示波器上的弹球游戏的设计

RainEggplant[1], xxx[2]
（1.无 76 2017******；2.无 76，2017******）

0 课程任务和实验要求

0.1 课程任务

《电子电路课程设计》是一门综合应用模拟电路和数字电路理论进行电子系统设计的课程，要求设计并制作具有较完整功能的小型电子系统，它侧重于电子技术理论知识的灵活运用和设计的创新，因此具有系统性、综合性和探索性。课程任务有：

（1）掌握一般电路系统的设计思路和方法

（2）培养系统观念和工程观念、解决电路实际问题的能力和探索创新精神

（3）培养实验研究的总结和表达能力

0.2 实验要求

设计并制作一个示波器上的弹球游戏。

（1）基本要求：

示波器上显示“弹球” 和“球拍”。

（2）提高要求：

a. 球拍没有接到球时，球不应当继续反弹；

b. 计数连续击球数目。

此外，为了提高游戏体验，增加游戏趣味性，在基本要求和提高要求的基础上，我们还增加了以下功能：

c. 显示游戏边框；

d. 小球的移动速度会随分数增加而加快；

e. 可以随时重新开始游戏。

1 实验设计

本次实验设计的示波器上的弹球游戏系统综合运用了模拟电路与数字电路。由 FPGA 生成用于显示弹球、球拍以及边框的数字 X,Y 信号，并实现游戏的控制逻辑与计分；由面包板上的电路实现球拍位置的控制、ADC、DAC 等。

整体电路框图如图 1。表示球拍位置的电平在 FPGA 控制下，经过 ADC 转换为数字信号，输入 FPGA。FPGA 控制弹球的移动、反弹与加速，根据弹球与球拍位置判定是否成功接球，并实现计分。因为弹球、球拍、边框的显示共用一对 X, Y 端子，所以 FPGA 需要以快速扫描切换的方式，轮流输出三者的位置信号，从而利用示波器的余辉以及人眼的视觉暂留效应，达到在示波器上“同时”显示三者的效果。

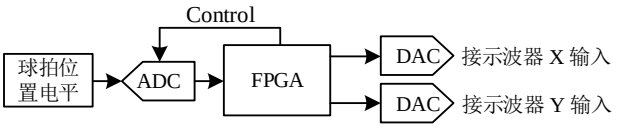


图 1 整体电路框图

1.1 游戏接口标准设计

首先根据示波器上的预期游戏界面，设计数字电路与模拟电路之间的接口标准。

该游戏占用的区域应当为 $[0, X\_MAX] \times [0, Y\_MAX]$ 的矩形区域。充分考虑小球位置的精细度需求和复杂度之后，我们决定分别用 8-bit 信号来表示 X, Y 坐标。

同时，为了减小周期性，$(X\_MAX + 1)$ 和 $(Y\_MAX + 1)$ 需要为复杂整数比。因此取 $X\_MAX = 2^8 - 1 = 255$, $Y\_MAX = 220$。

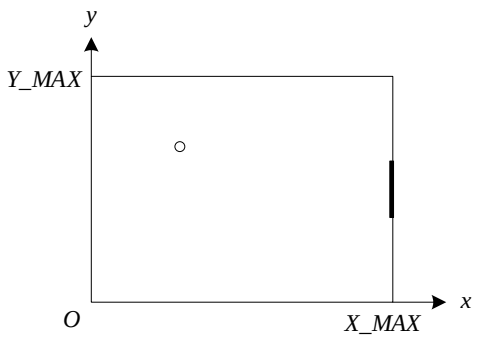


图 2 示波器界面

1.2 FPGA 程序设计

FPGA 程序主要包含以下几个模块：

(1) 时钟分频模块

实验时间：8 月 28 日-8 月 29 日上午

实验地点：清华大学中央主楼 903 室

(2) 游戏逻辑控制模块
(3) 界面模块
(4) 分数显示模块
(5) ADC 控制模块

1.2.1 时钟分频模块

FPGA 开发板本身提供的系统时钟频率为 100 MHz，实际使用不到这么高的频率，所以需要进行分频。

我们采用了计数满则翻转的方式，利用系统时钟获得了 1 kHz 的 clk_ctr, 5 kHz 的 clk_refresh 和 100 kHz 的 clk_100k 时钟。

对应代码请见附件：src/oscilloscope_pong/clk_gen.v

1.2.2 游戏逻辑控制模块

分析弹球游戏的过程，可以构建如下模型：

设小球的坐标为 (x_b, y_b)。 v_x, v_y 指示小球沿 x 或 y 的速度方向，1 表示正方向，0 表示负方向。那么，x_b, y_b, v_x, v_y 均可视为有限状态机，其状态转移方程为：

$$x_b^+ = \begin{cases} x_b + 1, & v_x = 1 \text{ and } x_b \neq X_{MAX} \\ x_b - 1, & v_x = 0 \text{ and } x_b \neq 0 \end{cases}$$

$$y_b^+ = \begin{cases} y_b + 1, & v_y = 1 \text{ and } y_b \neq Y_{MAX} \\ y_b - 1, & v_y = 0 \text{ and } y_b \neq 0 \end{cases}$$

以上两式即实现了小球的移动。

$$v_x^+ = \begin{cases} \sim v_x, & (v_x = 1 \;\&\&\; x_b = X_{MAX}) \;||\; \\ & (v_x = 0 \;\&\&\; x_b = 0) \\ v_x, & otherwise \end{cases}$$

$$v_y^+ = \begin{cases} \sim v_y, & (v_y = 1 \;\&\&\; y_b = Y_{MAX}) \;||\; \\ & (v_y = 0 \;\&\&\; y_b = 0) \\ v_y, & otherwise \end{cases}$$

以上两式即实现了碰到边界时小球的反弹。

游戏开始时，设定 $x_b = X_{MAX}/2$, $y_b = Y_{MAX}/2$, $v_x = 1$, $v_y = 1$，小球即从中央向右上方移动。

游戏逻辑控制模块 game_ctr 以 clk_ctr 为工作时钟，接收球拍的中心坐标 y_{p_mid} 作为输入。设球拍宽度的一半为 PLATE_HALFWIDTH (之后简写为 PH), 则球拍的范围应为 $[y_{p\_min}, y_{p\_max}]$。其中，

$$y_{p\_min} = \begin{cases} y_{p\_mid} - PH, & y_{p\_mid} \geq PH \\ 0, & otherwise \end{cases}$$

$$y_{p\_max} = \begin{cases} y_{p\_mid} + PH, & y_{p\_mid} \leq Y_{MAX} - PH \\ Y_{MAX}, & otherwise \end{cases}$$

于是，在小球向右运动到最远时，即 $v_x = 1 \;\&\&\; x_b = X_{MAX}$ 时，进行成功接球与否的判断：当 $y_b \in [y_{p\_min}, y_{p\_max}]$ 时，即判定接中，计分器加 1。否则游戏结束，计分器清零，重新开始游戏。

以上部分只是基础的游戏逻辑，要实现小球速度随分数增加而加快，还需要做进一步修改。

我们定义了 cycle_between_step 变量，表示小球每两次移动中间间隔的时钟周期数（小球移动要单独占用一个周期）。最开始的时候，cycle_between_step 为 15，随着分数增加，cycle_between_step 最终会达到 1（没有启用 0，因为实在是太快了）。从而小球就会越变越快。

本节对应代码请见附件：src/oscilloscope_pong/game_ctr.v

1.2.3 界面模块

小球的坐标信号已经在上一节的游戏逻辑控制模块中产生。本模块需要负责产生球拍和边框的坐标信号，同时负责扫描轮流输出三者的坐标信号。

球拍坐标信号由 plate_view 模块产生，该模块以 clk_100k 为工作时钟，接收球拍的中心坐标 y_{p_mid} 作为输入，采用与上一节相同的方式，得到球拍的范围。然后固定输出 X 信号为 X_{MAX}，扫描输出球拍范围内的值作为 Y 信号。

边框坐标信号由 border_view 模块产生，该模块以 clk_100k 为工作时钟，扫描输出左、上、下三边的坐标值，形成边框。

扫描切换输出由 output_view 模块产生，该模块以 clk_refresh 为工作时钟，接收小球、球拍、边框的坐标信号作为输入，然后轮流输出三者信号到 DAC。

本节对应代码请见附件：src/oscilloscope_pong/plate_view.v, src/oscilloscope_pong/border_view.v, src/oscilloscope_pong/output_view.v。

1.2.4 分数显示模块

分数显示模块 ssd_display 以 clk_ctr 为工作时

钟，接收游戏逻辑控制模块的分数 score 为输入，实现二进制数字到 BCD 编码（bin_to_bcd 模块）再到七段数码管编码（bcd_to_ssd 模块）的译码以及七段数码管的扫描显示，从而实时地显示当前分数。

本节对应代码请见附件：src/oscilloscope_pong/ssd_display.v, bin_to_bcd.v, bcd_to_ssd.v。

1.2.5 ADC 控制模块

ADC 控制模块 dual_slope_adc 负责控制 ADC 的工作并返回转换结果。具体说明请见 1.4 节。

1.3 球拍控制电路设计

球拍控制使用一个 50kΩ 双联电位器。其内部电路如下图：

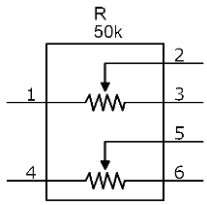


图 3 50 kΩ双联电位器内部电路

1 端接地，3 端接 5V 直流，取 2 端电压作为输出。这样，在玩家通过旋钮改变电位器 1、2 端间的阻值时，输出电压就在 0-5V 间变动。

为了消除后面负载对该控制电路的影响，我们在输出后又接入了一个电压缓冲器。最终的电路如下图所示：

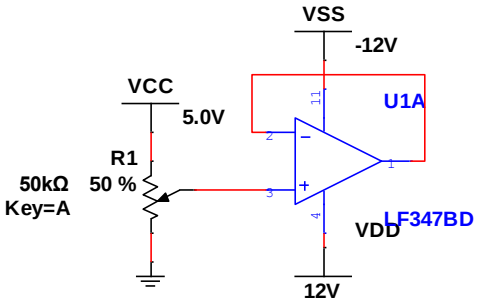


图 4 球拍位置控制电路

1.4 ADC 设计

1.3 节中球拍位置控制电路输出的是模拟信号，我们需要将其转换为数字信号以输入 FPGA 进行处理。这里我们采用 8 位双斜式积分 ADC 电路。

1.4.1 双斜式积分 ADC 的原理

下图是双斜式积分 ADC 的基本电路图：

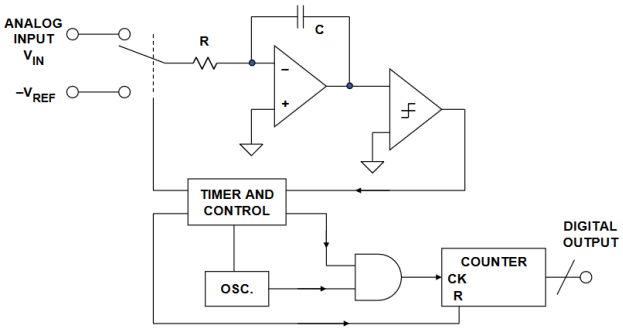
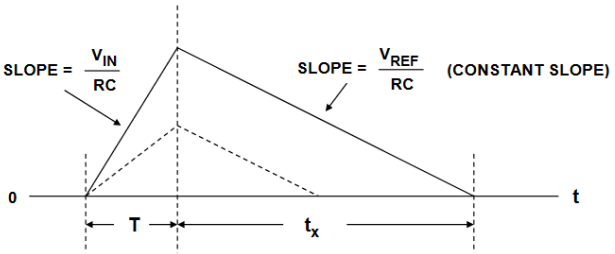


图 5 双斜式积分 ADC 基本电路图

模拟输入电压作用于积分器，同时计数器从 0 开始计数。经过一段预设的时间 T 后，具有相反极性的参考电压被作用于积分器。在此刻，积分电容上的电量和 0~T 时间内输入电压的平均值是呈正比的。

此后，在参考电压的作用下，电容开始反向充电，同时计数器再次从 0 开始计数。当积分器的输出为 0 时，计数器停止计数。

积分器的充放电过程如图 6，注意充电电荷量和放电电荷量是相等的。



$$\frac{V_{IN}}{RC} T = \frac{V_{REF}}{RC} t_x$$

$$t_x = \frac{V_{IN}}{V_{REF}} T$$

因此，若充电阶段计数器计数为 COUNT，放电阶段计数为 $count_x$，则 $\frac{count_x}{COUNT} = \frac{t_x}{T} = \frac{V_{IN}}{V_{REF}}$。如果 V_{REF} 对应的数字表示为 COUNT，则 V_{IN} 对应的数字表示就为 $count_x$。

1.4.2 本实验使用的双斜式积分 ADC 的设计

基于以上原理，我们设定本实验的 8 位 ADC 计数区间为 0~255。由于球拍位置电平范围为 0~5V，故参考电压取 -5V。

由于该游戏对延迟的要求较高，计数器的工作

频率应尽量高，使得一次转换操作的用时尽量短。取最坏情况考虑，即 $V_{IN} = -V_{REF}$，此时计数器需要两次从 0 计数到 255，故总用时 $t = \frac{512}{f}$ 。我们最后取频率 f = 100 kHz，故最坏情况下一次转换耗时 5.12 ms，这样的延迟是完全可以接受的。

下面开始设计 ADC 的各部件。

首先是用于选择模拟输入电压和参考电压的多路选择器。这里我们选择 CD4052 多选器。

由于 Multisim 的器件库里没有 CD4052，我们选择了功能相似的 ADG409BN 搭建电路和仿真。

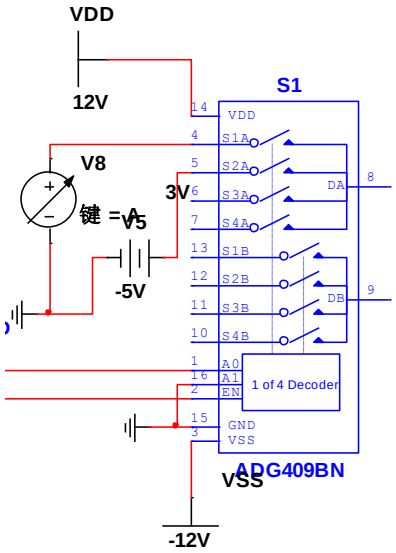


图 6 多路选择器

图 6 中可变直流电压源 V8 是用来模拟球拍控制电路的输出。在实际电路中，S1A 端口直接与球拍控制电路的输出相连。

-5V 直流电压源 V5 即为参考电压。在实际电路中采用 5V 直流电源和增益为 -1 的反相放大器实现（如图 7）。

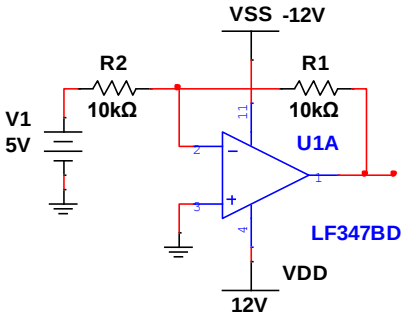


图 7 增益-1 的反相放大器

A0 是用来选择 DA 端输出的信号。查阅 CD4052 手册，我们发现 FPGA 的高电平（3.3V）必须经放大后才能与 A0 端相连，直接使用会被视作低电平。因此我们搭建了 3 倍的同相放大电路，如图 8。

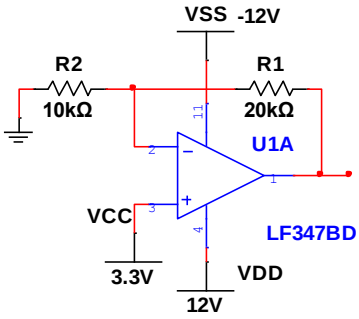


图 8 3 倍同相放大电路

DA 端是多路选择器的输出，为了消除负载效应，我们在 DA 之后添加一个电压缓冲器（图中未绘出）。

接下来是积分器（如图 9），这里选择 5.1 kΩ 电阻和 1μF 电容，是因为充电时有下式成立：

$$-\frac{V_{IN}}{R}t = CV_C$$

取 V_{IN} 为最大值 5V，则充电完成时，电容上的电压为

$$V_C = -\frac{5}{5.1\text{k}\cdot 1\mu}\cdot\frac{256}{100\text{k}} \approx -2.51\text{V}$$

处于安全、有效范围内。

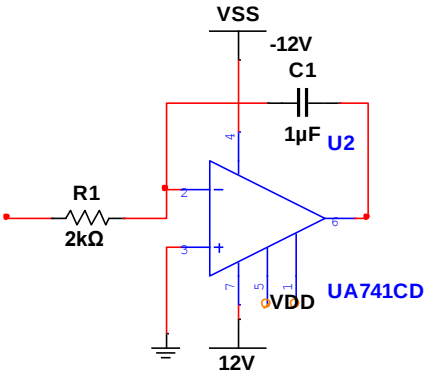


图 9 积分器

接下来是比较器、电平匹配和整形电路。比较器用于比较积分器输出电平与 0 电平的大小关系。由于运放作比较器的输出信号幅度（-10~10V）与数字电路的输入电平要求（低电平不能低于 0V，高电平不能高于 V_{CC}）不匹配，故需要加入电平匹配电路。

74 系列 V_{CC} 为 5V，FPGA 的 V_{CC} 为 3.3V

匹配后的输出波形前后沿不陡，当电压处在数字电路的开关电压之间或附近时，其噪声容限低，可能会在输出端出现若干个毛刺脉冲，为此还需增加整形电路。

采用两个施密特反相器单元即可实现电平匹配和整型电路。因为我们需要把信号输入 FPGA，所以我们采用 FPGA 上的 3.3V 电源为反相器 CD40106 供电（不是图上显示的 5V）。

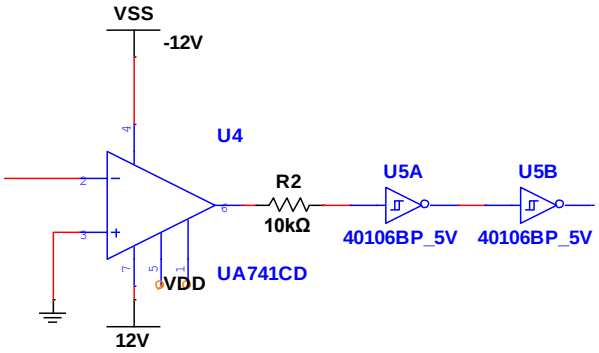


图 10 比较器、电平匹配和整形电路

以上就是 ADC 在面包板上的部分。该部分接收 FPGA 输出的多路选择器选择信号（记作 select_v_ref），使用球拍位置电压或者参考电压进行积分，并将积分器输出电平是否为负（记作 is_neg_v）作为输出提供给 FPGA。

下面是 FPGA 上的 ADC 控制模块的设计。

该模块该模块以 clk_100k 为工作时钟。工作时，首先置 select_v_ref 为 0，选择球拍位置电压积分 256 个周期。然后置 select_v_ref 为 1，清零计数器，选择参考电压进行积分。在用参考电压积分的过程中，当 is_neg_v 变为低电平，即积分器输出电平为正时，将参考电压积分周期数赋给 digit_val（此即模拟信号经 ADC 转换的结果），同时置 select_v_ref 为 0，清零计数器，开始下一轮转换。

该部分代码请见附件：src/oscilloscope_pong/dual_slope_adc.v

1.4.3 以上设计的非 FPGA 版本

在最开始设计 ADC 时，我们没有想到使用 FPGA 进行控制与计数，便采用了集成数字电路完成了这一部分。虽然该部分内容最后被 FPGA 版本替代，但其对我们的设计、分析能力起到了很好的锻炼，同时对仿真 ADC 电路也提供了一些帮助。因此这一版本也作为附录附在报告后。

1.4.4 ADC 电路的仿真结果

我们使用 1.4.3 中的版本对 ADC 电路进行了仿真，仿真结果如下：

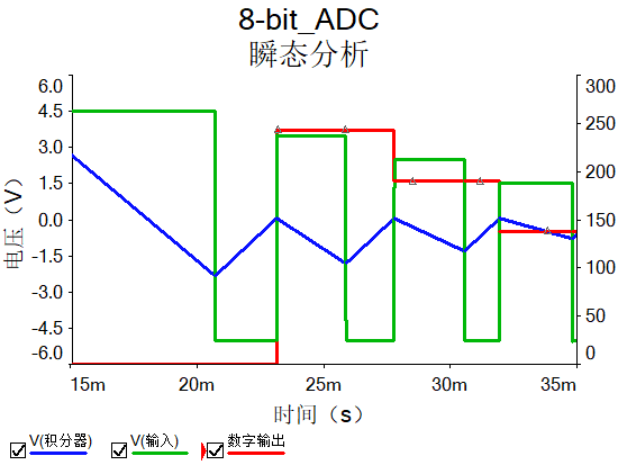


图 11 ADC 仿真结果

可以看到，ADC 按预期正确工作。首先，ADC 能自动进入工作状态；其次，ADC 正确地实现了输入信号的切换；最后，采用不同的电压作为输入，ADC 均产生了正确的结果。

1.5 DAC 设计

DAC（数模转换电路）的形式有多种，主要有比例放大法和梯形加权网络法。

虽然比例放大法电路简单，但是理论上，各加权电阻对计数器的负载效应不同，每个阶梯高度很难一致，实际电路中，电路对电阻精度要求高，且实验室中并没有连续 8 个 2 等比阻值的电阻，或者使用相同阻值的电阻但工作量巨大不可实现。另一方面，实验室电阻 1%的误差也会使该 ADC 失真严重，且难以手动校准。

于是我们采用了梯形加权网络法，使用 9 个 2kΩ 电阻和 7 个 1kΩ 电阻实现，结构如下图，底部从左到右依次是高位到低位的输入。

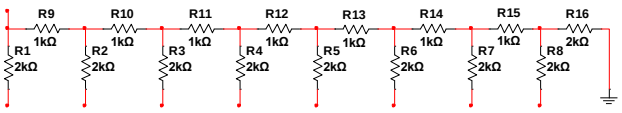


图 12 DAC（数模转换电路）电路

电路输出结果仿真如下图（仿真为从 255 到 0）。

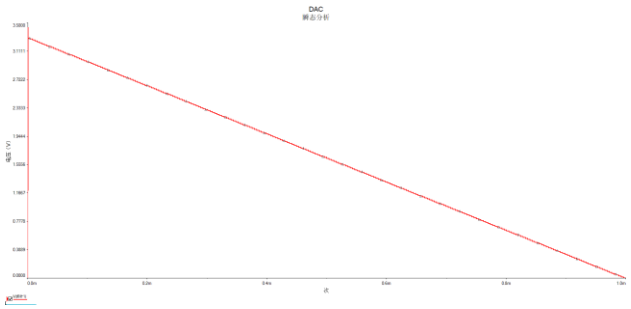


图 13 DAC（数模转换电路）仿真结果

可见，8-bitDAC 的阶梯宽度相当小，线性度较高，实际显示时相当于将 0~3.3V 分成了 256 份，在示波器长宽内人的肉眼较难分辨，球在弹跳过程应该会很连续。

为了方便搭建实际电路时对 DAC 进行调试，我们特意编写了 FPGA 测试程序。该程序能够周期性输出从 0 到 255 的周期信号。源代码见附件文件夹：src/0_to_255 。

2 实验数据整理和分析

因为采用 FPGA 完成了游戏的控制与显示部分，故能够用示波器采集到的数据并不多。下面主要列出实验中的现象。

下图为示波器上显示的小球、球拍和由滚动的小点组成的边框。

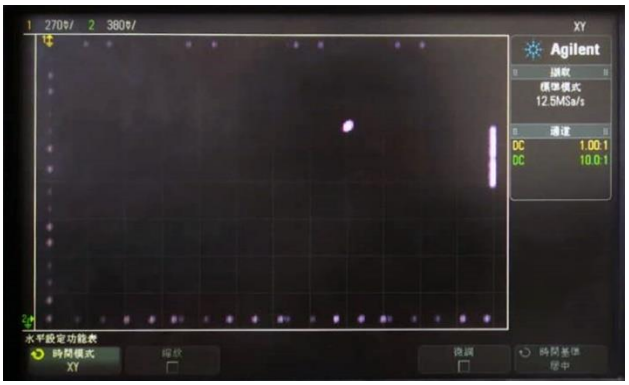


图 12 示波器上的游戏界面

在调节 FPGA 程序中的 clk_refresh 时钟的频率时，我们注意到了一个有趣的现象。如果该时钟频率过低，显示的球拍就不再是一条连续的直线，而会出现间断；若时钟频率过高，则小球就会出现明显的拖影。

分析其原因，是因为我们采用的是扫描切换输出的方式，轮流输出小球、球拍以及边框的坐标值，并且瞬时输出的信号只能是一个点。如果时钟频率过低，则输出不能及时地在三者间切换，超过了示波器的余辉效应和人眼的视觉暂留效应，显示的东西就自然是断断续续的。如果时钟频率过高，由于输出的转换是不可能瞬间完成，存在一定的变化过程的，因此切换太快时，这一不理想因素就显露出来，体现为拖影。

值得一提的是游戏界面中的“滚动小点边框”。实际上我们本来是想做成实线边框的，但是由于边框所占的空间跨度很长，包含的点数很多，因此在我们设定的 clk_refresh 频率下出现了明显的间断。同时，由于每次切换到边框时边框的输出都不一致，便出现了滚动的效果。修正的思路并不复杂，需要调高 clk_refresh 的频率，但在切换输出时，要让边框显示占据多个时钟周期。但是由于这个效果很“酷炫”，可以说是意外之喜，我们便保留了这个特性。

为了使得游戏效果更加“酷炫”，可以调整示波器的 display - 余辉设置，将持续性设为可变，时间设为 5~8 秒，这样小球在运动时就可以产生流星般逐渐消逝的尾迹。如下图：

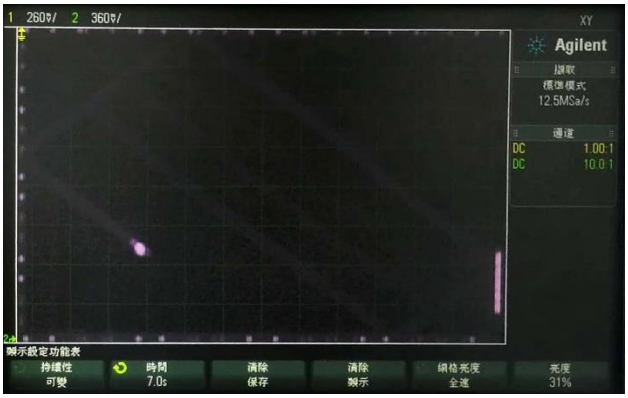


图 13 开启可变余辉后的酷炫效果

下图为 FPGA 上的分数显示效果。另外，通过按下 FPGA 右侧中间的按钮，可以实现重新开始游戏。

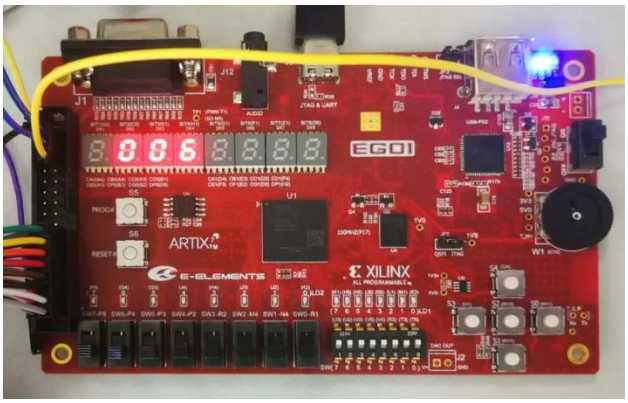


图 14 FPGA 上的分数显示

下图为调节球拍旋钮时积分器的输出。拍摄本图时，玩家正在向右旋转电位器的旋钮（即增加输出电压）。可以看到波形和仿真波形是一致的。

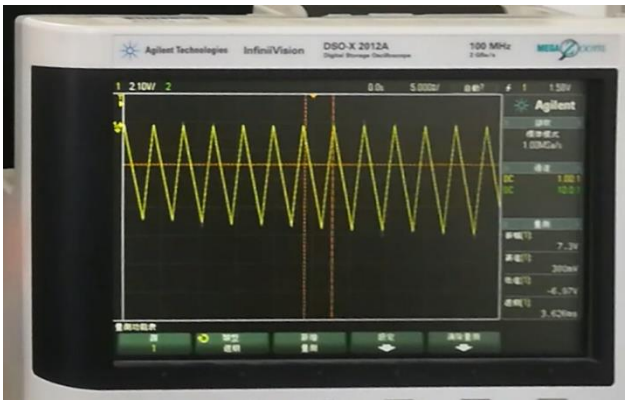


图 15 调节球拍旋钮时积分器的输出

3 实验总结

本次实验中，我们很好地完成了所有的基本要求和提高要求，并在该基础上发挥想象力，增加了新功能，提高了游戏的体验和趣味。

附件中提供了一个简短的展示视频。该视频展示了游戏界面、计分器和操作方式，并体现了特色功能（小球自动加速、滚动小点边框）。

虽然实验过程相对来说比较顺利，但设计电路的过程并没有那样一帆风顺。最开始我们小组两人讨论时提出了两种设计电路的思路：B 同学提出使用模拟电路和集成数字电路实现，而 A 同学则提出利用 FPGA 实现。于是我们决定分头行动，准备两套方案。

最初，B 同学试图“简单”地以 555 定时器为核心实现要求，如：小球的两个三角波可以不那么准确，直接通过 555 定时器产生一个占空比并不为 50%的方波（设计可通过 Multisim 电路向导工具直接产生，见下图）在通过一个时间常数较大的 RC 电路即可产生一个近似三角波的信号。同样，板子实际上也并不需要标准的三角波或正弦波，也可以直接采用这个电路，而小球与板子的转化信号也可以通过下图去掉 RC 电路后产生的方波实现，通过大量重复简单的设计完成整个实验，最初设计便不一一列举了。但是即使将实现目标降低，如小球移动移动轨迹可能不太直等，设计及搭建仍然较为繁杂。

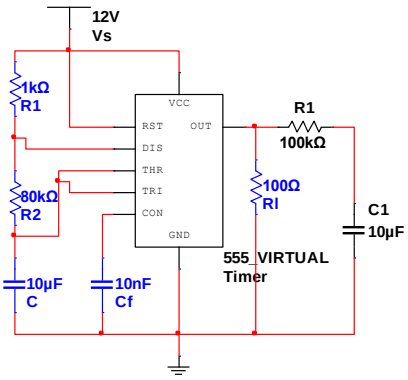


图 16 555 定时器电路产生近似方波与近似三角波

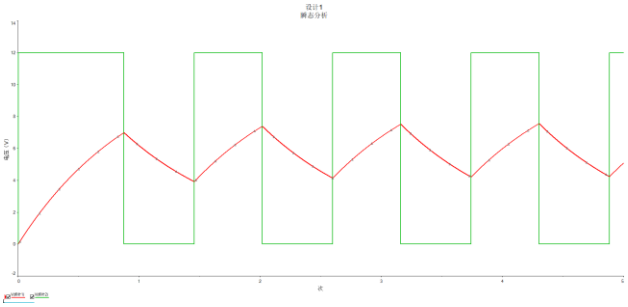


图 17 近似方波与近似三角波的仿真波形

而 A 同学在利用 FPGA 实现设计要求时，最开始并没有想到可以使用 FPGA 控制 ADC 电路并实现计数，因此设计了较复杂的集成数字电路来实现控制与计数。这样就浪费了 FPGA 的优势。

最终，为了高效并高质量地完成实验，我们在以 FPGA 方案上做了进一步改进，改用 FPGA 进行 ADC 的控制与计数，快速地完成了实验。但是实验过程中也出现了一些问题。

实验中，虽然 FPGA 产生的数字信号精准度相当高，但是当经过 DAC 转化为模拟信号时，受限于电阻精度不够，导致最后的输出偏差较大，这也是数模转化中一个最常见的问题。

DAC 最主要的问题就是在数模转化时偏差较大，最开始我们采用万用表选出偏差小的电阻，但是最终还是会出现较大偏差，这种偏差是系统性的，在屏幕上的表现为会卡顿甚至回退，如下图所示：

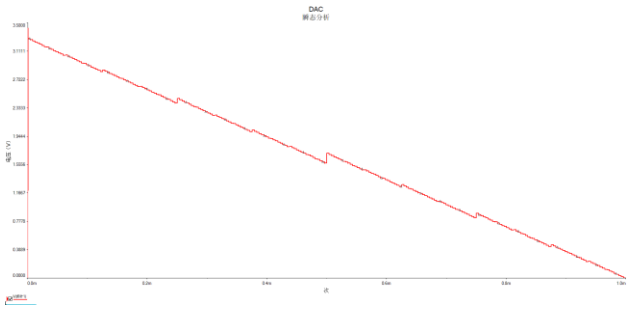


图 18 DAC 出现的问题波形

可见，在周期的 1/2、1/4、1/8 等处出现较大的偏差，主要原因是是梯形加权网络中，权值越高的输入旁的电阻误差对整个 DAC 的影响越大，而这个误差通过选高精度电阻来解决基本不可能实现，最终我采用最高位输入旁的电阻由电位器替代，并手动校准，解决了 DAC 输出电压断崖式的上升或下降。

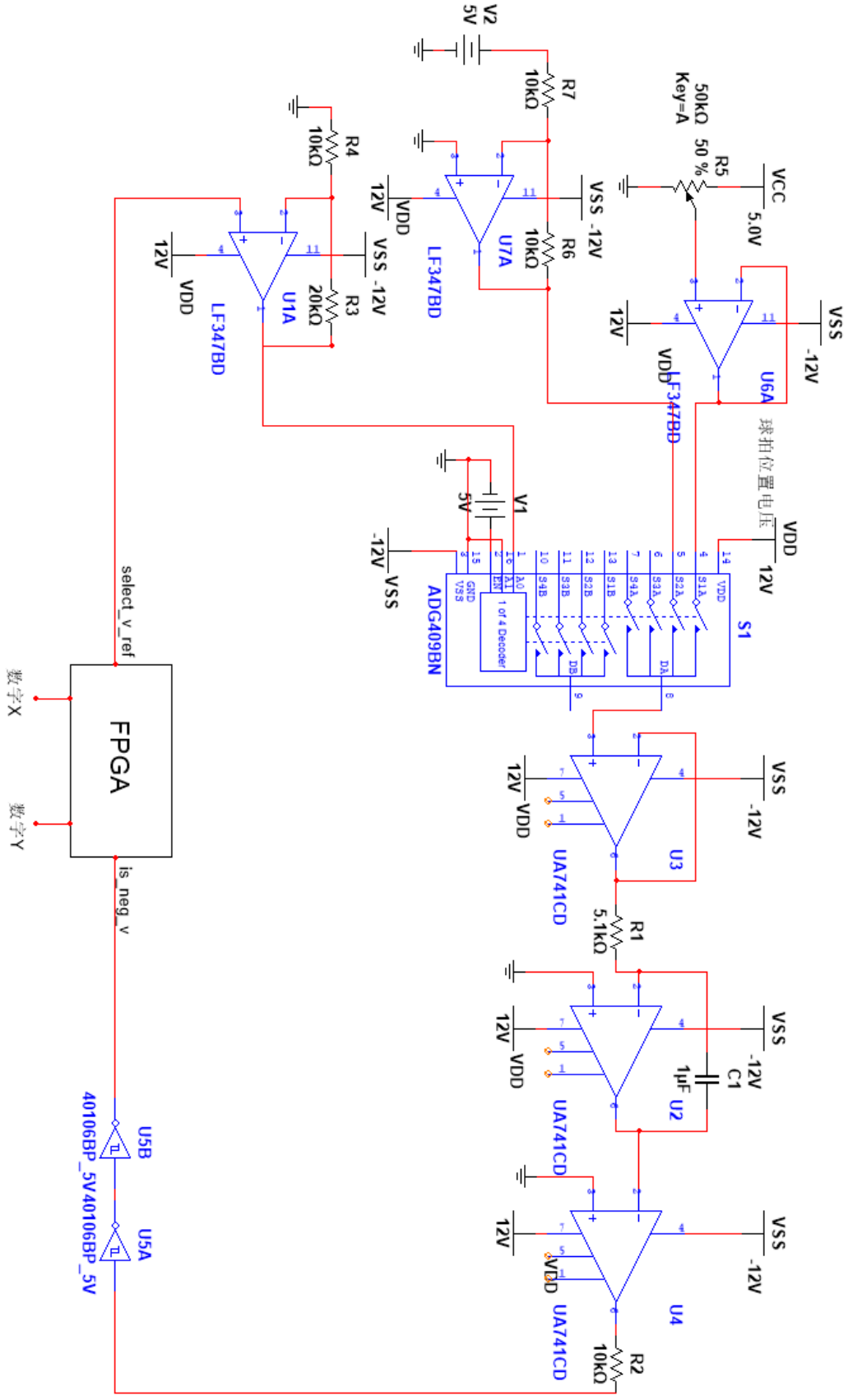
DAC 另一个问题就是如果将输出直接接示波器，会导致小球与球拍不停抖动，幅度虽然较小但是肉眼可察。原因应该是输出电压在小球和板子之间来回高频切换，但是由于输出阻抗较大，探头输入电容无法及时充放电。为了解决这个问题，我们在 DAC 输出和探头间接入一个电压缓冲器，使得示波器显示平稳，小球也能在屏幕上“顺滑”地移动。

总之，通过这次自主设计实验，我们加深了对于模拟电路、数字电路知识的理解，锻炼了知识的运用能力与电路的设计、分析、调试能力。此外，我们也充分体会到了数字电路在信号处理上的优势，尤其是 FPGA 这种可编程数字元件的威力。同时，我们也深刻体会到数模转换、模数转换器件的质量（如精度、转换速度）在沟通数字电路与模拟电路中的重要性。

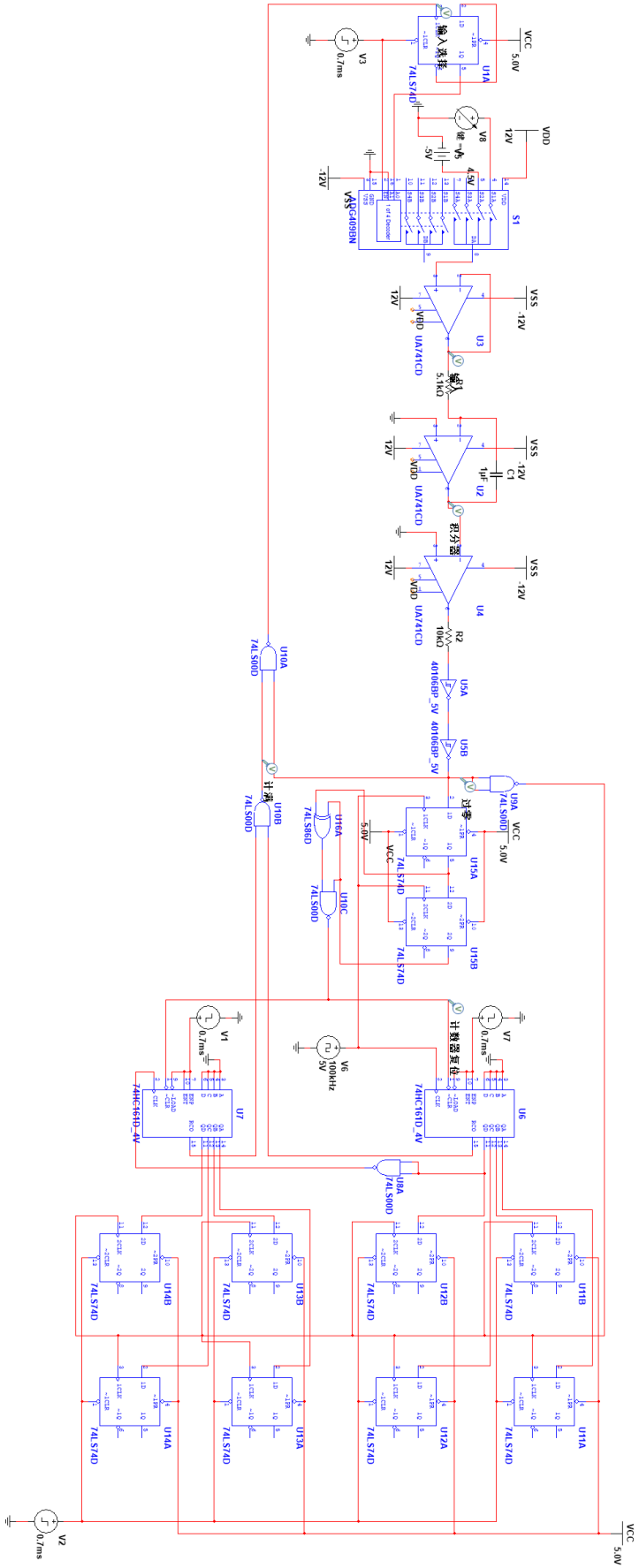
这次实验总的来说“开心且有收获”。最后，感谢老师和助教的辛苦付出！

附录 A：电路原理图

下图为除 DAC 外的所有电路：



下图为不采用 FPGA，而使用集成数字电路进行控制和计数的 ADC 电路：



附录 B：实验原始数据

本实验不属于测量型或验证型实验，故没有需要记录的数据。

附录 C：成本分析

FPGA 的资源利用情况如下：

Resource	Utilization	Available	Utilization %
LUT	197	20800	0.95
FF	158	41600	0.38
IO	30	210	14.29

图 1 FPGA 资源利用情况

因此，整体利用率为 $\frac{197+158+30}{20800+41600+210} \times 100\% = 0.615\%$. FPGA 的芯片价格为 350 元，故折算成本为 $350 \times 0.615\% = 2.15$ 元。

其余使用到的元器件如下：

表 1 元器件成本

Tab.1 Component Cost

元件	数量	单价（元）	总价（元）
1 kΩ 电阻	14	0.1	1.4
2 kΩ 电阻	16	0.1	1.6
5.1 kΩ 电阻	1	0.1	0.1
10 kΩ 电阻	4	0.1	0.4
20 kΩ 电阻	1	0.1	0.1
50 kΩ 电位器	2	0.5	1.0
50 kΩ 双联电位器	1	1.5	1.5
1 μF 电容	1	0.1	0.1
CD4052	1	1.0	1.0
LF347	2	1.5	3.0
74LS04	1	1.8	1.8
uA741	1	0.8	0.8
电源	1	20	20
显示	1	20	20
总计			50.8

故所设计的电路的造价成本为 52.95 元。