四川大学计算机学院、软件学院

实验报告

学号：2016141482154 姓名：何长鸿 专业：计算机科学与技术 班级：163040114 第14 周

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 课程名称 | 计算机组成与设计课程设计 | | 实验课时 | 2 |
| 实验项目 | CPU设计-数据通路的设计与实现 | | 实验时间 | 2018.6.8 |
| 实验目的 | 1. 复习寄存器传输控制技术  2. 掌握CPU的核心组成：数据通路组成与原理  3. 设计数据通路 | | | |
| 实验环境 | Windows 7，ISE软件，SWORD开发板 | | | |
| 实验内容（算法、程序、步骤和方法） | **1.实验流程**  1. 调入已提供的工程  2. 输入与完善相关模块  3. 代码综合  4. 硬件约束与实现  5. 生成流代码与下载  **2.实验原理**  完整的实验SOC逻辑图5.1所示  图5.1  本实验将调用以下10个模块：   * + U1： CPU -SCPU   + U2： ROM -ROM\_D   + U3： RAM -RAM\_B   + U4： 总线(含外设3~4) -MIO\_BUS   + U5： 七段显示接口 -Multi\_8CH32   + U6： 外设1- 七段显示设备 -SSeg7\_Dev   + U7： 外设2- GPIO接口及LED -PIO   + U8： 辅助模块一，通用分频模块 -clk\_div   + U9： 辅助模块二，机械去抖模块 -Anti\_jitter   + U10： 通用计数器 -Counter\_x   其中，CPU模块的内部结构如图5.2所示。  图5.2 CPU内部结构  数据通路DataPath模块的内部基本结构如图5.3所示。  图5.3 数据通路基本结构  数据通路主要包括：ALU，寄存器堆，指令计数器PC，多选器，加法器等部件。其中ALU可实现add、sub、and、or、nor、slt等运算功能，其结构如图5.4所示。  图5.4 ALU内部结构  本次实验其余模块均已完成，只需完成数据通路模块的设计和实现。  **3. 实验步骤**  打开已经提供的工程（OExp05-DataPath），可以看到逻辑图中缺少SCPU模块。因此，进行SCPU模块的设计。   1. **设计DataPath模块**   添加Source，选择and32.v，or32.v，nor32.v，srl32.v， xor32.v，or\_bit32.v，addc32.v，ABC32.sch进行添加，保存。再在 MUX文件夹下选择MUX8T1\_32.v进行添加，保存。  然后在工程目录下选择alu.sch进行添加，保存。从而构造好ALU模块。  接下来，从common文件夹下选出add\_32.v、Ext\_imm16.v，从CPU文件夹下选出Regs.v、REG32.v，从MUX文件夹下选出MUX2T1\_32.v、MUX2T1\_5.v进行添加、保存。后在创建SDPath.sch模块。  在Symbols中依次选择alu，Regs，add\_32，Ext\_32，REG32，MUX2T1\_32和 MUX2T1\_5等符号拖入画布中对应位置，修改其实例名，然后参看文档DataPath.pdf，做好线的连接，保存。再选中SDPath模块，进行检测，正确无误后生成对应的symbol符号SDPath.sym。  **2. 构造SCPU模块**  切换到Design视图，双击SCPU模块，可以看到逻辑图中缺少SDPath模块，参看文档SCPU.pdf，将SDPath添加到该逻辑图中，从而构造好SCPU模块。  **3. 完善顶层逻辑图**  回到顶层模块OExp05\_DataPath，参看文档DataPath\_Top.pdf，将SCPU模块添加到顶层模块中，完善整个逻辑图。  **4. 最终实现**  将整个工程进行保存，综合，实现。如果调试通过，双击Generate Programming File生成比特流文件。下载比特流文件，观察SWORD开发板运行结果。 | | | |
| 数据记录和计算 | 1**.测试开关设置**   * 图形功能测试   按表一测试验证。  **表一**   |  |  |  | | --- | --- | --- | | **开关** | **位置** | **功能** | | DSW[0] | 0 | 七段码图形显示 | | DSW[2] | 0 | CPU全速时钟 | | DSW[4:3] | 00 | 7段码从上至下亮点循环移动:跑马灯 | | 11 | 7段码图形变幻 | | DSW[7:5] | 000 | 作为外设使用 |   **测试结果：**  **功能1：** 当开关DSW[4:3]是00时，7段码从上至下亮点循环移动:跑马灯。  **实验现象1：**如下图，由1.1->1.2->1.3->….的变化，闪光点从右至左，从下至上流动跑。说明实现了跑马灯的功能,    **图1.1 图1.2**  **图1.3**  **功能2：**当开关DSW[4:3]是11时，7段码图形变幻  **实验现象2:**七段码管上的图形不断变化，其中就出现了图2.1，2.2，2.3这样的图形。说明实现了七段码管图形变换的功能    **图2.1 图2.2**  **图2.3**   * 文本功能测试   按表二测试验证:  **表二**   |  |  |  | | --- | --- | --- | | **开关** | **位置** | **功能** | | DSW[0] | 1 | 七段码文本显示 | | DSW[2] | 0 | CPU全速时钟 | | DSW[4:3] | 01 | 7段码显示内存数据:0～F | | 10 | 7段码显示累加 | | DSW[7:5] | 000 | 作为外设使用 |   **测试结果：**  **功能3：**当DSW[4:3]是01时，7段码显示内存数据:0～F。  **实验结果3：**如下图实现了从0~F的内存显示                    **功能4：**当DSW[4:3]为10时，7段码显示累加  **实验现象4：**实现了从图4.1->4.2->4.3->….的递增  图4.1 图4.2  图4.3  **2.仅定性观测**  按表三进行定性观测:  **表三**   |  |  |  | | --- | --- | --- | | **开关** | **位置** | **功能** | | DSW[0] | 1 | 七段码文本显示 | | DSW[2] | 1 | CPU单步时钟 | | DSW[7:5] | 001 | CPU指令字地址 PC\_out[31:2] | | 010 | Counter值输出 | | 011 | ROM指令输出Inst\_in | | 100 | CPU数据存储地址addr\_bus(ALU输出) | | 101 | CPU数据输出Cpu\_data2bus(寄存器B) | | 110 | CPU数据输入Cpu\_data4bus(RAM输出) | | 111 | CPU指令字节地址PC\_out |   **测试结果：**  功能5：当DSW[7:5]从001变化到111，功能如上表变化。  实验结果5：对应功能结果如下 | | | |
| 结论  （结果） | 在搭建完SCPU核心部件数据通路之后，实现了单周期SCPU，再加入到之前的SOC，使SOC能实现与上一次实验相同的结果。 | | | |
| 小结 | 通过实现单周期的数据通路，更理解了数据通路的设计思路以及数据通路所实现的不同功能。实验过程中，在给总线分支“4”赋值的过程中出现了问题，后来通过先在总线分支上侧添加文本，再给总线分支赋值的方式，成功给总线分支赋上期望的值。另外，还遇到了无法对SOC中加入SCPU和SDPath的IP核的情况，通过分析，我们认为虽然我们正常生成了SOC和SDPath的sym文件，但在覆盖原有的同名sym文件时出现了问题，所以我们通过在最初导入SOC的时候就向SOC的sch文件中添加SCPU模块后面在修改了SCPU模块之后，通过更新SOC中的SCPU的方式，解决了问题。 | | | |
| 指导老师  评议 |  | | | |
| 成绩评定： | 指导教师签名： | | |