
项目选题：**关于浏览器小恐龙游戏的实现**

课程名称：**数字逻辑设计**

姓　　名：**叶之凡、胡若凡**

学　　号：**3200104787、3200102312**

# 1 综述

## 1.1 背景介绍

生活中，我们时常会碰到网络卡断或延迟的问题，为了解决在这段时间中的疲乏，浏览器为我们提供了一些小游戏，而其中最有名的便是谷歌浏览器的小恐龙冒险。

这个游戏的原理是，在一场黑白地图之中，具有随机出现的背景白云，障碍物仙人掌与障碍物飞禽，玩家将会控制一只小恐龙进行闯关冒险。其中遇到仙人掌需按空格键跳过，遇到飞禽需要保证自己的位置在其之下。计数板将会记录下小恐龙当前冒险距离和最远冒险距离。

对设计来说，地面与白云的地图保持循环，仙人掌与飞禽随机出现，而小恐龙具有行走状态，跳跃状态与死亡状态，计分板需实现比较，变化与记录功能。

## 1.2 设计说明

**A. 设计开发环境**

实验平台：Xilinx kintex7

开发环境：Xilinx ISE

硬件描述语言：Verilog HDL

本次设计所用的硬件编程软件为 Xilinx 公司设计的 ISE 14.7 软件，所使用的语言为 Verilog 语言。该软件的使用和 Verilog 语言的学习都较为简单，和 c 语言有相似之处。同方式或混合方式对设计建模。这些方式包括：行为描述方式建模；数据流方式建模；结构化方式建模等。Verilog HDL 中有两类数据类型：线网数据类型和寄存器数据类型。线网类型表示构件间的物理连线，而寄存器类型表示抽象的数据存储元件。通过模块化的设计与相应的外部接口，可以实现较好的人机交互。

**B. 核心模块设计**

程序主要分为以下 6 个模块，下面对其进行简单列举。

显示驱动（vga.c）

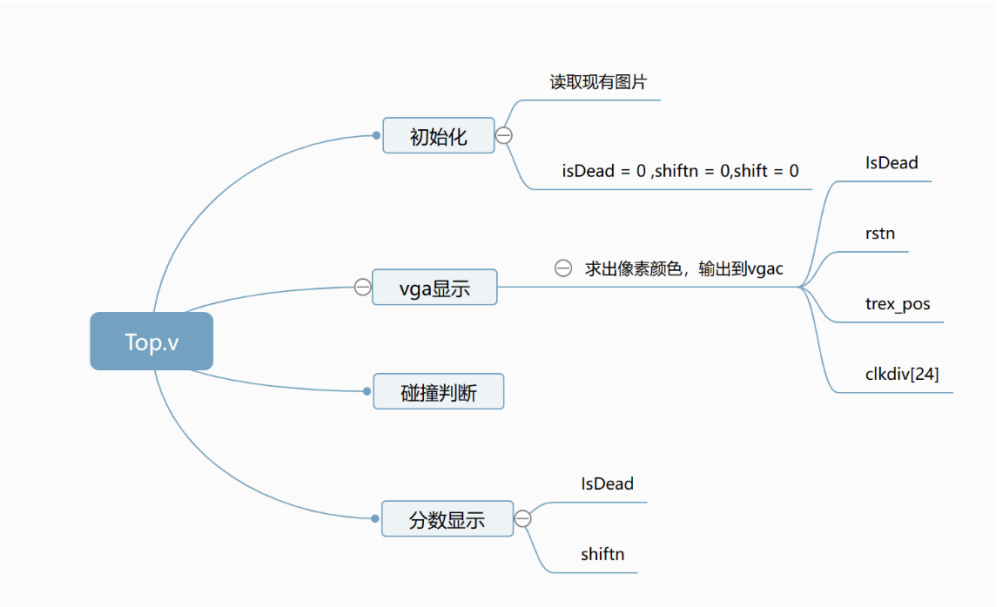
图像输入

游戏主逻辑（top.v）

跳跃逻辑（jump.v）

计分板模块 （score.v）

七段数码管显示（HexTo8SEG.v）




C. **辅助工具**

随机地图生成工具：matlab

主要任务：由于我们需要将图片输入到 vga 中，以 01 信号形式输出，因此编写恐龙(迈左脚和迈右脚)、背景、障碍、结束界面四个图，输出二值图像的 0/1 矩阵。

主要流程：用 matlab 自带函数 unidrnd，产生一组离散均匀随机整数。在整数和元素类型间产生一一映射的

对应关系。以此将图片写入 31500x480 的地图。

## 1.3 游戏说明

### 1.3.1 游戏输入

玩家通过按钮和开关的输入控制游戏的进行：

btn[0](W16_V18)：控制小恐龙的起跳；

btn[1](W17_V19)：在 gameover 之后重新开始游戏；

sw(AA10)：控制游戏无敌模式的开启

### 1.3.2 游戏输出

VGA 游戏界面、用七段数码管输出当前得分和最高得分。

# 2 核心设计模块

## 2.1 显示驱动 （vga.c）

### 2.1.1 需求分析

将主模块的 640*480 的图像信息转化为 vga 信号，并输出在开发板的 vga 显示屏幕上

### 2.1.2 接口参数

输入参数

vgaclk --------25MHz 时钟信号

d_in ----------要输出的像素点的 RGB12 位颜色值

clrn ----------清屏信号，负逻辑，为 0 则本次不输出

输出参数

row_addr ------8 位行坐标（0-479）

col_addr ------9 位列坐标（0-639）

read ----------可读信号，为 1 则当前时间输出在可显示区域

r，g，b-------3 个 4 位颜色输出

hs ------------水平同步信号

vs ------------垂直同步信号

### 2.1.3 实现原理

#### 2.1.3.1 vga 信号规范

VGA 是 IBM 在 1987 年随推出的一种视频传输标准，具有分辨率高、显示速率快、颜色丰富等优点，彩色显示器领域得到了广泛的应用。

| 信号线 | 定义 |
| --- | --- |
| HS | 列同步信号（3.3V 电平） |
| VS | 行同步信号（3.3V 电平） |
| R | 红基色（0~0.714V 模拟信号） |
| G | 绿基色（0~0.714V 模拟信号） |
| B | 蓝基色（0~0.714V 模拟信号） |

我们大作业使用的屏幕，刷新频率为 60Hz，所以我们需要使用 800525*60 = 25.2MHz 的驱动信号，在这里直接使用 25MHz。

在实验中，我们主要需要在一个 VS-HS 完成一个周期的时间内，按从左上角到右下角的顺序输出各个像素点的颜色，就能在显示器上显示出对应的图案。需要注意的是，VGA 颜色的输出必须严格遵循时序，只能在 cd 段输出非 0 数据，否则会对显示造成干扰。

VGA 显示依赖于行扫描信号、场扫描信号和当前像素的 RGB 值。电子枪从左上角开始从左向右扫描，一行扫描完后，进行行消隐（使电子束不再发射，同时将其回归到下一行的最左边，此时行扫描信号为 0），接着扫描下一行，当所有行扫描完后，进行场消隐（电子束回到第一行的最左边，此时场扫描信号为 0），在每次指定的时钟的上升沿，输出场扫描信号、行扫描信号和对应的 RGB 值。

#### 2.1.3.2 驱动模块具体实现

- 在每一个 vgaclk 的正边沿，水平计数+1，水平计数达到 799 后下一次为 0：

  当水平计数为 0~95 时，设定同步信号为 0，为同步期，产生行同步脉冲；

  96~152 为消隐后期，同步信号为 1，但是不能输出数据信号

  153~792 位显示期，共 640 个时钟周期，输出当前行数据信号

  792~799 为消隐前期，同步信号为 1，不输出数据信号，等待下一个同步脉冲
- 当水平计数每次到 0 时，说明当前行已经输出完成，垂直计数+1，垂直计数达到 524 后下一次为 0：

  当垂直计数为 0~1 时，设定同步信号为 0，为同步期，产生场同步脉冲；

  2~42 为消隐后期，同步信号为 1，但是不能输出数据信号

  43~522 位显示期，共 480 个，输出当前帧数据信号，在这里输出每一行

  523~524 为消隐前期，同步信号为 1，不输出数据信号，等待下一个同步脉冲

这里使用的数据与理想值尽管存在一定的偏差，但是确是我们实验中得到的， 因此我们可以认为这是符合实际的值。

## 2.3 游戏主逻辑 （Top.v）

### 2.3.1 需求分析

利用各个之前实现的模块，实现出小恐龙不断奔跑，在跳跃按钮按下时跳起，在撞到障碍时死亡的效果。同时实现开关控制的无敌模式、按钮控制的重新开始游戏等功能。

### 2.3.2 设计原理

调用 trex_pos 模块根据按钮和小恐龙当前的高度计算小恐龙下一个时钟周期的高度。通过分图层显示的方式实现不同类型元素的分别显示。通过在背景和障碍图层像素矩阵中的坐标偏移量实现背景和障碍的左移，从而制造出小恐龙不断向右边奔跑的效果。通过判断小恐龙的图像和障碍的图像是否重叠来判断小恐龙是否撞到了障碍，从而让小恐龙在撞到障还是死亡。

### 2.3.3 具体说明

#### 2.3.3.1 与 vgac 模块的交互

vgac 模块会以一定频率向主模块给出像素的横纵坐标，对于给出的横纵坐标，主模块需要根据该像素在各个图层中的情况来为其赋值。具体来说，如果其在某个图层中是黑色的，则该像素为黑色。仅当在所有图层中它都为白色时，该像素才为白色。

#### 2.3.3.2 与 trex_pos 模块的交互

将经过防抖处理后的跳跃按钮传给 trex_pos 模块，trex_pos 模块根据小恐龙当前离开地面的高度以及跳跃按钮是否被按下来决定小恐龙接下来的高度。当小恐龙位于地面时，按下跳跃按钮可以让其开始跳跃，当小恐龙离开地面时，按下跳跃按钮对其没有影响。

#### 2.3.3.3 图层显示

vga 的显示分为四个图层——背景图层、障碍图层、恐龙图层、文字图层。

**背景图层：**包含地平线、云朵等游戏中在小恐龙经过时不会导致其死亡的物体。大小为 640 * 480，被存储在 qwq.mif 文件中，被程序读取到 map_pixel 数组中，在小恐龙行进时循环播放。

**障碍图层：**障碍图层包含仙人掌、飞鸟等游戏中在小恐龙经过时会导致其死亡的障碍物。大小为 32000 * 480，被存储在 mapa.mif 和 mapb.mif 中，被程序读取到 mapa 和 mapb 数组中，在小恐龙行进时循环播放。由于障碍图层的宽度是屏幕的 50 倍，因此用户无法察觉到障碍的循环。

为了让游戏中的障碍飞鸟可以上下扇动翅膀，mapa 和 mapb 中的飞鸟并不相同。mapa 中翅膀朝上的鸟，在 mapb 中翅膀朝下，反之亦然。在短时间内轮流显示 mapa 和 mapb 中的飞鸟，就实现了鸟扇动翅膀的效果。

**恐龙图层：**恐龙图层包含游戏的主要角色小恐龙。大小为 80 * 80，被存储在 T_LEFT, T_RIGHT, T_DOWN, T_DEAD.mif 中，被程序读取到 trex_left，trex_left，trex_right，trex_dead 中。恐龙图层被显示在地图上特定的一个 80 * 80 像素的区域内。根据小恐龙所处的不同状态，恐龙图层显示的恐龙图样有细微的差别。在小恐龙在地面行进时，小恐龙
左脚着地和右脚着地的图像轮流显示，以体现其动态。在小恐龙跳跃在空中时，显示其两脚跳跃的图像。在小恐龙撞到障碍死亡时，显示其瞳孔放大的死亡图像。

**文字图层：**文字图层包含游戏结束的 "GAME OVER" 字样。大小为 640 * 480，被存储在 game_over.mif 中，被程序读取到 game_over 数组中，当小恐龙撞到障碍死亡时，在显示屏的中上方显示 "GAME OVER" 字样。

#### 2.3.3.4 地图移动

为了显示出小恐龙向前跑的效果，我们让障碍图层和背景图层向后移动。由于这两个图层宽度并不相同，而都要循环播放，因此在这两个图层中使用不同的偏移变量。在障碍图层中使用 shiftn，并对 32000 取模，在背景图层中使用 shift，对 640 取模。

#### 2.3.3.5 碰撞判断

碰撞判断和显示同时进行，在 vgac 向主模块提供的像素坐标位置，若恐龙图层中该位置为黑色（表明

该位置属于小恐龙身体）且障碍图层中该位置也为黑色（表明该位置属于障碍），则说明小恐龙与障碍发生了碰撞，将 isDead 赋值为 1。

## 2.4 跳跃逻辑（jump.v）

### 2.4.1 需求分析

在跳跃按钮按下且小恐龙为起跳时时使小恐龙跳起，且在空中变换速度使小恐龙跳跃效果逼真。

### 2.4.2 设计原理

利用分段函数实现速度分级，模拟重力加速度。

### 2.4.3 具体实现

输入：clk 时钟信号 jump_btn 跳跃按钮 speed 当前速度

输出：height 小恐龙目前高度

## 2.5 计分板模块（score.v）

### 2.5.1 需求分析

需要在七段数码管上显示两个分数：当前得分与历史最高分。当前的得分随着小恐龙奔跑更新，历史最高分也需要随之更新。当小恐龙碰撞到物理死亡时停止计分，并在下一次重新开始游戏时将分数清零。

### 2.5.2 设计原理

使用一个四位十进制计数器来作为计分板，用四个四位寄存器组作为最高分记录。利用分频时钟作为触发计数的信号；利用周期更短的分频时钟作为更新最高分的信号，这样就能避免更新分数过程的延迟。四位十进制计数器由四个一位十进制计数器串联而成。每个一位计数器计数至 9 时将产生进位信号，并作为下一位的计数使能信号。将小恐龙的存活状态作为最低位一位计数器的使能信号，可以实现小恐龙死亡停止计分的功能。将重置游戏信号作为计数器 reset 输入可以实现重置游戏分数清零的功能。

### 2.5.3 具体实现

首先设计一位计数器模块。当遇到时钟信号上升沿时触发计数：如果已经计数至 9 则归零，如果已经计数至 8 则产生进位信号并计数至 9。当遇到重置信号下降沿（即重置按钮被按下）时异步重置计数器。

然后设计计分模块，实例化四个一位计数器模块，并选择合适的时钟信号（我们选择 clkdiv[22] ）来作为触发计数信号。对于每一个计数器的计数使能信号，第一个计数器为小恐龙的存活状态，其余计数器为比它小的位数的进位信号以及小恐龙存活状态的与。最高分设计为当遇到一个比计数时钟周期更小的时钟上升沿时更新一次最高分。

## 2.6 主板七段管显示（HexTo8SEG.v）

主要过程和课程中要求一致，并无过多变化

# 3 心得体会

这次实验的心得体会我们主要想从两个方面展开，即收获成长与不足之处。

首先，是不足之处。这次的大实验时间确实比较仓促，在经历了三个礼拜的线上实验后，我们缺少足够的讨论与准备时间，平时的 lab 也依然在紧张的准备之中，因此整个过程我们只能边学边做，因此整个实验都是在与学长姐的讨论与求教之中一点点做出来的，我们参考了老师给我们的许多实验报告，研究了他们的核心代码，并且在两个 18 级的学长的帮助下，确立了我们需要做的这个浏览器上的小恐龙游戏。可以说，这个过程里我们更多的像是一个被帮助者，而缺乏了一些主动创造性，感觉这其中有一些可惜，因为我们的整个工程之中，他人对我们的帮助真的对我们十分的重要。

但是总的来说，我们也有十分大的成长。首先，相比于平时的 lab，这次的工程里我们第一次完全实

现了自主构思，我们思考了游戏中需要的环节，并且设计流程、存储变量以及算法，这让我们对于如何设计一个完整的大程序有了更多的了解；其次，我们在这个过程中详细复习了在 13 个 lab 模块中使用过的各个工程，特别是 led 灯，这让我们对这个学期中的实验内容有了更加深入的了解；最后，是对这种语言，由于这次进行了代码的编写，我们既熟悉了这种编程语言的相关内容，也加深了对引脚的理解。

最后，进行一个总结。在这次的实验之中，我们既经历了许多困难，也得到了十分多的进步，但是整个工程之中还有许多能够改动的地方。比如说，随着游戏时间增长，小恐龙的速度参数应该发生变化，但在时钟的调整上，时钟得不到较好的结果；再比如说，我们可以做出一张白天的图，本来我们还希望设置一张不一样的背景，然后使用一个按钮进行地图的切换，但是事实上是，我们碰到了一些困难，出现了一些暂时我们还没法解决的问题；还有一个较需要解决的，就是在游戏开启的时候，设置一个开始界面。由于时间已经逼近考试周，因此我们最后只能作罢。

但是我们相信，在未来的学习过程中，我们一定能发挥的更好，学到更多的知识，带着在数字逻辑设计课程中打好的基础，在计算机组成中面对接下来的挑战。

# 4 成员介绍

**胡若凡：竺可桢学院混合 2002 班，学号 3200102312**
**叶之凡：竺可桢学院混合 2005 班，学号 3200104787**