，然后去看老师的讲解视频，在讲解过程中和讲义对照着看，很快就能理解实验过程的逻辑和相应的实现方式了（老师讲解的真的很清楚）。

3、当需要实现的功能较为复杂时，可以每实现一个一部分时，就进行验证，以保证实验的每个阶段均成功实现，避免在最后一次性验证时，无法找出错误出处的情况。

十、意见和建议

1、对于声音是否清脆的定义很模糊，在做实验时，听见其他同学的声音不尽相同，有些人的声音虽然很清脆，但音色并不相同，应当是实现上仍有问题。此外，对于和声是否符合要求，仍不好判断。建议可以提供一段声音正常输出以及和声的音频，以供实验者参考。

2、实验所用的这种接口的耳机不太好找……建议像键盘和显示器一样可以提供一下，价格也不贵，自己买又有些浪费，用了一次之后就不用了……有这种耳机的同学也不多，就算有也可能在做这个实验，不容易借到。