实验大作业：用 Verilog HDL 完成３６条ＭＩＰＳ指令单周期处理器开发

一、 设计说明

1. 完成以下指令集。

a) 先完成教材上介绍到的 以 下 1 0 条指令：

add，addiu，sub，and，or，slt，lw，sw，beq 和 J 指令。不支持溢出。

b) 实验最终完成实验（课设）指导书"3-从 74 页起 MIPS 指令集解释--指导手册.pdf"（简称“指导书”）"中的 36 条指令。

2. 处理器为实验（课设）指导书中的单周期 MIPS 处理器。

二、 设计要求

1. 单周期处理器由 datapath(数据通路)和 controller(控制器)组成。
   a) 数据通路由如下 module 组成：PC(程序计数器)、NPC(NextPC 计算单元)、Register File (寄存器文件、寄存器堆)、ALU(算术逻辑单元)、EXT(扩展单元)、IM(指令存储器)、DM(数据存储器)。
   b) IM：容量为 4KB(32bit×1024 字)。
   c) DM：容量为 4KB(32bit×1024 字)。
2. Figure1 为供你参考的数据通路架构图。
   a) 我们不确保 Figure1 是完全正确的；我们也不确保 Figure1 能够满足上述指令集。
   b) 鼓励你从数据通路的功能合理划分的角度自行设计更好的数据通路架构。
   c) 如果你做了比较大的调整，请务必注意不要与要求 5 矛盾。

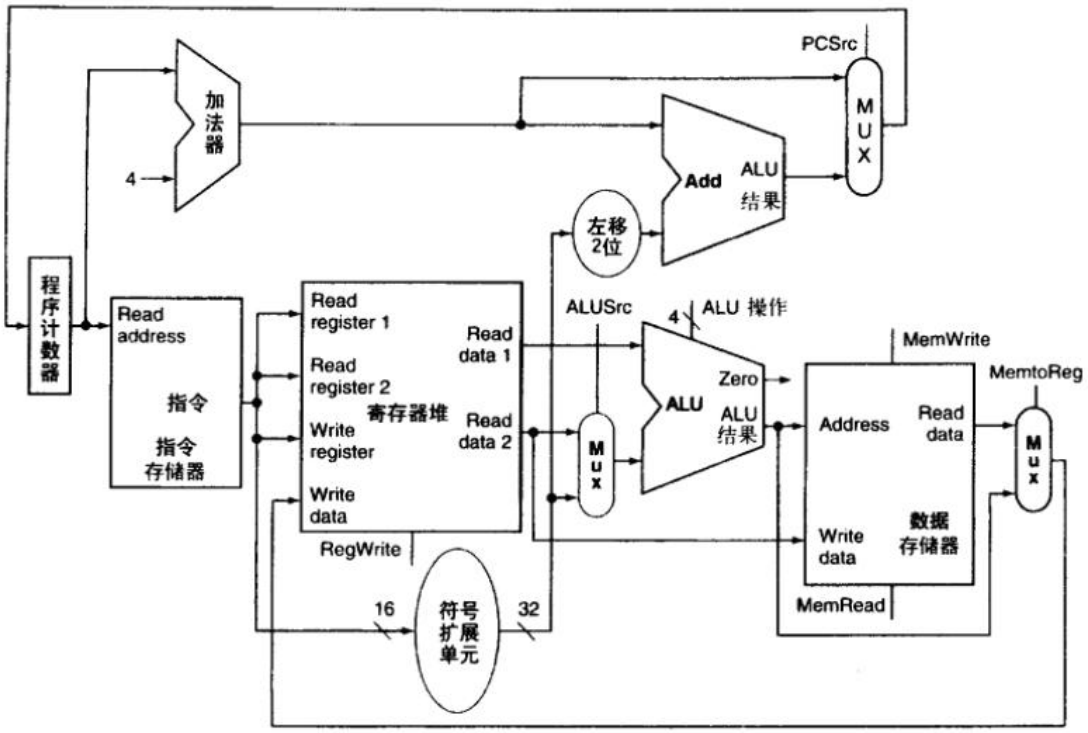

Figure1 数据通路(供参考)

3. 整个 project 必须采用模块化和层次化设计。

   a) Figure2 为参考的目录结构和文件命名。其中红色框的目录名称及文件名称不允许调整(control、datapath、mips.v、code.txt 都属于同一层；control 目录下包括 ctrl.v；datapath 目录下包括 im.v、dm.v，等等)。

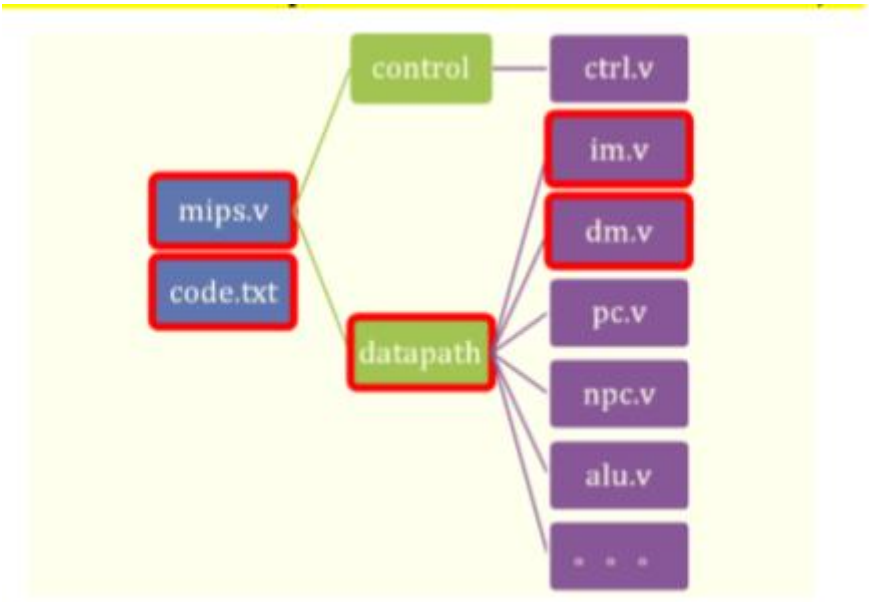

Figure2 参考的 project 目录组织

   b) 顶层设计文件命名为 mips.v。

   c) 建议 datapath 中的每个 module 都由一个独立的 VerilogHDL 文件组成。

d) 建议所有 mux（包括不同位数、不同端口数等）都建模在一个 mux.v 中。可以有多个 module。

4. code.txt 中存储的是指令码

a) 用 VerilogHDL 建模 IM 时，必须以读取文件的方式将 code.txt 中指令加载至 IM 中。

b) code.txt 的格式如 Figure3 所示。每条指令占用 1 行，指令二进制码以文本方式存储。

```
1 34010001
2 34020008
3 34100000
4 34110008
5 3c12aabb
6 12200009
```

Figure3code.txt 文件格式

5. 为使得代码更加清晰可读，建议多使用宏定义，并将宏定义组织在合理的头文件中。

6. 【不是本实验的基本要求，附加题： 有加分，如果将测试指令首地址改为 0x0000_3000，后面指令的地址依次增加 0x0000_3000，来体现与实际 MARS 仿真器中的情况，即开始执行时，PC 指向 0x0000_3000。另外，对于 J 类指令及相关的指令也要做些修改，来体现跳转目标地址。这样的话，先在 MARS 中完成模拟仿真，然后再在单周期 CPU 中运行测试。】

三、 模块定义【WORD】

1. 仿照下面给出的 PC 模块定义，给出所有功能部件的模块定义。
2. PC 模块定义(参考样例)

(1) 基本描述

PC 主要功能是完成输出当前指令地址并保存下一条指令地址。复位后，PC 指向 0x0000_0000 【注：如果按照上面附加题的要求，0x0000_3000】，此处为第一条指令的地址。

(2) 模块接口

信号名	方向	描述
NPC[31:2]	I	下条指令的地址
clk	I	时钟信号

Reset	I	复位信号。 1：复位 0：无效
PC[31:2]	O	30 位指令存储器地址(最低 2 位省略)

(3) 功能定义

序号	功能名称	功能描述
1	复位	当复位信号有效时，PC 被设置为 0x0000_3000。
2	保存 NPC 并输出	在每个 clock 的上升沿保存 NPC，并输出。

7. 下列模块必须严格满足如下的接口定义：

a) 你必须在 VerilogHDL 设计中建模这 3 个模块。

b) 不允许修改模块名称、端口各信号以及变量的名称/类型/位宽。

文件	模块接口定义

mips.v	`module mips(clk, rst) ;`
	`input clk ; // clock`
	`input rst ;// reset`

im.v	`im_4k( addr, dout ) ;`
	`input [11:2] addr ; // address bus`
	`output [31:0] dout ; // 32-bit memory output`
	
	`reg [31:0] im[1023:0] ;`

dm.v	`dm_4k( addr, din, we, clk, dout ) ;`
	`input [11:2] addr ; // address bus`
	`input [31:0] din ; // 32-bit input data`
	`input we ; // memory write enable`
	`input clk ; // clock`
	`output [31:0] dout ; // 32-bit memory output`
	
	`reg [31:0] dm[1023:0] ;`

四、 测设要求

1. 所有 36 条指令都应按照提供的测试文件，测试充分。
2. 如果完成了 10 条(或 N 条，0<N<36,)指令的， 构造包括至少 10 条(或 N 条)指令的测试程序（将提供给你测试程序），并测试通过，每条指令至少出现 1 次以上。
3. 详细说明你的测试程序原理及测试结果。【WORD】
   a. 应明确说明测试程序的测试期望，即应该得到怎样的运行结果。
   b. 每条汇编指令都应该有注释。

五、 问答

1. C 语言是一种弱类型程序设计语言。C 语言中不对计算结果溢出进行处理，这意味着 C 语言要求程序员必须很清楚计算结果是否会导致溢出。因此，如果仅仅支持 C 语言，MIPS 指令的所有计算指令均可以忽略溢出。
   a. 请说明为什么在忽略溢出的前提下，addi 与 addiu 是等价的，add 与 addu 是等价的。提示：阅读《MIPS32® Architecture For Programmers Volume II: The MIPS32® Instruction Set》中相关指令的 Operation 部分。

六、 其他要求

1. 打包文件：Verilog HDL 工程文件、code.txt、code.txt 所对应的汇编程序、项目报告。
2. 本实验要求文档中凡是出现了【WORD】字样，就意味着该条目需要在实验报告中清晰表达。
3. 实验报告要求排版规范。

七、 成绩及实验测试要求

1. 实验成绩包括但不限于如下内容：初始设计的正确性、增加新指令后的正确性、实验报告等。
2. 实验测试时，你必须已经完成了处理器设计及开发。
   ① 允许实验报告可以未完成。
   ② 实验测试时，你需要展示你的设计并证明其正确性。
   ③ 应简洁的描述你的验证思路，并尽可能予以直观展示。
3. 考查时，实验指导教师会检查学生用提供的 testbench 和 code.txt 检测代码执模拟仿真运行情况，并进行现场提问。

八、 开发与调试技巧

1. 对于每条指令，请认真阅读《206_实验指导-含 36 条 MIPS 汇编指令解释及格式-参考 68 到 78 页》，(《MIPS32® Architecture For Programmers Volume II: The MIPS32® Instruction Set》作为参考资料)。如果测试时，你无法清楚的解释所要求的指令，测试成绩将减一档！

2. 利用$readmemh 系统任务可以给存储器初始化数据。例如可以把 code.txt 文件中的数据加载至 my_memory 模块。

```
reg [31:0] my_memory[1023:0] ;

initial
    $readmemh( "code.txt", my_memory ) ;
```

3. 有时我们需要较为集中的在顶层 testbench 中观察甚至修改下层模块的变量，那么你可以通过使用层次路径名来非常方便的达到这一目的。例如：

```
module testbench ;
    ChilC1(…) ;

    $display(C1.Art) ;
endmodule

moduleChil(…) ;
    reg Art;

    …
endmodule
```