进击的小鸟

组员：
黄予 2013011363 音频处理模快
杨晓成 2013011383 游戏逻辑模块
姚炫容 2013011379 显示、存储模块

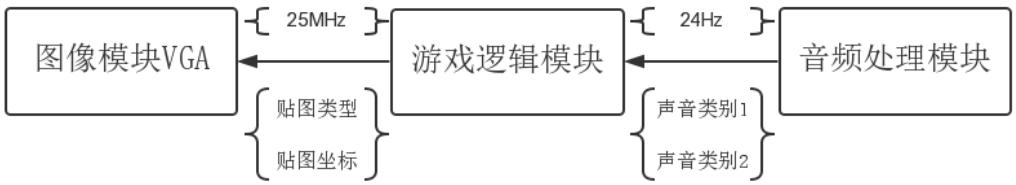
【故事背景】

当 Flappy Bird 来到 Angry Bird 的世界，它必须要为了自己的生存进行必要的斗争。

我们的游戏的亮点在于采用了声音控制。我们购置了 wm8731 声卡芯片，通过发出不同强度和频率的声音，来控制我们的主角 Flappy Bird 进行上升和发射子弹等操作。

初始状态下，屏幕各处会随机刷出一些较弱的怪物，当杀死这些怪物的数量达到一定程度时，将会召唤出一只强大的 Angry Bird 与你进行战斗。若你不幸阵亡，Angry Bird 的团体成员会同时出现在屏幕上表达对你的鄙视。此时若你仍不死心，对着话筒喊出足够大的声音即可重新开始游戏。

【系统结构】



我们的系统采用统一的基准时钟，其余时钟在基准时钟的基础上生成。

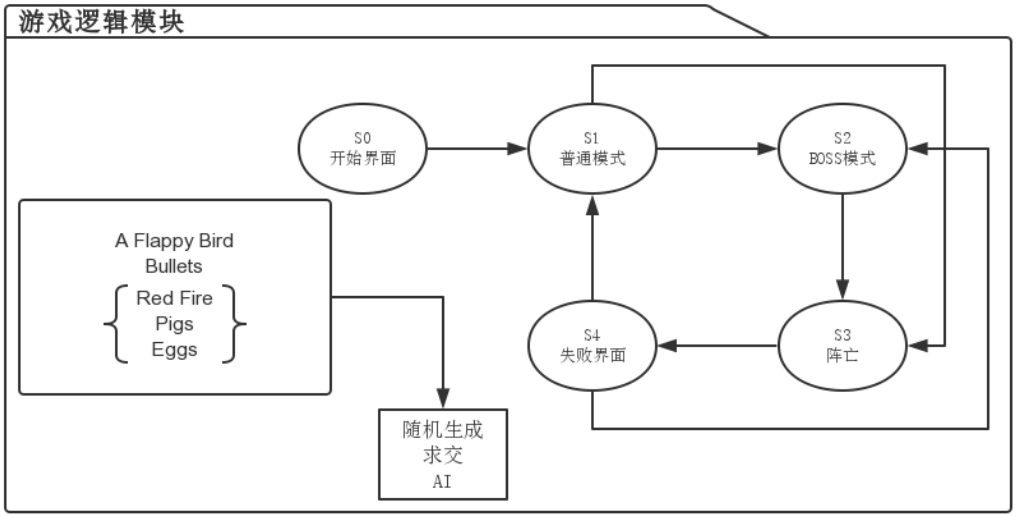
游戏逻辑模块与音频处理模块的信息交流使用 24Hz 的时钟。在下降沿游戏逻辑模块根据音频处理模块传输的声音类型决定主角的行为。

游戏逻辑模块在每个 25MHz 的下降沿处理一次逻辑信息。

游戏逻辑模块与图像模块的信息交流使用 25MHz 的时钟。图像模块在每个上升沿向游戏逻辑模块发送像素的位置信息，在每个下降沿接收由逻辑模块返回的信息。我们将游戏中所有的图片储存在了 SRAM 中，逻辑模块返回该像素与图片哪一个像素相对应。

可以看出，每次图像模块在下降沿接收的信息实际上对应上上个上升沿向游戏逻辑模块发送的信息。在硬件设计中硬件间的通讯会有一定延迟，这样设置可以保证收到所需的信息。

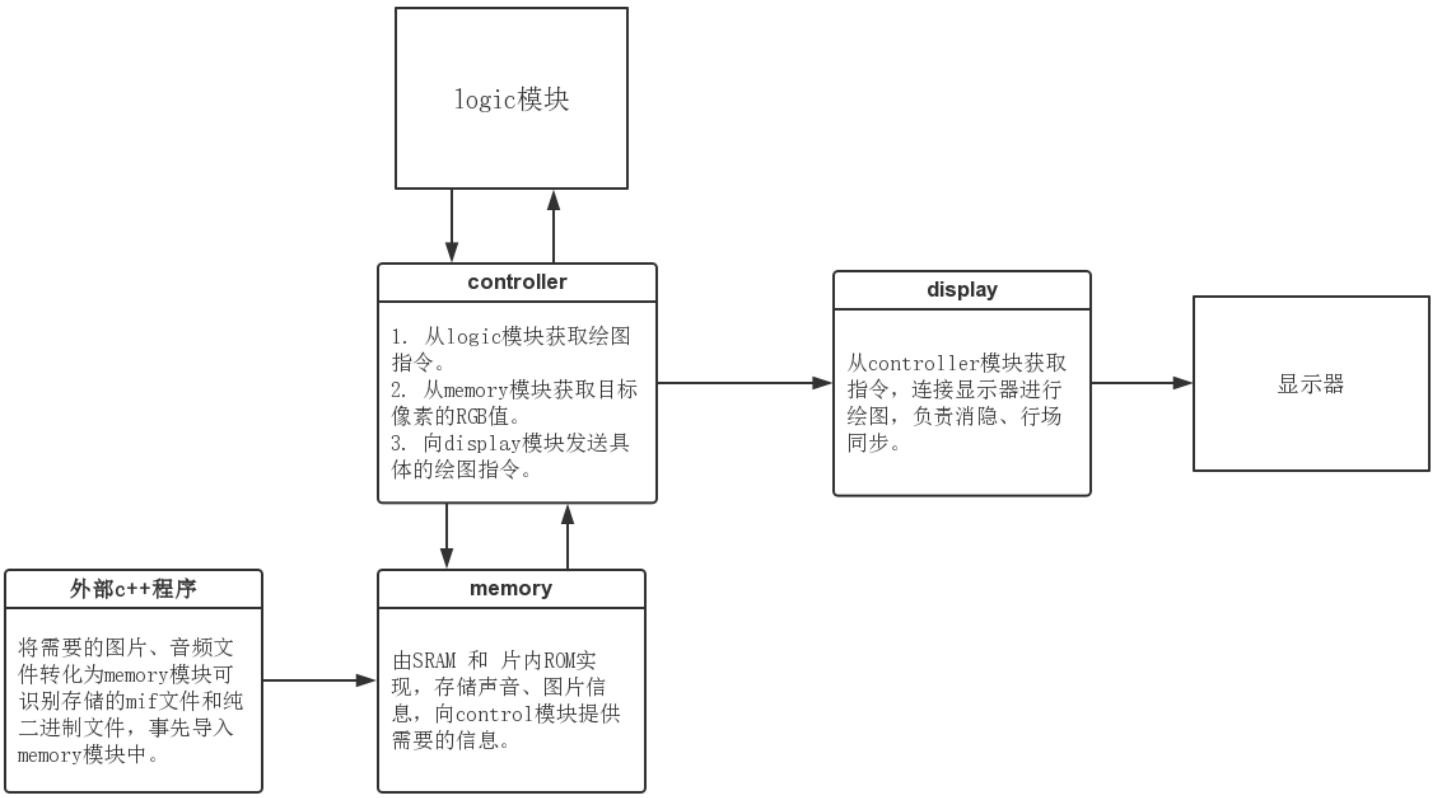
【游戏逻辑模块】



游戏逻辑模块的主要内容是各状态间的转换，以及游戏中角色的生成、求交、和 AI 的撰写。初期在键盘控制下效果较好，后期替换为声控模块后出现了一系列由于电路延迟产生的问题，用了较多时间对模块进行优化，最终实现了比较灵敏的控制效果。

展示结束后对于逻辑模块进行了反思，认为加入声控模块后出现的延迟可能是由于以下两个方面。一是没有考虑到硬件对于 if-else 结构的优化。如我们在 if 条件下将 flag 置为 1，然后并没有在 else 条件下对 flag 进行赋值。此时认为在 else 条件下 flag 仍保持原值，但实际并不一定如此。二是没有除去所有的 warning 信息，在小组展示中某一小组提及他们在分别编程时除去了所有的 warning 信息，最后在合一时一次通过，由此想到平时没有注意到的 warning 问题可能在最后造成了较大困扰。

【显示、存储模块】



图中箭头代表信息流方向。

关键技术分析：

1. bmp 文件格式分析、读取，并转换为需要的格式，实现过程中遇到几个问题。第一，不是所有的 bmp 文件都是 24 通道，有些是 32 通道（带 alpha 值），这时则需要手动转换（其实可以利用 alpha 值来避免显示时由于图片是矩形而带来的重叠问题）。第二，bmp 文件为了对齐，每行字节数需要是 4 的倍数，不足补齐。

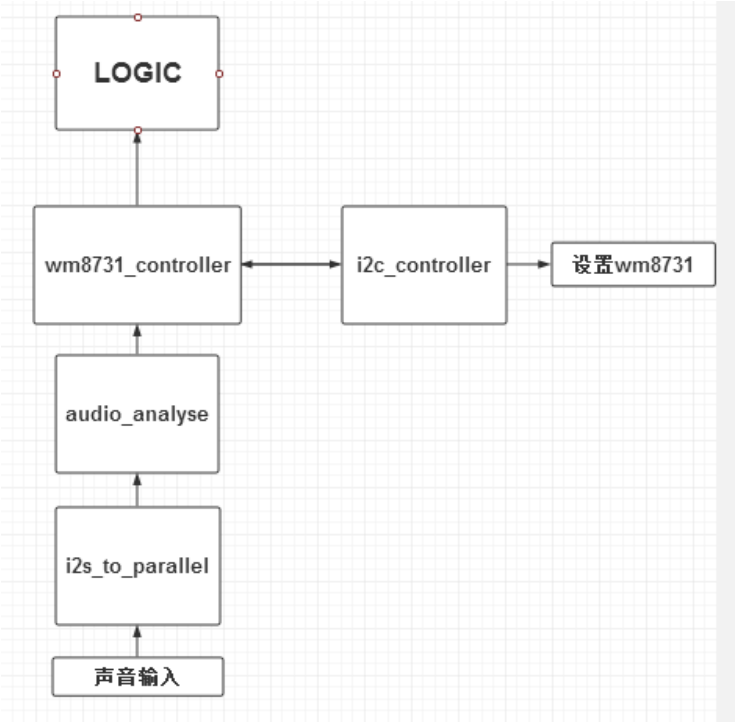
2. wav 文件格式分析、读取并转换为需要的格式。其中关键在于补码的识别、大端法和小端法的转换、左右声道的区分，以及尽量节省空间（wav 文件为无损格式，非常大，在降低采样率的情况，最多也只能存 5 秒左右的声音）。

3. sram 的设置、写入。这部分较为简单。由于 sram 本身频率较高，在一个上升沿设置地址位，在下一个上升沿读取即可。

4. 美工什么的，都是泪啊

【音频处理模块】

一、 总控制



设计中由 wm8731_controller 进行总调度，先调用 i2c_controller 通过 I2C 协议设置 wm8731，当设置成功后转入传输状态，通过 I2S 协议接受音频信号。详见源码 wm8731_controller.vhd

二、 I2C 协议设置 wm8731

a) I2C 协议：

开始信号：让时钟信号 SCL 保持高电平，使数据信号 SDA 由高电平变为低电平，即告知 wm8731 将开始传输数据设置 wm8731

结束信号：让时钟信号 SCL 保持高电平，使数据信号 SDA 由低电平变为高电平，即告知 wm87331 已经设置完毕，结束 I2C 协议传输

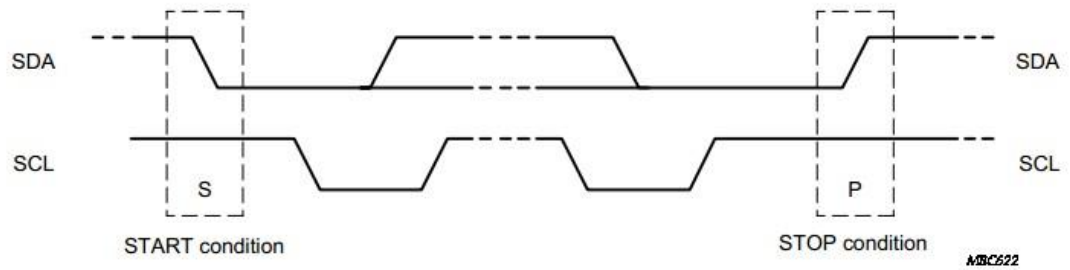


图 5 起始和停止条件

传输过程：wm8731 会在时钟信号 SCL 为高电平时取走 SDA 上的数据，故在 SCL 为高电平时，SDA 的数据应保持不变，否则将视为开始信号或结束信号。在 SCL 为低电平期间，SDA 的数据改变为下一位数据位。

响应信号：I2C 协议每次需串行传输 24 个数据位（即 3 个字节），而每传输 8 位（即一个字节），SDA 线会返回一个响应信号，若响应信号为低电平，则代表传输成功，否则代表传输失败，需重新进行传输。响应信号在 SCL 为高电平时读取。故 SDA 应设为 inout 接口，在时钟信号 SCL 的前 8 个周期向 wm8731 发送数据，在第 9 个周期接受响应信号。

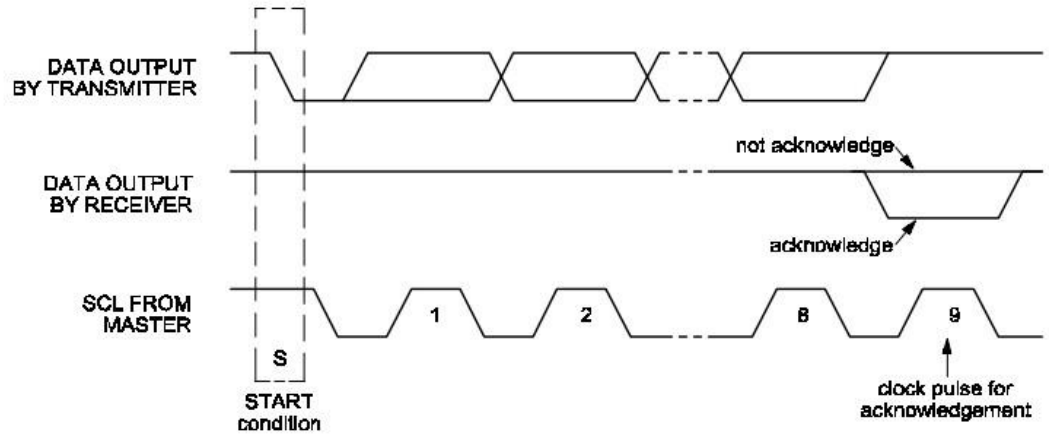


图 7　I^2C 总线的响应

传输的数据：24 位数据中，前 8 位为 wm8731 的设备号，固定为 X”34”（00110100）；剩下的 16 位中，前 7 位为 wm8731 寄存器地址，后 9 位为要设置的数据，即将后 9 位数据写在 wm8731 的相应寄存器中。

时钟信号 SCL：wm8731 允许的传输频率为 400KHz 以内，本设计采用 20KHz 的时钟信号进行传输。这一部分的源代码见 i2c_controller.vhd，源代码中 i2c_sclk 对应 SCL，i2c_sdat 对应 SDA。

b) 设置的参数：

```
with index select
    set_data <= X"001F" when 0,
                X"021F" when 1,
                X"0479" when 2,
                X"0679" when 3,
                X"0810" when 4,
                X"0A06" when 5,
                X"0C00" when 6,
                X"0E4A" when 7,
                X"1000" when 8,
                X"1201" when 9,
                X"ABCD" when others; --should never happened
```

c) 遇到的问题：先后尝试过三种写法，前两种为较精简的状态机，但总是设置不成功，最后采用有 61 个状态的状态机，精确描述了 24 位数据的传输过程。

三、 I2S 协议传输数字音频信号

I2S 协议：I2S 协议具有 5 根数据线，分别为位时钟 BCLK，数字转模拟采样率时钟 DACLRC，模拟转数字采样率时钟 ADCLRC，数字转模拟数据信号 DACDAT，模拟转数字数据信号 ADCDAT。本设计使用了 BCLK，ADCLRC，ADCDAT 三根数据线，以接受人声（模拟信号），wm8731 将模拟信号转为数字信号 DACDAT 后，再对数字信号进行分析处理。

在 I2S 协议中，数据的最高位总是出现在 LRC 跳变之后第 2 个 BCLK 脉冲处，在本设计中，在 LRC 跳变之后的 BCLK 第 2 个上升沿开始读取数据，之后均在上升沿读取数据，详见源代码 i2s_to_parallel.vhd，该部分代码负责将串行传输的 24 位数据信号转为并行的 24 位数据信号。

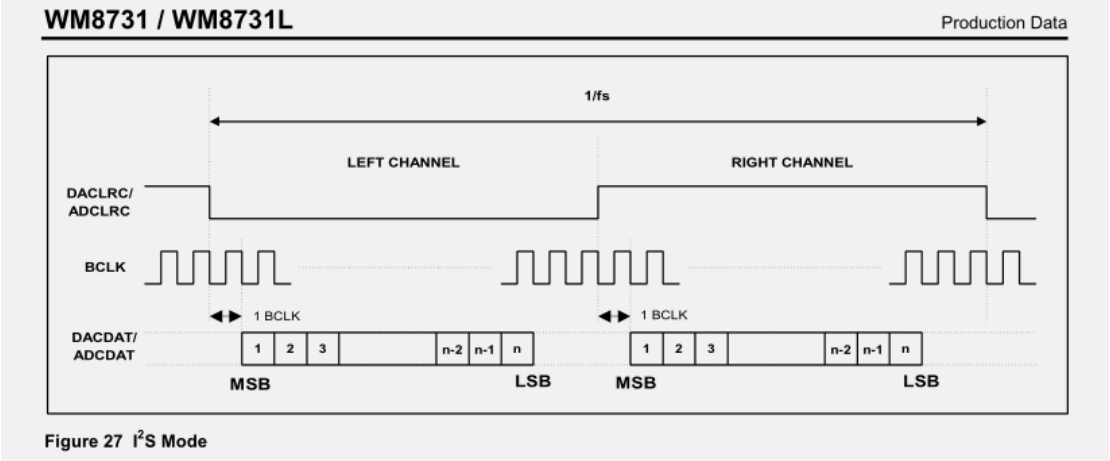
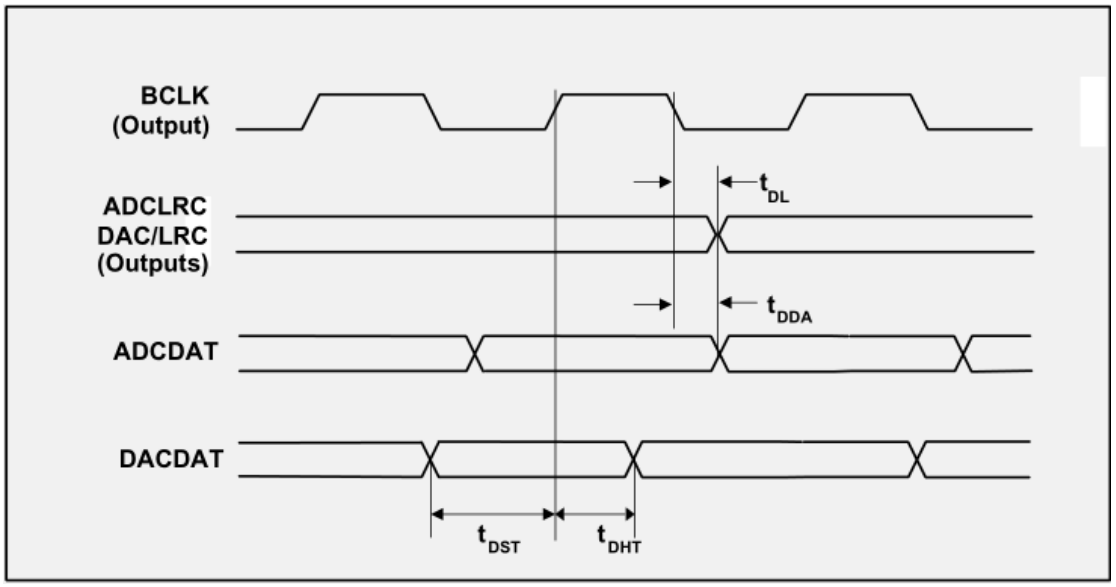


Figure 27 I²S Mode



四、 声音分析

a) 原理：本设计采用以声强区分不同的声音信号，通过取得一段时间内的声音的最大振幅作为该段时间的声音标记。之后根据该标记的大小分为16个等级，当等级小于等于1时，视为无声，不采取任何操作；当等级大于等于2，小于等于10视为游戏中鸟的上升信号；当等级大于等于11，小于等于15视为游戏中鸟的发子弹信号。在设计中我们发现，“啊”声音的等级一般为2、3、4，爆破音的等级一般为10以上，故在游戏中发出“啊”声即控制鸟上升，发出“啪”声或朝话筒狠吹一口气即控制鸟发子弹。详见源代码audio_analyse.vhd。

b) 其他设计：本打算采用声音的持续时间作为判断标准，当声音持续时间较长时，视为上升信号，较短时视为发子弹信号，但是效果并不理想，故放弃。

五、 未解决的问题

在设计中本想在使用AD模块接收声音的同时，使用DA模块播放一些背景声音，然而发现DA模块会对AD模块产生一些干扰，表现为在播放声音时游戏鸟会自动上升下降。

【总结】

在本次硬件编程实验中，我们小组的内部进行了较好的分工与合作，最终都认为有了较大的收获。同时我们也思考了一些欠缺的方面，如没有进行有效的版本控制。没有使用git是一大失误，造成我们在合作时要频繁地进行代码交换，有时还要考虑当前使用的是否为最新的代码。还有就是对vhdl语言的理解还有所欠缺，如quartus对于vhdl的编译问题等。