Lab12实验报告

小组组员

陈子元 211240069

林凡琪 211240042

徐研 211240047

班级 2班

实验时间 2023/02/23

一、实验目的

本实验的目标是在Nexys A7-100T开发板的FPGA上实现一个简单的计算机系统，能够运行简单的指令，并处理一定量的输入输出。在所有功能开发完毕后，希望能够完成基本的terminal 功能，即键盘输入命令，并在显示器上输出结果。

二、实验原理（知识背景）

本实验需要实现的指令包括算术运算、逻辑运算、转移指令等常见指令。为了实现这些指令，需要对计算机系统的结构、指令执行流程、寄存器等相关知识有一定的了解。同时，需要掌握Verilog语言的基础知识并熟练掌握FPGA开发环境的使用。

在实验过程中，需要使用Nexys A7-100T开发板进行开发和测试。在进行设计时，需要充分考虑系统的可扩展性、稳定性和可靠性，同时需要进行充分的测试和调试，确保实验能够正常运行。

三、实验环境/器材

Verilog 2022.1

Windows 10

Nexys开发板

显示器

键盘

四、功能实现

4.1 硬件部分

- 支持键盘、显示器、七段数码管多设备的调度处理
- 实现了RV32I架构的单周期CPU，在一个CPU周期内进行一条指令的取指、译码、计算和回写。
- 实现了键盘、显示器等的内存映射I/O。
- 实现了终端界面（光标、字符等）的绘制

4.2 软件部分

- 实现了一个简易的操作系统
  - 接收键盘输入并解析用户命令
- 4个用户程序
  - 打入hello，显示Hello World!
  - 打入time，显示时间
  - 打入fib n，计算斐波那契数列并显示结果
  - 打入未知命令，输出Unknown Command。

五、组内分工

陈子元（211240069）：软件部分和调试

林凡琪（211240042）：硬件部分（ cpu指令等设计、内存映射、七段数码管、软硬件接口）

徐研（211240047）：硬件部分（ cpu指令等设计、内存映射、VGA显示、键盘）

（按首字母排列）

六、硬件部分

6.1 顶层模块调度

- 生成所需要的所有时钟

- RAM读写频率：50MHZ
- VGA所需时钟：25MHZ
- CPU运行的时钟频率：1MHZ

- 将时钟以及所需要的数据传入对应的处理模块，并在多模块之间完成信号交互与信号处理。

```
// 时钟生成
clkgen #(25000000) my_clk(clk_50, SW[0], 1'b1, VGA_CLK); // 25HZ
VGA_CLK
clkgen #(1000000) my_clk2(clk_50, SW[0], 1'b1, CPU_CLK); // CPU_CLK
clkgen #(10000) my_clk3(clk_50, SW[0], 1'b1, clk_i2c); // i2c_clk
// CPU模块
cpu my_cpu
(SW[2], CPU_CLK, clk_50, debug_addr, debug_out, now, iodata_r,
mem_wren, io_wdata, mem_addr, HEX0, HEX1, HEX2, HEX3, SW[9:5], bg_color,
ty_color, LEDR, en_piano, io_music);
// io处理模块
io io_part
(clk_50, SW[0], mem_wren, mem_addr, io_wdata, iodata_r,
VGA_BLANK_N, VGA_B, VGA_CLK, VGA_G, VGA_HS,VGA_R, VGA_SYNC_N, VGA_VS,
PS2_CLK, PS2_DAT, HEX4, HEX5, SW[1], bg_color, ty_color, AUD_ADCDAT,
AUD_ADCLRCK, AUD_BCLK, AUD_DACDAT, AUD_DACLRCK, AUD_XCK, FPGA_I2C_SCLK,
FPGA_I2C_SDAT, clk_i2c, en_piano, io_music);
```

6.2 CPU

在本实验中，我们实现了单周期CPU，即在一个指令周期内，完成取指、译码、执行、访存、回写五个阶段。

根据Lab11，已经成功实现了RV32I架构中的整数运算指令、控制转移指令和存储器访问指令。这些指令是计算机运行过程中至关重要的组成部分。但是，Lab11中的内存板块由OJ平台给出，所以在本实验中我们还需要深入研究这些指令的细节，来写一个自己的内存板块。总之，虽然我们的实验已经有了一些基础，但仍然有很重要的工作要做，所以关于Lab11的取指、译码、ALU和回写等部分就不在此赘述。

结合实现在lab10中的数据通路，我们可以看到，CPU将地址信息发送到内存中，并且在内存中寻址。这个过程中，不仅包括将地址发送到内存中，还包括内存如何响应CPU的请求以及如何返回所需的数据。根据这个过程，我们成功地完成寻址取值。

```
assign memin = (memop[1:0]==2'b00)?{4{datain[7:0]}}:((memop[1:0]==2'b10)?datain:{2{datain[15:0]}}) ; //lb: same for all four, lh:copy

//four memory chips
testdmem mymem(.byteena_a(wmask),.data(memin), .rdaddress(addr[16:2]), .rdclock(rdclk), .wraddress(addr[16:2]), .wrclock(wrclk), .wren
//wmask,addr[16:2]
assign wordout = (addr[1]==1'b1)? dwordout[31:16]:dwordout[15:0];

assign byteout = (addr[1]==1'b1)? ((addr[0]==1'b1)? dwordout[31:24]:dwordout[23:16]):((addr[0]==1'b1)? dwordout[15:8]:dwordout[7:0]);
```

6.3 内存映射

根据和CPU的约定，内存分配如下：

ADDRESS	FUNCTION
0x00000000 - 0x000fffff	指令存储器
0x00100000 - 0x001fffff	数据存储器（常量、全局变量、栈及栈空间）
0x00200000 - 0x002fffff	显示器（VGA）
0x00300000 - 0x003fffff	键盘

6.4 CPU外设交汇

6.4.1 IO设备

内存中0x00300000处，注册了一个32位的空间，存放键盘的输出ASCII码；
在内存中的0x00200000～0x00200834中，注册了长度为30*70的空间，用于存储显存中字符。

- 键盘
  - 键盘复用了实验七的模块，支持组合键、大小写等功能，因没有做修改，在此不做介绍。
- 显示器
  - 显示器复用了实验八的模块，稍作修改，实现了适用于实验十二的VGA逻辑。

- 具体实现如下：

```
always @(posedge pclk) // deal with h direction
    begin
        if(h_valid == 1'b0)
        begin h_char <= 6'b0; h_font <= 4'b0; end
        else
        begin
            if(h_font >= 4'd8)
            begin
                h_char <= h_char + 6'd1;
                h_font <= 4'd0;
            end
            else
            begin
                h_font <= h_font + 4'd1;
            end
        end
    end
```

```
always @(posedge reset or posedge pclk)
        if (reset == 1'b1)
            x_cnt <= 1;
        else
        begin
            if (x_cnt == h_total)
                x_cnt <= 1;
            else
                x_cnt <= x_cnt + 10'd1;
        end

    always @(posedge pclk)
        if (reset == 1'b1)
            y_cnt <= 1;
        else begin
            if (y_cnt == v_total & x_cnt == h_total)
                y_cnt <= 1;
            else if (x_cnt == h_total)
                y_cnt <= y_cnt + 10'd1;
        end
```

6.4.2 其他

设备	功能
SW[0]	显示器的使能端
SW[1]	显示器的reset控制端口
七段数码管	显示按键键值和ASCII码

七、软件部分

7.1 简介

软件部分，我们实现了一个简单的操作系统，包含以下功能：

- 支持四种命令
  - hello：输出Hello world！
  - fib n：打印斐波那契数列的前n项和
  - time：打印系统开启时长（秒数）
  - 未知命令：Unknown Command

7.2 实现

- 底层函数
  - 实现框架中的putstr（）
- hello命令
  - 直接putstr(“Hello world!”)

- time命令
  - 从硬件中用clk计算时长，存在寄存器中
  - 在软件部分用内联汇编取出这个值，输出
- fib命令
  - 用for循环递推计算
  - 通过putstr输出

```
int fib(int n){
    int f1 = 1, f2 = 1;
    for (int i = 2; i <= n + 2; i++)
    {
        int tmp = f1;
        f1 = f1 + f2;
        f2 = f1;
    }
    return f2 - 1;
}
```

- 未知命令
  - 直接putstr(“Unknown Command”)
- putch实现

```
void putch(char ch) {
  if(ch==8) //backspace
  {
      //TODO
      vga_start[(vga_line<<7)+vga_ch-1] = 0;
      vga_ch--;
      if (vga_ch < 0){
 vga_line--;
 vga_ch = VGA_MAXCOL-1;
      }
      return;
  }
  if(ch==10) //enter
  {
      //TODO
      vga_line++;
      vga_ch = 0;
      return;
  }
  vga_start[ (vga_line<<7)+vga_ch] = ch;
  vga_ch++;
  if(vga_ch>=VGA_MAXCOL)
  {
     vga_line++;
     vga_ch = 0;
  }
  return;
}
```

八、实验结果

实现了基础实验要求，能正确处理四种不同指令。

九、实验中遇到的问题及解决办法

- 刚开始对于内存映射的认识很模糊，后来经过学习终于艰难地完成了这一部分。
- 因为对Verilog的理解不是很深刻，总是出现时序混乱，比如说，打了按键之后VGA没有出现任何显示。三个队友一起debug就把错误给修正了。

十、实验得到的启示

通过这次实验，结合计算机系统基础课程PA部分所学知识，我们更加了解了CPU的工作原理，以及与IO间的调度处理。

verilog虽然看着和c语言有很多相像之处，但是它作为一门硬件语言，绝对是不能够用理解c语言的方式去理解它的，否则就会出现很多难以理解的错误