**《数字电路与数字系统实验》**

**综合实验**

麦克风与音乐播放器

成员：尹浚宇 161130118

刘扬 171850524

时间：2018/12/25

目录

1、实现功能

2、组内分工与工作量

3、整体架构

4、实验原理

5、音频模块

6、播放器模块

7、遇到的问题及解决方案

8、使用手册

1、实现功能

（1）播放四种不同频率的音波

（2）四首乐曲的顺序播放和随机播放、暂停与继续

（3）通过麦克风输入声音，在耳机输出

2、组内分工与工作量

（1）组内分工

刘扬：实验整体框架设计与模块划分，调试程序纠正错误

尹浚宇：各模块的代码实现

（2）工作时长：事前对实验的分析和讨论花费大约3小时

工作总共约20小时，代码实现约12小时，调试程序约8小时

3、整体架构

（1）模块划分：顶层控制模块、音频模块与播放器模块

音频模块 ·音频控制模块

·音频信息配置模块

·音频时钟生成模块

·输入信息并串转换左右声道同步模块

·输出信息并串转换左右声道同步模块

·时钟沿捕获模块

播放器模块 ·播放器控制模块

·音乐时钟生成模块

·乐曲数据生成模块

·正弦波音波生成模块

4、实验原理

1、音频输出原理

在48kHz的采样率下，每个1/48000秒产生一个数字输出样本点，平滑后得到模拟信号。

正弦波与频率的数学表达式为：s(t)=sin(2Πft)=sin(2Πfn/48000)。

使用定点小数表示取样点间隔。16位数字，规定后10位是小数部分。fpga计算乘法比较复杂，使用累加的方式使样本点递增。

经过分析，生成频率是f的正弦波的过程如下：

(1)根据频率f计算递增值d=f\*65536/48000

(2)在系统中维持一个16位无符号整数计算器，每个样本点递增d

(3)根据16位无符号整数的高10位获取查表地址k

(4)使用查表结果作为当前的数字输出

2、音频输入原理

当处于音频时钟的边沿时设置输入有效位为1；之后在音频数据比特时钟的边沿进行一次输入数据的捕获，达到存储数据宽度时将输入有效位置为0不接受输入。

为了对左右声道加以区分，再音频时钟上升沿读取左声道数据，下降沿读取右声道数据，并在下一个时钟边沿将数据写入到输入队列中即可完成音频输入数据的保存。

3、音频接口



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，通过绿色的Line Out接口输出。

FPGA与音频编解码芯片的接口包括两部分：控制接口和音频信号接口。控制接口的功能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。音频信号接口在发送音频时配置音频编解码芯片和通过I2S接口将数字传给音频芯片。

4、I2C接口

两根线分别时数据线SDIN和时钟线SCLK。

通信基本方式为：主节点拉低SDIN数据线，保持SCLK高电平，发出起始信号。之后，主节点拉低SCLK，将第一位数据放在SDIN上，从节点在SCLK的上升沿接受第一个bit的数据。主节点首先发送七位地址来寻找从节点。之后发送一位的R/W位，低电平代表写入寄存器，高电平代表读取，本实验中置为低。之后主节点将SDIN置为高阻态一个周期，此周期内从节点发送总是低电平的ACK信号。随后主节点发送两次8位的数据，每次后面都带有从节点的ACK信号。最终主节点拉低SDIN一个周期后将SDIN和SCLK置为高阻态，完成停止位的发送。

通过I2C接口发送的16位数据，前七位时寄存器地址，后9位是对寄存器设置的值，通过设置这些寄存器可以改变音频通道、音量，调整采样率和音频数据格式等。

5、I2S接口

用于接受和发送数字音频信号

AUD\_XCK：音频信号的基本时钟

AUD\_BCLK：音频数据每个比特的同步时钟

AUD\_DACDAT：输出数字信号数据

AUD\_DACLRCK：输出的左右声道同步

AUD\_ADCDAT，AUD\_ADCLRCK：输出音频信号

使用ip库来产生18.432MHz的时钟

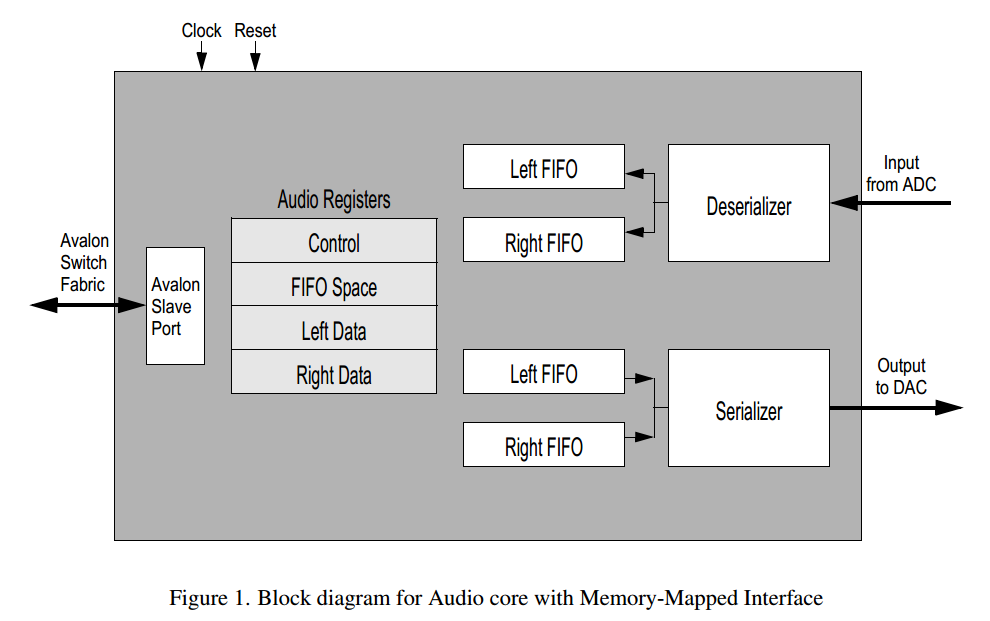
5、音频模块

（1）音频控制模块

在此模块中要控制整个音频工程的输出输出。

需要完成的工作大体包括对于输入信息AUD\_ADCDAT的处理和输出数据AUD\_DACDAT的获取，配置音频信息。

在本模块中，调用时钟沿捕获模块来捕捉音频数据比特时钟、音频输入信号时钟和音频信号输出时钟的边沿；调用音频输入信号并串转换模块来处理并存储输入数据；调用音频输出信号并串转换模块得到输出数据；调用音频时钟生成模块产生音频信号的基准始终；调用音频信息配置模块来配置好音频工程所需要的配置。



（2）音频信息配置模块

本模块是对音频信息进行配置，使用IP核生成

（3）音频时钟生成模块

本模块用于产生音频信号的基准时钟，时钟频率是18.432MHz，使用IP核生成

（4）输入信息并串转换左右声道同步模块

本模块的主要工作是将麦克风数据存储在两个队列中。

数据是32位的，所以调用模块Audio\_Bit\_Counter进行计数。这个模块主要作用是在音频输入时钟的边沿开始进行计数，计数达到数据宽度时，将输出结果置为0，否则是1，这个结果可以作为输入信息的valid信号来使用。

在valid信号有效且在音频数据比特时钟的下降沿时，将现有的一位输入数据放在寄存器data\_in\_shift\_reg中。由于音频数据每个比特同步时钟与音频信号时钟的差距，使用音频信号的时钟作为寄存器写入的信号，可以保证每获取32比特数据正好进行一次写入，寄存器中32比特的数据要分别在音频信号时钟的下降沿、上升沿写入左、右声道数据输入队列中。

（5）输出信息并串转换左右声道同步模块

本模块作用是在左右声道队列中取出数据作为输出。

在音频信号时钟的边沿将输出数据的寄存器清零，之后在音频数据比特同步时钟的下降沿，每次将寄存器内数据左移一位，在读取信号为1时，在相应的队列中每次取出32位数据，通过移位操作，每次只需要选择寄存器最高位的数据作为输出即可。

（6）时钟沿捕获模块

本模块使用50MHz的时钟作为驱动，在上升沿检测目标时钟的状态，从而捕获目标时钟的上升沿和下降沿。

实现方法是：使用两个变量保存目标时钟的当前状态和上一状态，这两个状态都是在50MHz时钟的上升沿进行更新，两个状态不同时，说明目标时钟越过了一个边沿，根据当前时钟状态和上一时钟状态即可判断是上升沿还是下降沿。

6、播放器模块

（1）播放器控制模块

在本模块中调用以下的三个模块，使用50MHz的时钟作为输入产生想要的时钟来驱动对于数据的采集，通过对于Song模块的调用得到freq来作为间隔在正弦波中取数据，得到的数据就是最终可以传给输出队列的信息。

（2）音乐时钟生成模块

通过提供参数，依赖已有的50MHz的时钟来生成想要的时钟。在此模块设置参数clk\_freq来修改另一个参数countlimit = 50000000/2/clk\_freq，在计数达到countlimit时反转一次输出时钟，以达到产生目标时钟的效果

（3）乐曲数据生成模块

通过读取事先存储在四个mif中的数据来初始化四个存储器，另外使用一个mif初始化一个随机数存储器。

通过mode来选择顺序播放还是随机播放。顺序播放时，变量song\_number + 1即可得到下一首乐曲的编号；随机播放时，依次在随机数存储器中取出一个随机数来赋值给song\_number来确定下一首乐曲编号。

state表示乐曲的播放和暂停，播放时通过计数依次在存储器中取数据，暂停时不在乐曲存储器中取数据，取出的数据作为freq返回。

（4）正弦波音波生成模块

本模块使用sintable.mif来初始化一个存储器来存储正弦的音波。

乐曲数据生成模块的输出freq作为本模块的输入，在本模块中，freq\_counter是一个16位的数据，低六位代表小数部分，通过将其每次加freq进行改变，以其高10位作为地址进行索引，在存储器中取出数据，这个数据最终返回给顶层模块将其赋值给左右声道的输出数据，达到乐曲播放的效果。

7、遇到的问题及解决方案

（1）麦克风输入时，耳机的输出有杂音

**问题**：一开始使用的是软件IP核生成的音频工程，工程中包括的Auiod\_ctrl模块的参数中包括有音频输入的valid、ready信号（from\_adc\_left\_channel\_valid、from\_adc\_left\_channel\_ready）和音频输出的vaild、ready信号（to\_dac\_left\_channel\_valid、to\_dac\_left\_channel\_ready）。对信号的设置为：from\_adc\_left\_channel\_valid 和to\_dac\_left\_channel\_ready是输出信号不需要自己手动设置；另外两个输入信号设置为1。这样设置之后在硬件上运行时，耳机会有很强烈的噪音。

**解决方案**：修改对于信号的设置为： from\_adc\_left\_channel\_ready为真时表示可以读取数据，将其设置为采样的时钟（一开始为50MHz）；to\_dac\_left\_channel\_valid是输出数据有效信号，将其设置为from\_adc\_left\_channel\_valid 和to\_dac\_left\_channel\_ready两个输出信号的与，结果发现依然是有噪音。后来修改采样时钟的频率，发现噪音会有变化，但是又发现同一个采样时钟频率下，两次硬件实现上的噪音声音并不同，说明修改时钟频率并非正确的解决方式。

最终选择是重写整个音频工程。在重写中, 没有使用IP核, 而是借鉴并简化了IP核的音频控制模块, 并借鉴了实验10提供的音频芯片配置模块, 两个模块配合使用完成了整个音频工程。使用重写后的音频工程测试时, 噪音大大减少, 达到了预期效果。

（2）开始尝试实现变声器时，在添加存储信息的输入队列后，耳机听不见声音。

原因：音频信息采样需要48KHz的时钟，这种时钟无法由IP核生成也不能由50MHz的时钟准确得到，所以在判断向队列中写入数据时，敏感列表中使用的不是时钟边沿而是输入数据的vaild位。

解决方案：改用50MHz时钟沿驱动，在always语句块中检查valid进行写入，读取是在每写入256个数据以后便进行一次读取，读取512个数据准备送入海宁窗送。

继而产生的问题是：在输入队列中取出来的数据送入耳机中，听见的是随着麦克风输入而变化的电流撕扯的声音而不是人声。

（3）加入使用IP核生成的FFT之后完全没有声音

此模块调试失败，之后转向实现音乐播放器。QAQ

8、使用手册

（1）KEY0是整个音频工程的重置按钮，按下重置

（2）KEY1是音乐暂停与继续的切换

（3）KEY3是切换音乐，SW5用于标志顺序播放还是随机播放，SW5为1的时候，按下KEY3切换为下一首，SW5为0的时候，按下KEY3是随机切换。

（4）SW0 –SW3可以切换播放四种不同频率的音波

（5）SW4用于开启音乐播放器，当所有的6个开关全部为0时才可以使用麦克风。