实验 19 音乐播放实验实验报告

3240102873 胡东东

一、实验目的

（1）掌握音符产生的方法，了解 DDS 技术的应用；

（2）了解音频编解码的应用；

（3）掌握系统自上而下的数字系统设计方法。

二、实验任务与要求

（1）设计并仿真一个 DDS 正弦信号发生器，要求：

①采样频率fc = 48kHz；

②正弦信号频率范围为 20Hz ~ 20kHz；

③正弦信号序列宽度 16 位，包括一位符号；

（2）设计一个音乐播放器，要求：

①可以播放四首乐曲，设置 play/pause_button、next_button、reset 三个按键。按 play/pause_button 键，音乐在播放和暂停间切换；按 next_button 播放下一首乐曲；

②LED0 指示播放情况（播放时点亮）、LED2 和 LED3 指示当前乐曲序号。

三、实验原理与设计思路

（1）DDS（直接数字频率合成）的基本原理

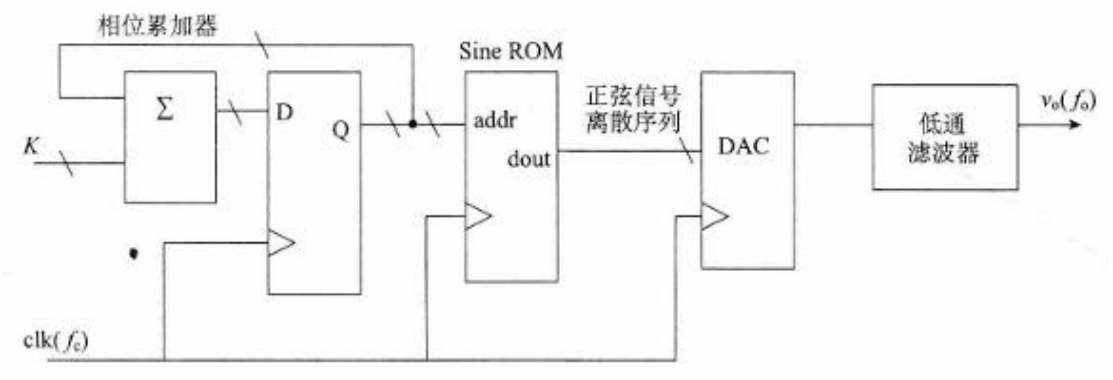
直接数字频率合成（Direct Digital Synthesis, DDS）是一种通过数字方式生成模拟信号的技术，其核心思想是在数字域构建信号波形，再通过数模转换与滤波输出模拟信号。

在数字域生成正弦信号的基本思路是：用 ROM 或 RAM 存储一张正弦波采样表，再按特定规律读取表中的数据，经数模转换器（DAC）转换为模拟波形，最终通过低通滤波器平滑输出。为了实时改变输出信号的频率，有两种常见实现方式：

①改变查表寻址的时钟频率，从而改变波形数据的读取速率；

②改变查表的地址步长（相位增量），即 DDS 采用的方式，输出频率与相位增量呈线性关系，因此频率控制更加灵活、精准。

DDS 的基本结构由相位累加器、正弦查询表（Sine ROM）、数模转换器（DAC）和低通滤波器四部分组成。



相位累加器：在每个时钟周期内，将频率控制字（相位增量）与当前相位值累加，得到新的相位地址；

正弦查询表（Sine ROM）：存储了一个完整周期的正弦波量化采样值，其地址线位为m，数据线宽度为n，采样值按公式$S(i) = (2^{n-1} - 1) \times \sin\left(\frac{2\pi i}{2^m}\right), i = 0,1,2,\dots,2^{m-1}$生成，数据以补码形式存储；

数模转换器（DAC）：将查询表输出的数字采样值转换为阶梯状的模拟信号；

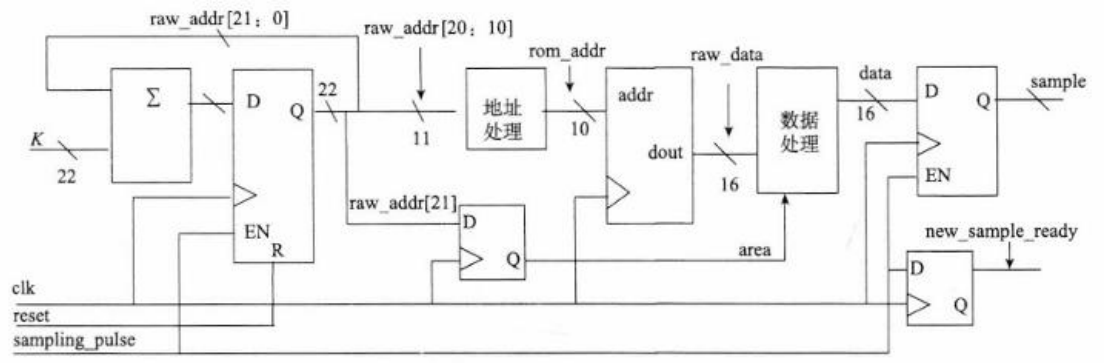
低通滤波器：滤除 DAC 输出中的高频分量与量化噪声，得到平滑的正弦波信号。

DDS 的输出信号频率f_0由系统时钟频率f_c、相位累加器位数m和频率控制字K共同决定，满足关系：$f_0 = \frac{K \times f_c}{2^m}$，可以看出，输出频率与频率控制字$K$呈线性关系，因此可以通过改变$K$实现频率的连续、快速调整。

为了提高频率分辨率，相位累加器通常会扩展为“整数+小数”的混合结构（m位整数部分和p位小数部分），仅取高m位整数作为 ROM 的地址。根据奈奎斯特采样定律，DDS 的理论最高输出频率为时钟频率的一半（$f_0 \le f_c/2$）；实际应用中为了抑制杂散噪声，一般限制最高输出频率不超过时钟频率的 40%（$f_0 \le 40\% \times f_c$），对应的频率控制字最大值约为$40\% \times 2^{m-1}$。

（2）DDS 总体设计

DDS 总体电路框图如下所示：



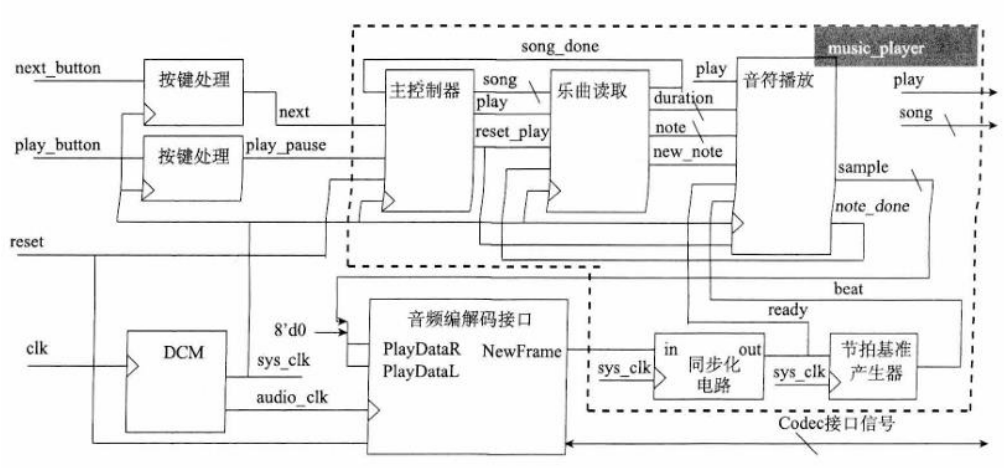
相位累加得到 22 位原始地址 raw_addr[21:0]，整数部分 raw_addr[21:10]即为完整周期正弦信号样品的地址，其中高两位地址 raw_addr[21:20]可区分正弦的四个区域。由于 sine_rom 只保存了四分之一周期的 1024 个样品，所以 raw_addr [21:10]不能直接作为 sine_rom 地址，必须进行必要处理，处理方法如下表所示：

sine_rom 的地址和数据处理方法

正弦区域	Sine ROM 地址	正弦样品 data	备注
0	raw_addr[19:10]	raw_data[15:0]	
1	当 raw_addr[20:10]=1024 时，rom_addr 取 1023，其他情况取~raw_addr[19:10]+1	raw_data[15:0]	地址镜像翻转
2	raw_addr[19:10]	~raw_data[15:0]+1	数据取反
3	当 raw_addr[20:10]=1024 时，rom_addr 取 1023，其他情况取~raw_addr[19:10]+1	~raw_data[15:0]+1	地址镜像翻转 数据取反

（3）音乐播放器的基本原理

根据实验任务可将系统划分为时钟管理模块(DCM)、按键处理、主控制器、乐曲读取、音符播放(note_player)、同步化电路、节拍基准产生器和音频编解码接口电路等子模块，如下图所示：



各主要子模块作用如下：

时钟管理模块(DCM)：产生 100MHz 的系统时钟 sys_clk 和 12.5MHz 的音频时钟 audio_clk。

主控制器模块(mcu)：接收按键信息，通知 song_reader 模块是否要播放(play)及播放哪首乐曲(song)。

乐曲读取模块(song_reader)：根据 mcu 模块的要求，逐个取出音符信息{note, duration}送给 note_player 模块播放，当一首乐曲播放完毕，回复 mcu 模块乐曲播放结束信号(song_done)。

音符播放模块(note_player)：接收到需播放的音符，在音符的持续时间内，以 48kHz 速率送出该音符的正弦波样品给音频编解码接口模块。当一个音符播放结束，向 song_reader 模块发送一个 note_done 脉冲索取新的音符。

音频编解码接口模块：负责将音符的正弦波样品转换为串行输出并发送给音频编解码芯片 ADAU1761。音频编解码芯片 ADAU1761 接收正弦波样品，再进行 AD 转换并放大，最后送至扬声器播放。注意，note_player 模块产生的正弦波样品为 16 位二进制，需在低位加 8 个 0 后送入音频编解码接口模块。

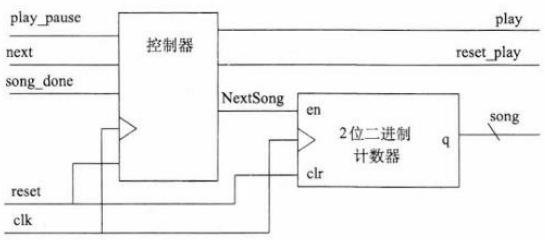
节拍基准产生器：产生 48Hz 的节拍定时基准脉冲信号(beat)，而 ready 信号频率为 48kHz，因此，节拍基准产生器即为分频比为 1000 的分频器。

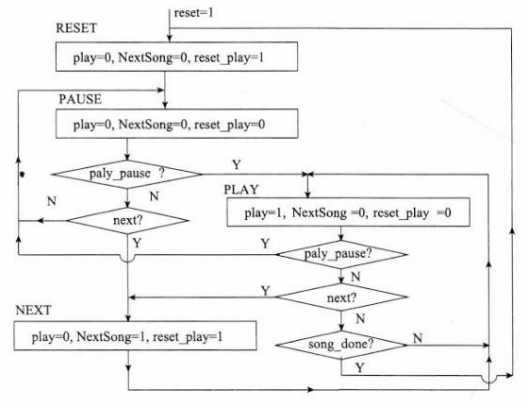
按键处理模块：完成输入同步化、防颤动和脉宽变换等功能。

（4）音乐播放器各子模块设计：

①主控制模块 mcu 的设计

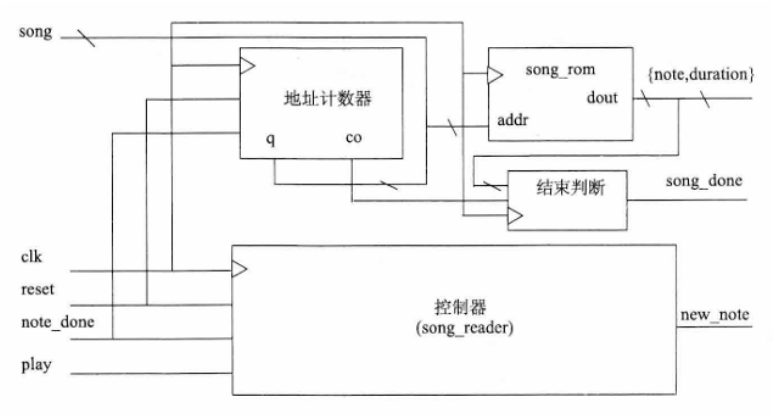
主控制模块 mcu 有响应按键信息、控制系统播放两大任务。其原理框图和算法流程图如下图所示。

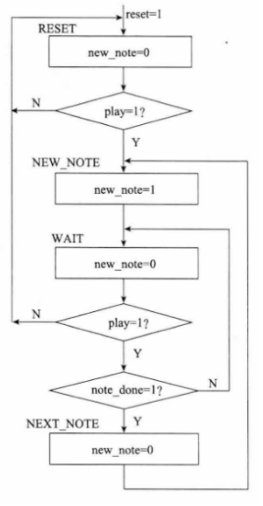




②乐曲读取模块 song_reader 的设计

song_reader 模块结构框图与控制器算法流程图如下所示





song_rom 是一个只读存储器，用来存放乐曲，容量为 27×12bits，共存放四首乐曲，每首乐曲占用 27×12bits 空间，即每首乐曲最长由 32 个音符组成。因此，song_rom 高 2 位地址决定哪首乐曲，而低 5 位地址决定这首乐曲的哪个音符。song_rom 每个地址存放一个音符信息，音符信息由 12 位二进制组成，高 6 位表示音 符标记 note，低 6 位表示音长 duration。

地址计数器为 5 位二进制计数器，其中 note_down 为计数器使能输入，当 note_down 为高电平时允许计数。当地址计数器出现进位或 duration 为 0 时，表示乐曲结束，应输出一个时钟周期宽度的高电平脉冲信号；结束判断模块采用状态机实现脉宽变换电路。

③音符播放模块 note_player 的设计

音乐播放模块 music_player 是本实验的核心模块，它主要任务有：

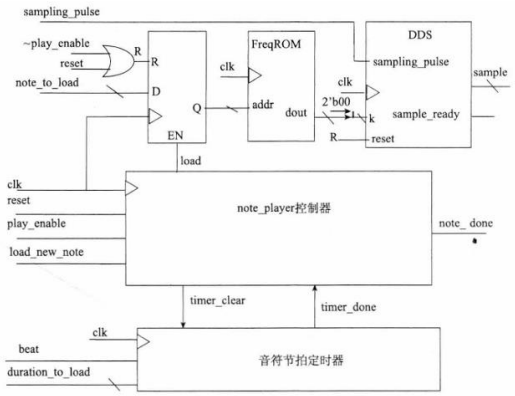
1、从 song_reader 模块接收需播放的音符{note, duration}；

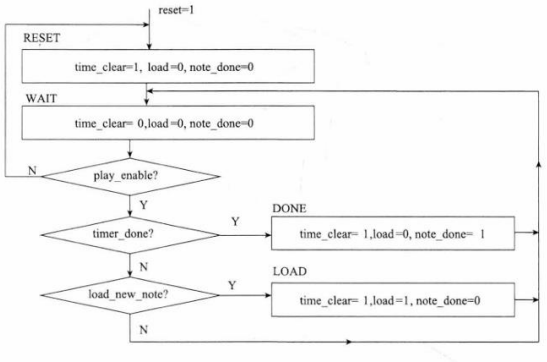
2、根据 note 值找出 DDS 的相位增量 k；

3、以 48kHz 速率从 Sine ROM 取出正弦样品送给音频编解码器接口模块；

4、当一个音符播放完毕，向 song_reader 模块索取新的音符。

FreqROM 为只读存储器，完成音符标记 note 与 DDS 模块的相位增量 k 查找表关系。note_player 控制器负责与 song_reader 模块接口，读取音符信息，并根据音符信息从 Frequency ROM 中读取相应相位增量 k 送给 DDS 子模块。另外，note_player 控制器还需要控制音符播放时间。note_player 结构框图和算法流程如下图所示：



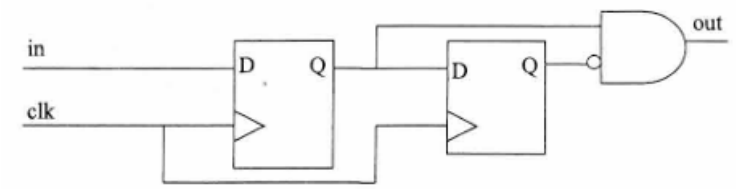


音符定时器为 6 位二进制计数器，beat、timer_clear 分别为使能、清 0 信号，均为高电平有效;定时时间由音长信号 duration_to_load 决定,即 duration_to_load-1 个 beat 周期，timer_done 为定时结束标志。

子模块 DDS 的功能就是利用 DDS 技术产生正弦样品，需要注意的是，DDS 模块的输入 k 为 22 进制,因此需要 FreqROM 输出的 20 位相位增量高位加 2 个 0 后接入 DDS。

④同步化电路的设计

音频编解码接口模块的输出信号 NewFrame 的脉冲宽度为一个 audio_clk 时钟周期，需通过同步化处理，产生与 sys_clk 同步且脉冲宽度为一个 sys_clk 时钟周期的信号 ready。电路由同步器和脉冲宽度变换电路组成:



⑤时钟管理模块的设计

IP 内核时钟管理模块的输入时钟 clk 频率为 100MHz，产生 100MHz 的系统时钟和 12.5MHz 的音频时钟。

⑥节拍基准产生器模块的设计

节拍基准产生器产生 48kHz 的节拍定时基准脉冲信号(beat)，而 ready 信号频率为 48kHz，因此节拍基准产生器即为一个分频比为 1000 的分频器。

四、主要仪器设备

（1）装有 Vivado 和 ModelSim SE 软件的计算机；

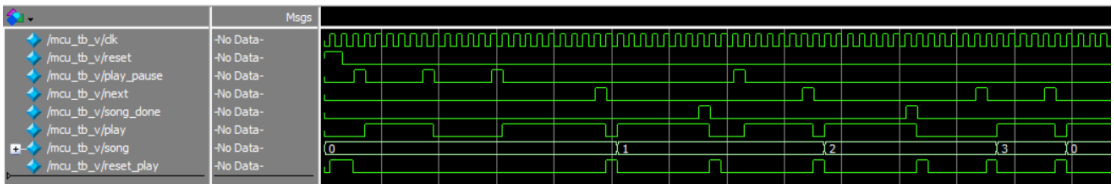
（2）Nexys Video Artix-7 FPGA 多媒体音视频智能互联开发系；

（3）有源音箱或耳机。

五、实验内容与仿真分析

（1）编写 mcu 模块的 Verilog HDL 代码，并用 ModelSim 仿真验证；

mcu 模块由顶层模块 mcu.v 和子模块控制器 mcu_controller.v、计数器 counter_n.v 组成。ModelSim 仿真波形部分截图如下所示：

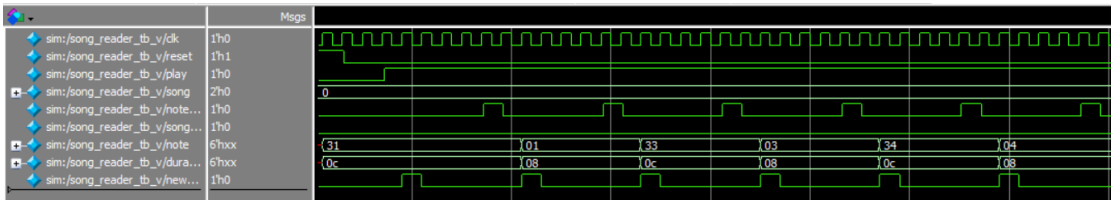


根据 mcu 控制器的算法流程图，分析上图波形：

从零时刻开始观察，时钟上升沿到达之前 reset = 1，故当时钟上升沿到达时，控制器处于 RESET 状态，此时输出 play = 0，nextsong = 0，reset_play = 1，符合控制要求；接着在下一个时钟上升沿到来前 reset 置为 0，故当时钟上升沿到来时，状态机进入 PAUSE 状态，并输出输出 play = 0，nextsong = 0，reset_play = 0，符合控制要求；再接着，在下一个时钟上升沿到达前，reset = 0，play_pause = 1，表示用户按下 play 键，故当时钟上升沿到达时，进入 PLAY 状态，此时输出 play = 1，nextsong = 0， reset_play = 0，满足播放乐曲的设计要求；观察 play_pause 曲线的下一个脉冲，时钟上升沿到来前 reset = 0，play_pause = 1，play = 1，表示按下 pause 键，故当时钟上升沿到来时，state = PAUSE，此时输出 play = 0，nextsong = 0，reset_play = 0，满足暂停乐曲的需求。后续波形的分析与这部分相仿，状态机根据 play/pause 信号以及乐曲的长度，循环播放存储的四首乐曲。由此可知该模块设计符合要求。

（2）编写 song_reader 模块的 Verilog HDL 代码，并用 ModelSim 仿真验证；

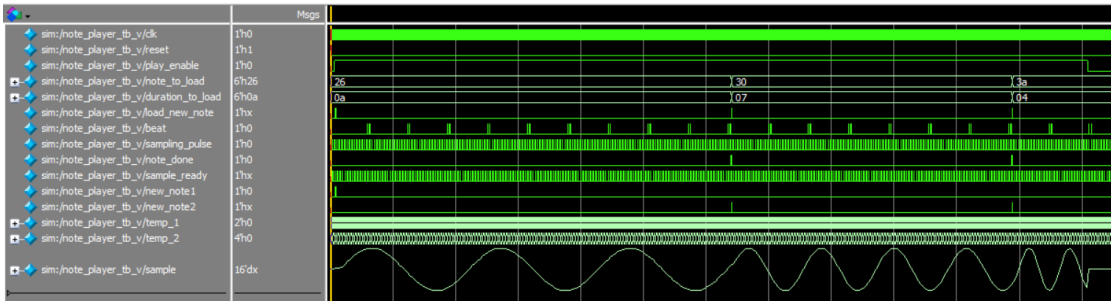
song-reader 模块由顶层模块 song_reader.v 和子模块控制器 song_reader_controller.v、地址计数器模块 counter_n.v、歌曲存储模块 song_rom.v 和结束判断模块 end_detector.v 组成，结束判断模块采用状态机方法实现。ModelSim 仿真波形部分截图如下所示：

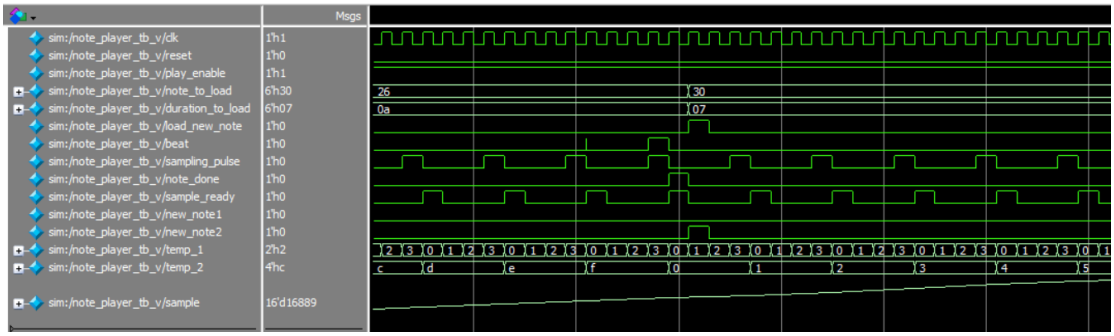


由仿真波形图可见，当 reset = 1 时，状态机进入 RESET 状态；当 reset 置为 0 并且 play = 1 时，状态机进入 NEW_NOTE 状态并输出 new_note = 1，然后进入 WAIT 状态，输出 new_note = 0，进入判断循环；当 note_done = 1，下一个时钟上升沿到来时，进入 NEXT_NOTE 状态，输出 new_note = 0，读取下一个音符的地址；等到下一个时钟上升沿到来时，进入 NEW_NOTE 状态，输出 new_note = 1，进行下一个音符的播放，由此可以实现乐符读取与播放的控制，符合设计要求。

（3）编写 note_player 模块的 Verilog HDL 代码，并用 ModelSim 仿真验证；

note_player 模块由顶层模块 note_player.v 和子模块控制器 note_player_controller.v、音符节拍定时器 beat_timer.v、只读存储器 frequency_rom.v 以及之前编写的 DDS 模块组成。ModelSim 仿真波形部分截图如下所示：

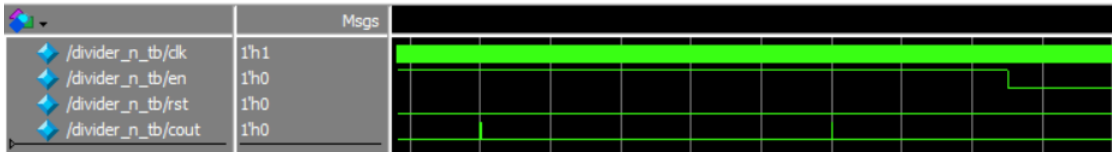




当 note_to_load 增大时，输出正弦信号频率增大，正弦信号频率与 note_to_load 的值呈正相关，且当 play_enable=0 时，sample=0，符合设计要求。当 note_down 信号变为高电平，load_new_note 信号产生，用于更新播放的音符。波形符合预期，note_player 子模块符合设计要求。

（4）编写 1000 分频器模块的 Verilog HDL 代码及其测试代码并用 ModelSim 仿真；

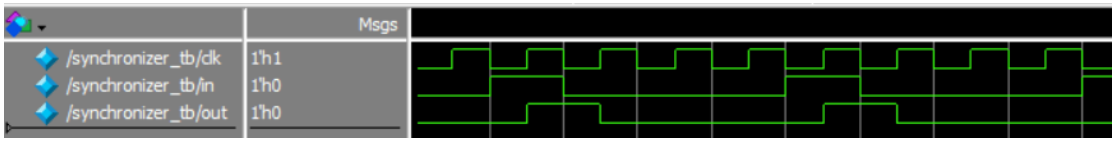
ModelSim 仿真波形部分截图如下所示（但 1000 分频的波形效果不易展示）：



仿真结果显示，当复位信号 rst 有效后，计数器归零。随后使能信号 en 置为高电平，分频器开始工作。在 clk 时钟驱动下，每经过 1000 个时钟周期，输出信号 cout 产生一个宽度为一个时钟周期的高脉冲，表明计数达到 999（即 n-1）且 en 有效。当 en 被拉低时，计数器暂停计数，cout 保持低电平，无脉冲输出；en 重新置高后，分频器恢复计数并从当前值继续，后续仍能准确产生 1000 分频的脉冲波形。整个过程中，cout 输出脉冲周期与 clk 周期之比严格符合 1000:1，且脉冲宽度符合一个时钟周期，证明分频器逻辑正确、功能满足设计要求。

（5）编写同步化电路的 Verilog HDL 代码及其测试代码，并用 ModelSim 仿真验证；

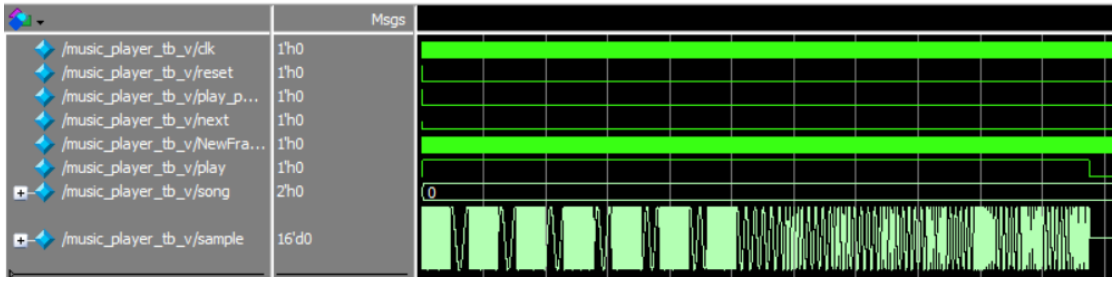
ModelSim 仿真波形部分截图如下所示：



由仿真波形可见，输出信号 out 跟随着输入时钟 in 变化，其脉冲宽度为一个时钟周期，设计符合要求。

（6）编写次顶层 music_player 模块的 Verilog HDL 代码及其测试代码，并用 ModelSim 仿真验证；

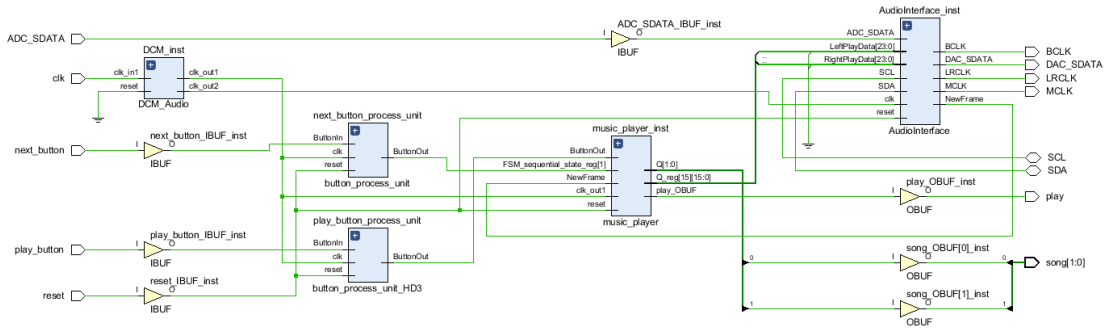
music_player 次顶层模块由子模块主控制器 mcu.v、乐曲读取 song_reader.v、音符播放 note_player.v、同步化电路 synchronizer.v 和节拍基准产生器 divider_n.v 组成，ModelSim 仿真波形部分截图如下所示：



由仿真波形可见，当 play 信号为 1 时，sample 产生随乐谱播放情况变化的正弦信号，且仿真结果与参考书基本吻合，符合设计要求。

六、建立 Vivado 工程

建立 Music_player 的 Vivado 工程，添加 music_player 模块及其子模块、音频编解码接口模块的网表文件和端口文件、按键处理模块的网表文件和端口文件以及约束文件，同时生成符合要求的 DCM 内核，对工程进行综合，实验综合得到的电路原理图如下图所示：



下载工程文件到开发板后，将耳机接入实验开发板音频输出插座，操作相关按键：

①按下 reset（中间按键）实现复位；

②按下 play/pause（右边按键）实现暂停/播放；

③按下 next（下方按键）实现切换歌曲。

LED0 指示灯指示乐曲播放与暂停，LED3 指示灯与 LED2 指示灯用于指示乐曲序号。

验收过程中播放音乐正常，功能正常实现。

七、思考题

（1）在实验中，为什么 next_button、play_pause_button 两个按键需要消颤动及同步化处理，而 reset 按键不需要消颤动及同步化处理？

机械按键按下与松开时会产生电平抖动，FPGA 高速时钟会把抖动误识别为多次按键信号。next_button 和 play_pause_button 属于功能控制按键，要求按下一次只执行一次操作，若不做消抖处理，电平抖动会引发多次误触发，出现反复切换播放状态、连续切歌等问题。同时这两个按键是外部异步输入，和 FPGA 内部系统时钟不同步，容易产生亚稳态，造成逻辑错误，因此必须进行消抖和同步化处理。本实验中 reset 为全局异步复位信号，直接连接各模块的异步复位端，只要检测到有效电平就能完成系统复位，短暂的电平抖动不会影响复位功能，且异步复位信号无需同步到系统时钟域，因此 reset 按键不需要做消抖和同步处理。

（2）在主控制器(mcu)设计中，是否存在接收不到按键信息？若存在，概率多大？有没有必要修改设计？

该设计存在接收不到按键信息的情况。本实验中按键处理模块输出的有效信号为单个系统时钟周期的窄脉冲，主控制器依靠时钟沿采样信号，若脉冲落在两个时钟沿之间就会出现漏采样；若异步信号同步不彻底，产生亚稳态，也会导致按键信息丢失。系统主时钟为 100MHz，时钟周期仅 10ns，在理想时序且无亚稳态的情况下，按键漏检的理论概率较低；但如果同步电路不完善，亚稳态会大幅提升漏检概率，长期运行时按键丢失问题会频繁发生。综合实际使用场景，原有设计有必要进行优化。可以将按键脉冲展宽为 2 至 3 个时钟周期，保证控制器稳定采样；也可以在信号输入端增加两级同步寄存器，消除亚稳态；还能增加寄存器锁存按键事件，等待控制器响应后再清零，避免按键信号丢失。

八、总结与体会

本次音乐播放实验内容涵盖 DDS 波形生成、状态机控制以及多模块协同工作等多个方面，整体工程规模较大、难度较高。实验之初，面对厚厚的任务书和复杂的流程图，我内心确实产生过畏难情绪，总想着先把教材原理完全吃透再开始动手操作。然而在真正进入设计阶段后，我逐渐意识到，许多知识点单靠阅读很难真正理解，反而是在实际编写代码、仿真调试的过程中，原先抽象的原理才一点一点变得清晰起来。边做边学、在实践中验证理论，成为我完成这次实验最重要的学习方法。

在整个实验推进的过程中，我遇到了不少困难，但大多并非无法克服。初期，代码编译或仿真出现报错是家常便饭，好在和 ModelSim 提供的错误提示往往能指引排查方向。按照提示一步步定位问题、修正程序，最终各模块的波形都顺利达到了设计预期。这一阶段的反复磨练，让我对 Verilog 语法细节和排错流程有了更熟练的掌握。

我遇到的另一个问题是生成符合要求的 DCM 时钟管理模块。做实验室已经遗忘了之前做过的 DCM 设置，后来我重新翻看教材中关于 IP 内核使用与仿真的章节，对照其中的步骤仔细检查配置界面，才逐渐理解了输入时钟频率、输出频率与分频/倍频系数的关系，最终成功生成了正确的 DCM 模块。这次经历让我明白，遇到不熟悉的工具或流程时，随时回到书本中寻找依据，往往比无目的尝试更高效。

另外，在乐曲扩展环节，我尝试往 song_rom 中添加一首自选的第四首乐曲，但播放出的旋律并不理想。反思原因，主要是自己对音符频率对照表中的音符记号与日常简

谱音高之间的对应关系还没有完全厘清，导致编曲时部分音准出现偏差。这提醒我，硬件设计不仅需要工程实现能力，也需要对应用领域知识有足够了解，否则功能正确也不能保证效果如意。

回顾整个实验流程，从最初的 DDS 子模块设计，到主控制器、乐曲读取、音符播放等一个个子系统的完成，再到顶层模块的例化与板级验证，我切身体会到了“自顶向下”数字系统设计方法的优势。模块化的结构让代码变得层次分明，既便于单独调试，也有利于在顶层中灵活集成。在这过程中，我对有限状态机的编写、分频定时器的应用等也有了更深的理解，不再仅仅停留于书本上的概念。

除此之外，这次实验也让我认识到，数字电路设计对细致和耐心有着很高的要求。一个小的信号连接错误、一个状态转移条件的疏忽，都可能导致整个系统无法正常工作。仿真波形中看似微小的一处异常，常常需要反复追溯好几层信号才能锁定根源。正是这样的过程，锻炼了我分析问题和推理判断的能力，也让我对硬件调试有了更真切的体会。

最终，当耳机中传出按动按键后正确切换的乐曲声时，先前的疲惫感一扫而空，取而代之的是满满的成就感。通过此次实验，我不仅学会了一整套音频播放系统的设计方法，也积累了 FPGA 开发与调试的实践经验，更重要的是克服了刚上手时的畏难心理，领悟到“纸上得来终觉浅，绝知此事要躬行”的道理。未来在更复杂的数字系统设计中，我将继续秉持这样的学习态度，勇于实践，勤于总结，不断提升自己的工程能力。