**实验二：基本单周期CPU设计实验报告**

杨健邦 515030910223

**一、实验目的：**

1. 理解计算机 5 大组成部分的协调工作原理，理解存储程序自动执行的原理。
2. 掌握运算器、存储器、控制器的设计和实现原理。重点掌握控制器设计原理和实现方法。
3. 掌握 I/O 端口的设计方法，理解 I/O 地址空间的设计方法。
4. 会通过设计 I/O 端口与外部设备进行信息交互。

**二、实验内容：**

1. 采用Verilog HDL在quartusⅡ中实现基本的具有20条MIPS指令的单周期CPU设计。
2. 利用实验提供的标准测试程序代码，完成仿真测试。
3. 采用I/O统一编址方式，即将输入输出的I/O地址空间，作为数据存取空间的一部分，实现CPU与外部设备的输入输出端口设计。实验中可采用高端地址。
4. 利用设计的I/O端口，通过lw指令，输入DE2实验板上的按键等输入设备信息。即将外部设备状态，读到CPU内部寄存器。
5. 利用设计的I/O端口，通过sw指令，输出对DE2实验板上的LED灯等输出设备的控制信号（或数据信息）。即将对外部设备的控制数据，从CPU内部的寄存器，写入到外部设备的相应控制寄存器（或可直接连接至外部设备的控制输入信号）。
6. 利用自己编写的程序代码，在自己设计的CPU上，实现对板载输入开关或按键的状态输入，并将判别或处理结果，利用板载LED灯或7段LED数码管显示出来。
7. 例如，将一路4bit二进制输入与另一路4bit二进制输入相加，利用两组分别2个LED数码管以10进制形式显示“被加数”和“加数”，另外一组LED数码管以10进制形式显示“和”等。（具体任务形式不做严格规定，同学可自由创意）。
8. 在实现MIPS基本20条指令的基础上，实现Y86相应的基本指令。
9. 在实验报告中，汇报自己的设计思想和方法；并以汇编语言的形式，提供以上两种指令集（MIPS和Y86）的应用功能的程序设计代码，并提供程序主要流程图。

**三、预习内容：**

1. 实验前仔细阅读Altera-DE1-SOC User Manual及相关用户应用数据手册，学习并掌握其板载相关资源的工作原理、连接方式、和应用注意事项。
2. 根据课程所讲单周期CPU设计原理，提前设计并仿真实现相关设计代码。

**四、实验器材：**

1. Altera-DE1-SOC实验板套件1套
2. 万用表 1台
3. 示波器 1台

**五、设计思想和方法：**

1. Control Unit的实现，先填写完成真值表，通过真值表来实现control unit模块，真值表见图一。



图一：真值表

1. 填写alu模块：根据注释的提示填写即可。
2. 时钟的生成：需要提供同步的两个时钟，其中有一个时钟的频率是另一个时钟频率的两倍。基本的解决方法是，传入一个由FPGA板子AF14引脚生成的时钟，通过这个时钟再生成一个频率为其一半的时钟（通过一个寄存器保存，每次在传入时钟上升沿的时候取反），这样就可以保证两个时钟是同步的了。

module clock\_gen(origin,cpu\_clk,mem\_clk);

input origin;

output reg cpu\_clk;

output mem\_clk;

assign mem\_clk = origin;

always @ (posedge origin)

cpu\_clk <= ~cpu\_clk;

initial cpu\_clk <= 0;

endmodule

1. IO设计：板上的10个开关分成2组，每组都作为一个输入端口。6个数码管分成3组，每组都作为一个输出端口，显示的是十进制的数字，由于数量的限制，只能显示十位和个位的数字，通过一个num2sevseg模块来转换。
2. 汇编代码的设计：有一个永真循环，从两个输入端口读入数据，再把两个输入加起来，三个输出端口分别显示两个加数和结果。最高只能显示31+31=62的数据。

0 : 20020080; % (00) addi $2, $0, 10000000b # address 80h %

1 : 20030084; % (04) addi $3, $0, 10000100b # address 84h %

2 : 20010088; % (08) addi $1, $0, 10001000h # address 88h %

3 : 200400c0; % (0c) addi $4, $0, 11000000b # address c0h %

4 : 200500c4; % (10) addi $5, $0, 11000100b # address c4h %

5 : 8c860000; % (14) loop: lw $6, 0($4) # input data from [c0h] %

6 : 8ca70000; % (18) lw $7, 0($5) # input data from [c4h] %

7 : 00c74020; % (1c) add $6, $7, $8 # $8 = $6 + $7 %

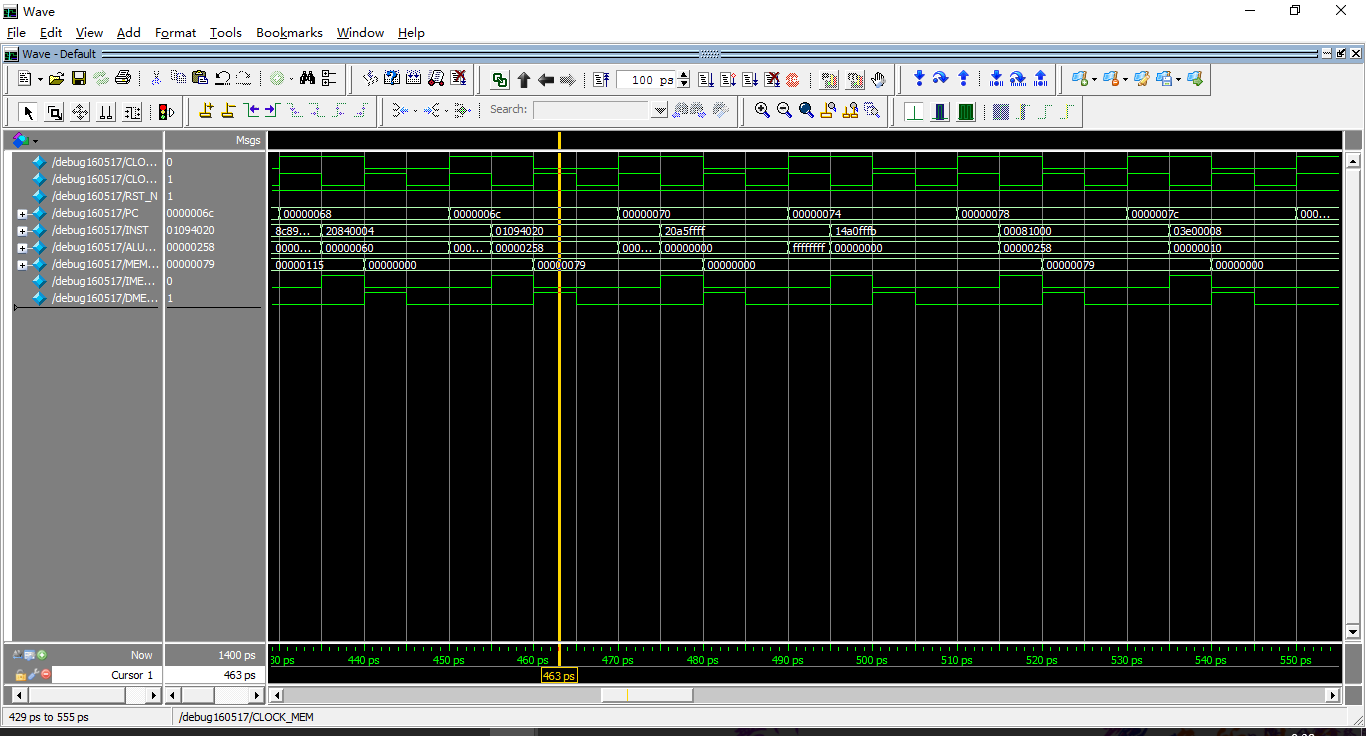
8 : ac460000; % (20) sw $6, 0($2) # output [C0h] to [80h] %

9 : ac670000; % (24) sw $7, 0($3) # output [c4h] to [84h] %

10 : ac280000; % (28) sw $8, 0($1) # output [c0h] + [c4h] to [88h] %

11 : 08000005; % (2c) j loop # loop %

1. 仿真：需要将sc\_computer替换成sc\_computer\_sim，并且还要添altera\_mf.v以及220model.v 两个库才能仿真成功。



**六、心得体会：**

1. 如果发现烧不进去板子，可以尝试添加一个顶层bdf文件，然后把所有引脚分配完，就能烧进板子。
2. 需要读取ram的程序的仿真需要添加altera\_mf.v以及220model.v 两个库。
3. 变量写错不会导致编译失败，需要很认真地检查。