P7 – MIPS微体系——异常与中断

#### 一．整体结构

1. 处理器应支持MIPS-lite4指令集。

MIPS-C4 = {LB、LBU、LH、LHU、LW、SB、SH、SW、ADD、ADDU、SUB、SUBU、MULT、MULTU、DIV、DIVU、SLL、SRL、SRA、SLLV、SRLV、SRAV、AND、OR、XOR、NOR、ADDI、ADDIU、ANDI、ORI、XORI、LUI、SLT、SLTI、SLTIU、SLTU、BEQ、BNE、BLEZ、BGTZ、BLTZ、BGEZ、J、JAL、JALR、JR、MFHI、MFLO、MTHI、MTLO、**ERET、MFC0、MTC0**}

2. 处理器为流水线设计。

#### 二．模块规格

1.pc.v

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| pc.v | module pc(  input clk,  input reset,  input en,  input[31:0] next\_pc,  output reg[31:0] pc,  output [4:0] excode\_F  ); |

模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 功能描述 |
| Clk | I | 时钟信号 |
| Reset | I | 复位信号  1：复位  0：无效 |
| en | I | 使能信号 |
| next\_pc | I | 更新的PC（时钟上升沿更新） |
| Pc | I | PC |

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0x00003000 |
| 2 | 更新pc | 时钟上升沿时改变pc=next\_pc |

2.im.v

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| im.v | module im(  input [31:0] PC,  output[31:0] instruction  ); |

模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 功能描述 |
| PC[31:0] | I | 32位PC |
| Instruction[31:0] | O | 32位当前指令 |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 取指令 | 根据 PC 从 IM 中取出指令 |

3.ID.v

模块接口

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| ID.v | module ID(  input clk,  input reset,  input en,  input [31:0] Instr,  input [31:0] PC,  output reg[31:0] IR\_D,  output reg[31:0] PC\_D,  output reg[31:0] PC4\_D,  output reg[31:0] PC8\_D,  output [4:0] excode\_D  ); |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | IF/ID流水线寄存器 | 保存PC,IR等信号的值 |

4.grf.v

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| grf.v | module grf(  input clk,  input reset,  input RegWrite,  input [4:0] RA1,  input [4:0] RA2,  input [4:0] WA,  input [31:0] WD,  input [31:0] PC,  output [31:0] RD1,  output [31:0] RD2  ); |

模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 功能描述 |
| WD[31:0] | I | 写入数据的输入 |
| RA1[4:0] | I | 读寄存器地址 1 |
| RA2[4:0] | I | 读寄存器地址 2 |
| WA[4:0] | I | 写寄存器地址 |
| Clk | I | 时钟信号 |
| Reset | I | 复位信号  1：复位  0：无效 |
| PC[31:0] | I | 当前PC |
| RegWrite | I | 是否可以写入控制信号(随时都可以读出)  1：可以写  0：不可以写 |
| RD1[31:0] | O | 32 位数据输出 1 |
| RD2[31:0] | O | 32 位数据输出 2 |

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，所有寄存器被设置为 0x00000000 |
| 2 | 读寄存器 | 根据输入的寄存器地址读出32位数据 |
| 3 | 写寄存器 | 根据输入的地址，把输入的数据写进所选的寄存器 |

5.cmp.v

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| cmp.v | module cmp(  input [31:0] D1,  input [31:0] D2,  output Equal  ); |

模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 功能描述 |
| D1[31:0] | I | 输入1 |
| D2[31:0] | I | 输入2 |
| Equal | O | 判断两个输入是否相等 |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 比较器 | 比较两个输入是否相等 |

6. ext.v

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| ext.v | module ext(  input [15:0] in,  input [1:0] ExtOp,  output reg [31:0] out  ); |

模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 功能描述 |
| In[15:0] | I | 16 位数据输入 |
| Out[31:0] | O | 32 位数据输出 |
| ExtOp[1:0] | I | 扩展方式选择信号 |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 高位符号扩展 | 高16位补符号位 |
| 2 | 高位0扩展 | 高16位补0 |
| 3. | 低位0扩展 | 低16位补0 |

7. npc.v

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| npc.v | module npc(  input [31:0] PC4,  input [31:0] PC4D,  input [25:0] I26,  input [31:0] MFRSD,  input [31:0] EPC,  input Zero,  input more,  input less,  input if\_beq,  input if\_bne,  input if\_bgtz,  input if\_blez,  input if\_bgez,  input if\_bltz,  input if\_j,//j或jal  input[1:0] PC\_sel,  input Interrupt,  output reg[31:0] next\_pc  ); |

模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 功能描述 |
| PC4 | I | PC+4的值(对应于无跳转 直接执行下一句) |
| PC4D | I | D级PC+4 |
| I26 | I | 26位立即数 |
| MFRSD | I | 转发PC的MUX结果（jr jalr需要转发） |
| Zero | I | 比较两个数是否相等的结果 |
| Branch | I | 判断是不是beq类指令 |
| If\_j | I | 判断是不是j/jal指令 |
| PC\_sel[1:0] | I | PC的选择信号 |
| Next\_pc | O | 更新的pc值 |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 更新PC | 更新PC |

8.controller.v （分布式译码 实例化4个）

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| controller.v | module controller(  input [5:0] op,  input [5:0] func,  input [4:0] rs,  input [4:0] rt,  output reg[3:0] ALUCtrl,  output reg[1:0] RegDst,  output reg ALUASrc,  output reg ALUBSrc,  output reg RegWrite,  output reg MemRead,  output reg MemWrite,  output reg [2:0] MemtoReg,  output reg [1:0]ExtOp,  output reg if\_beq,  output reg if\_bne,  output reg if\_blez,  output reg if\_bgez,  output reg if\_bltz,  output reg if\_bgtz,  output reg if\_j,  output reg [1:0]PCsel,  output reg if\_sh,  output reg if\_sb,  output reg[2:0] dataOp,  output reg[1:0] multdivOp,  output reg start,  output reg if\_mthi,  output reg if\_mtlo,  output reg if\_mfhi,  output reg if\_mflo,  output reg if\_1,  output reg if\_2,  output reg if\_3,  output reg cp0WE  ); |

模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 功能描述 |
| Op[5:0] | I | 6位opcode段 |
| Func[5:0] | I | 6位func段 |
| ALUCtrl[3:0] | O | ALU控制信号 |
| RegDst[1:0] | O | 写地址控制 选择RT,RD |
| ALUASrc | O | ALU第一操作数选择控制 |
| ALUBSrc | O | ALU第二操作数选择控制 |
| RegWrite | O | GRF 写入控制 |
| MemRead | O | DM读信号 |
| MemWrite | O | DM写信号 |
| MemToReg[1:0] | O | GRF写入数据的选择信号 |
| ExtOp | O | 高位扩展方式选择信号 |
| If\_beq | O | 判断是否为beq指令的信号 |
| If\_bne | O | 判断是否为bne指令的信号 |
| If\_bgez | O | 判断是否为bgez指令的信号 |
| If\_blez | O | 判断是否为blez指令的信号 |
| If\_bgtz | O | 判断是否为bgtz指令的信号 |
| If\_bltz | O | 判断是否为bltz指令的信号 |
| If\_j | O | 判断是不是jal/j指令 是则为1 |
| PC\_sel[1:0] | O | PC选择信号 |
| If\_sh | O | 判断是否为sh指令的信号 |
| If\_sb | O | 判断是否为sb指令的信号 |
| dataOp | O | 数据扩展方式控制信号 |
| multdivOp | O | 乘除法方式控制信号 |
| Start | O | 乘除法开始信号 |
| If\_mthi | O | 判断是否为if\_mthi指令的信号 |
| If\_mtlo | O | 判断是否为if\_mtlo指令的信号 |
| If\_mfhi | O | 判断是否为if\_mfhi指令的信号 |
| If\_mflo | O | 判断是否为if\_mflo指令的信号 |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 产生控制信号 | 产生控制信号 |

9.EX.v

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| EX.v | module EX(  input clk,  input reset,  input en,  input over,  input [31:0] IR\_D,  input [31:0] PC\_D,  input [31:0] PC4\_D,  input [31:0] PC8\_D,  input [31:0] RF\_RD1,  input [31:0] RF\_RD2,  input [31:0] EXT,  input Zero,  input more,  input less,  output reg[31:0] IR\_E,  output reg[31:0] PC\_E,  output reg[31:0] PC4\_E,  output reg[31:0] PC8\_E,  output reg[31:0] RS\_E,  output reg[31:0] RT\_E,  output reg[31:0] EXT\_E,  output [6:2] excode\_E,  output reg Zero\_E,  output reg more\_E,  output reg less\_E  ); |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | ID/EX流水线寄存器 | 保存PC,IR等信号的值 |

10.alu.v

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| alu.v | module alu(  input [31:0] A,  input [31:0] B,  input [3:0] ALUCtrl,  output reg[31:0] Result  ); |

模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 功能描述 |
| A[31:0] | I | 32 位输入数据 1 |
| B[31:0] | I | 32 位输入数据 2 |
| ALUCtrl[3:0] | I | 控制信号  000：与  001：或  010：加  011：减  100：移位 |
| Result[31:0] | O | 32 位数据输出 |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 与 | A&B |
| 2 | 或 | A|B |
| 3 | 加 | A+B |
| 4 | 减 | A-B |
| 5 | 异或 | A^B |
| 6 | 或非 | ~(A|B) |
| 7 | 逻辑左 | B << A[4:0] |
| 8 | 逻辑右 | B >> A[4:0] |
| 9 | 算数右 | $signed($signed(B) >>> A[4:0]); |
| 10 | 符号数小于置一 | ($signed(A)<$signed(B)) ? 32'b1 : 32'b0; |
| 11 | 无符号数小于置一 | (A<B) ? 32'b1 : 32'b0; |

11. Mult\_Div.v

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| Mult\_Div.v | module Mult\_Div(  input clk,  input reset,  input [31:0] A,  input [31:0] B,  input [1:0] op,  input start,  input if\_mthi,  input if\_mtlo,  output reg Busy,  output [31:0] High,  output [31:0] Low  ); |

模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 功能描述 |
| Clk | I | 时钟信号 |
| Reset | I | 复位信号 |
| A | I | 输入A |
| B | I | 输入B |
| Op | I | 运算方式选择 |
| Start | I | 开始信号 |
| If\_mthi | I | 判断是不是mthi |
| If\_mtlo | I | 判断是不是mtlo |
| Busy | O | 忙碌信号 |
| High | O | High寄存器 |
| Low | O | Low寄存器 |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 无符号乘 | 无符号乘 |
| 2 | 符号乘 | 符号乘 |
| 3 | 无符号除 | 无符号除 |
| 4 | 符号除 | 符号除 |

12.MEM.v

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| MEM.v | module MEM(  input clk,  input reset,  input en,  input [31:0] IR\_E,  input [31:0] PC\_E,  input [31:0] PC4\_E,  input [31:0] PC8\_E,  input [31:0] ALU,  input [31:0] Mult\_Div,  input [31:0] RT\_E,  output reg[31:0] IR\_M,  output reg[31:0] PC\_M,  output reg[31:0] PC4\_M,  output reg[31:0] PC8\_M,  output reg[31:0] AO\_M,  output reg[31:0] MDO\_M,  output reg[31:0] RT\_M  ); |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | EX/MEM流水线寄存器 | 保存PC,IR等信号的值 |

13.dm.v

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| dm.v | module dm(  input clk,  input reset,  input MemWrite,  input MemRead,  input [31:0] MemAddr,  input [31:0] WD,  input [31:0] PC,  output [31:0] RD  ); |

模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 功能描述 |
| Clk | I | 时钟信号 |
| Reset | I | 复位信号  1：复位  0：无效 |
| MemWrite | I | 读写控制信号  1：写操作 |
| MemRead | I | 读写控制信号  1：读操作 |
| MemAddr[31:0] | I | 操作寄存器地址 |
| WD[31:0] | I | 输入（写入内存）的32位数据 |
| PC[31:0] | I | 当前PC |
| RD[31:0] | O | 32 位数据输出 |

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，所有数据被设置为 0x00000000 |
| 2 | 读 | 根据输入的寄存器地址读出数据 |
| 3 | 写 | 根据输入的地址，把输入的数据写入 |

14.WB.v

模块接口

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| WB.v | module WB(  input clk,  input reset,  input en,  input [31:0] IR\_M,  input [31:0] PC\_M,  input [31:0] PC4\_M,  input [31:0] PC8\_M,  input [31:0] AO\_M,  input [31:0] MDO\_M,  input [31:0] DM,  output reg[31:0] IR\_W,  output reg[31:0] PC\_W,  output reg[31:0] PC4\_W,  output reg[31:0] PC8\_W,  output reg[31:0] AO\_W,  output reg[31:0] MDO\_W,  output reg[31:0] DR\_W  ); |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | MEM/WB流水线寄存器 | 保存PC,IR等信号的值 |

15. DataExt.v

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| DataExt.v | module DataExt(  input [31:0] Din,  input [2:0] dataOp,  input [1:0] Addr,  output reg[31:0] Dout  ); |

模块接口

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 功能描述 |
| Din[31:0] | I | 32位输入 |
| dataOp[2:0] | I | 扩展方式控制信号 |
| Addr[1:0] | I | 地址信号 |
| Dout[31:0] | O | 扩展结果 |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 扩展 | 扩展 |
| 2 | 无扩展 | 无扩展 |
| 3 | 无符号字节数据扩展 | 无符号字节数据扩展 |
| 4 | 符号字节数据扩展 | 符号字节数据扩展 |
| 5 | 无符号半字数据扩展 | 无符号半字数据扩展 |
| 6 | 符号半字数据扩展 | 符号半字数据扩展 |

16.mux.v

模块接口

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| mux.v | module mux(  input [31:0] EXT\_E,  input [31:0] IR\_E,  input [31:0] IR\_W,  input [31:0] DR\_Wnew,  input [31:0] AO\_W,  input [31:0] MDO\_W,  input [31:0] PC8\_W,  input [31:0] CP0\_W,  input [31:0] MFRSE,  input [31:0] MFRTE,  input [31:0] High,  input [31:0] Low,  input ALUasel,  input ALUbsel,  input if\_mfhi,  input if\_mflo,  input [1:0] RegDst,  input [2:0] MemtoReg,  output reg[31:0] ALU\_A,  output reg[31:0] ALU\_B,  output reg[4:0] MUX\_A3,  output reg[31:0] MUX\_WD,  output reg[31:0] MD\_out  ); |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 多路选择器 | 各级多路选择器  ALU\_A,ALU\_B,MUX\_WD,MUX\_A3 |

17.forward\_mux.v

模块接口

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| Forward\_mux.v | module forward\_mux(  input [31:0] RS\_E,  input [31:0] RT\_E,  input [31:0] RT\_M,  input [31:0] WD,  input [31:0] AO\_M,  input [31:0] MDO\_M,  input [31:0] CP0\_M,  input [31:0] MD\_out,  input [31:0] PC8\_E,  input [31:0] PC8\_M,  input [31:0] PC8\_W,  input [31:0] RF\_RD1,  input [31:0] RF\_RD2,  input [31:0] EPCout,  input [2:0] ForwardRSD,  input [2:0] ForwardRTD,  input [2:0] ForwardRSE,  input [2:0] ForwardRTE,  input [2:0] ForwardRTM,  input [2:0] ForwardERET,  output reg[31:0] MFRSD,  output reg[31:0] MFRTD,  output reg[31:0] MFRSE,  output reg[31:0] MFRTE,  output reg[31:0] MFRTM,  output reg[31:0] MFERET  ); |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 各级转发MUX | 转发信号的选择  MFRSD,MFRTD,MFRSE,MFRTE,MFRTM |

18. hazardUnit.v

模块接口

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| hazardUnit.v | module hazardUnit(  input [31:0] IR\_D,  input [31:0] IR\_E,  input [31:0] IR\_M,  input [31:0] IR\_W,  input Busy,  input start,  output IR\_D\_en,  output IR\_E\_clr,  output PC\_en,  output [2:0]ForwardRSD,  output [2:0]ForwardRTD,  output [2:0]ForwardRSE,  output [2:0]ForwardRTE,  output [2:0]ForwardRTM,  output [2:0]ForwardERET  ); |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 冒险控制单元 | 产生转发和暂停的控制信号 |

19. ExcCode.v

模块接口

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| ExcCode.v | module ExcCode(  input clk,  input reset,  input en,  input [4:0] ExcCodeIn,  output reg[4:0] ExcCodeOut  ); |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | 异常代码传递 | 异常代码传递 |

20.CP0.v

模块接口

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| CP0.v | module CP0(  input[4:0] A1,//读CP0寄存器编号 执行MFC0指令时产生  input[4:0] A2,//写CP0寄存器编号 执行MTC0指令时产生  input[31:0] DIn,//CP0寄存器的写入数据 执行MTC0指令时产生 数据来自GPR  input[31:0] PC,//中断/异常时的PC  input[31:0] IR\_M,//指令  input Zero,  input more,  input less,  input if\_bd,  input[6:2] ExcCode,//中断/异常的类型  input[5:0] HWInt,//6个设备中断  input We,//CP0写使能 执行MTC0指令时产生  input EXLSet,//用于置位SR的EXL(EXL为1) 流水线在M阶段产生  input EXLClr,//用于清除SR的EXL(EXL为0) 执行ERET指令时产生  input clk,  input reset,  output Interrupt,//中断和异常 是HWInt/IM/EXL/IE的函数  output[31:0] EPC,//EPC寄存器输出至NPC  output[31:0] DOut //CP0寄存器的输出数据 执行MFC0指令时产生，输出数据至GRF  ); |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | CP0 | CP0 |

21.CPU.v

模块接口

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| CPU.v | module CPU(  input clk,  input reset,  input [7:2] HWInt,  input [31:0] PrRD,  input [6:2] ExcCode,  output [31:0] PrAddr,  output [31:0] PrWD,  output PrWe  ); |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | CPU | CPU |

22.Bridge.v

模块接口

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| Bridge.v | module Bridge(  input [31:0] PrAddr,  input PrWE,  output [31:0] PrRD,  input [31:0] PrWD,  output [31:0] DEV\_Addr,  output [31:0] DEV\_WD,  input [31:0] DEV0\_RD,  input [31:0] DEV1\_RD,  output WeDEV0,  output WeDEV1  ); |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | Bridge | Bridge |

23.DEV0.v

模块接口

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| DEV0.v | module DEV0(  input clk,  input reset,  input [31:0] Addr,  input WE,  input [31:0] DataIn,  output [31:0] DataOut,  output IRQ  ); |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | DEV0 | 计时器 |

24.DEV1.v

模块接口

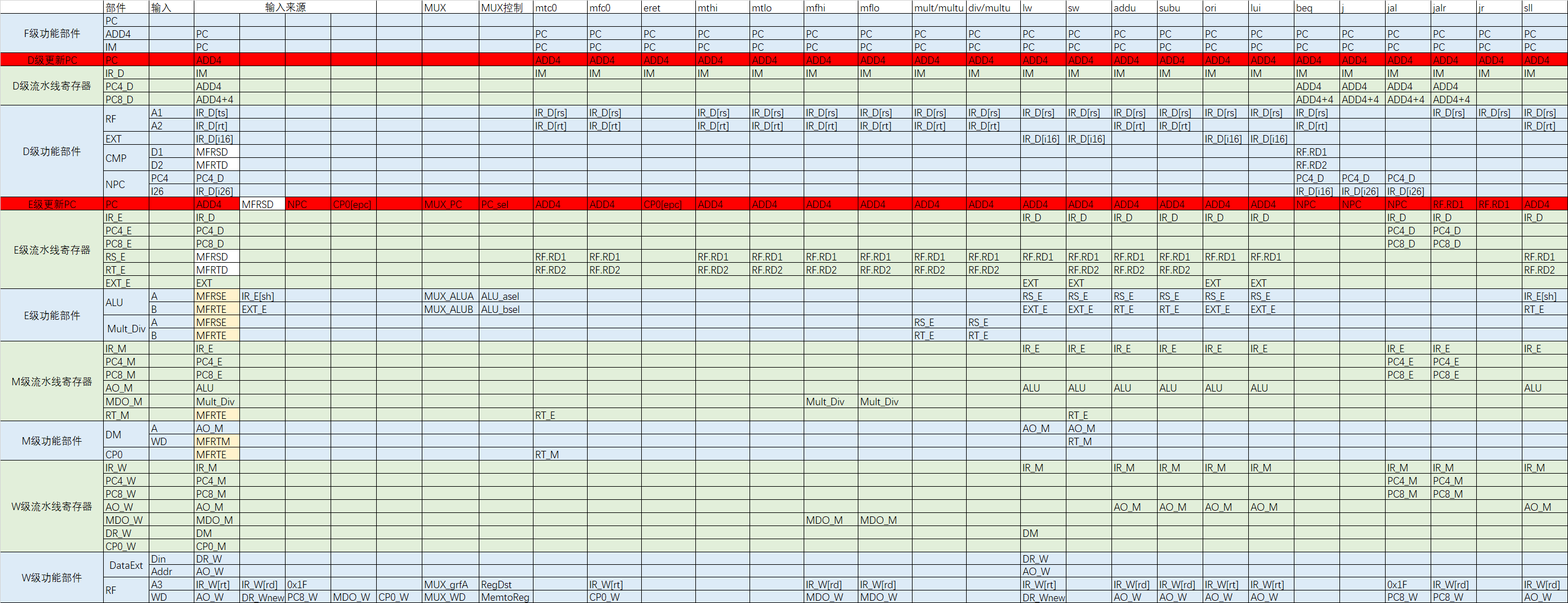
|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| DEV1.v | module DEV1(  input clk,  input reset,  input [31:0] Addr,  input WE,  input [31:0] DataIn,  output [31:0] DataOut,  output IRQ  ); |

功能定义

|  |  |  |
| --- | --- | --- |
| 序号 | 功能名称 | 功能描述 |
| 1 | DEV1 | 计时器 |

#### 三．控制器设计

数据通路如下



由此可见需要以下几个MUX多路选择器

1.GRF的WA端选择Rd,Rt需要一个MUX，控制信号RegDst[1:0]

2.GRF的WD输入端，有三种选择：RF.RD2，ALU的输出，lui指令直接对imm16后边补16位0，需要2选4MUX,选择信号MemToReg[1:0]

3.扩展方式的选择（符号扩展，0扩展）选择信号EXTOp[1:0]

4. ALU的A端两种选择，RF.RD1或IR\_E[sh]的输出，选择信号ALUASrc

5.ALU的B端两种选择，RF.RD2或EXT的输出，选择信号ALUBSrc

6.j/jal指令 跳转地址的选择 if\_j

7.PC的选择信号 PCsel[1:0]

8.beq类指令 跳转地址的选择 Branch

除了上述Branch, ALUASrc, ALUBSrc, EXTOp[1:0], MemToReg[1:0], RegDst[1:0] ,if\_j, PC\_sel[1:0] 还有三个读写控制信号，RegWrite是GRF写入信号，

MemRead, MemWrite是DM读写信号，ALUCtrl[2:0]是ALU控制信号，所以控制器Controller需要设计这12个控制信号。

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 功能描述 |
| Op[5:0] | I | 6位opcode段 |
| Func[5:0] | I | 6位func段 |
| ALUCtrl[3:0] | O | ALU控制信号 |
| RegDst[1:0] | O | 写地址控制 选择RT,RD |
| ALUASrc | O | ALU第一操作数选择控制 |
| ALUBSrc | O | ALU第二操作数选择控制 |
| RegWrite | O | GRF 写入控制 |
| MemRead | O | DM读信号 |
| MemWrite | O | DM写信号 |
| MemToReg[1:0] | O | GRF写入数据的选择信号 |
| ExtOp | O | 高位扩展方式选择信号 |
| If\_beq | O | 判断是否为beq指令的信号 |
| If\_bne | O | 判断是否为bne指令的信号 |
| If\_bgez | O | 判断是否为bgez指令的信号 |
| If\_blez | O | 判断是否为blez指令的信号 |
| If\_bgtz | O | 判断是否为bgtz指令的信号 |
| If\_bltz | O | 判断是否为bltz指令的信号 |
| If\_j | O | 判断是不是jal/j指令 是则为1 |
| PC\_sel[1:0] | O | PC选择信号 |
| If\_sh | O | 判断是否为sh指令的信号 |
| If\_sb | O | 判断是否为sb指令的信号 |
| dataOp | O | 数据扩展方式控制信号 |
| multdivOp | O | 乘除法方式控制信号 |
| Start | O | 乘除法开始信号 |
| If\_mthi | O | 判断是否为if\_mthi指令的信号 |
| If\_mtlo | O | 判断是否为if\_mtlo指令的信号 |
| If\_mfhi | O | 判断是否为if\_mfhi指令的信号 |
| If\_mflo | O | 判断是否为if\_mflo指令的信号 |

画出如下表格

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| name | lw | sw | beq | lui | ori | jal | j | addu | subu | jr | sll | Jalr |
| Op5 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Op4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Op3 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Op2 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Op1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| Op0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Func5 |  |  |  |  |  |  |  | 1 | 1 | 0 | 0 | 0 |
| Func4 |  |  |  |  |  |  |  | 0 | 0 | 0 | 0 | 0 |
| Func3 |  |  |  |  |  |  |  | 0 | 0 | 1 | 0 | 1 |
| Func2 |  |  |  |  |  |  |  | 0 | 0 | 0 | 0 | 0 |
| Func1 |  |  |  |  |  |  |  | 0 | 1 | 0 | 0 | 0 |
| Func0 |  |  |  |  |  |  |  | 1 | 1 | 0 | 0 | 1 |
| RegDst[1:0] | 00 | 00 | 00 | 00 | 00 | 10 | 00 | 01 | 01 | 01 | 01 | 01 |
| ALUASrc | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| ALUBSrc | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| RegWrite | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| MemRead | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| MemWrite | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| MemToReg[1:0] | 01 | 00 | 00 | 00 | 00 | 10 | 00 | 00 | 00 | 00 | 00 | 10 |
| EXTOp[1:0] | 00 | 00 | 00 | 10 | 01 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| If\_beq | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| ALUCtrl[3:0] | 0010 | 0010 | 0111 | 0010 | 0001 | 0111 | 0111 | 0010 | 0011 | 0111 | 0100 | 0000 |
| If\_j | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| PC\_sel[1:0] | 00 | 00 | 10 | 00 | 00 | 10 | 10 | 00 | 00 | 01 | 00 | 01 |

分布式译码 实例化四级控制器（译码器）

controller my\_controllerD(.op(IR\_D[`op]),.func(IR\_D[`func]),.rt(IR\_D[`rt]),.ExtOp(EXTop),.if\_beq(if\_beq),.if\_bne(if\_bne),.if\_blez(if\_blez),.if\_bgtz(if\_bgtz),.if\_bgez(if\_bgez),.if\_bltz(if\_bltz),.if\_j(if\_j),.PCsel(PC\_sel));

controller my\_controllerE(.op(IR\_E[`op]),.func(IR\_E[`func]),.rt(IR\_D[`rt]),.ALUCtrl(ALUCtrl),.ALUASrc(ALUASrc),.ALUBSrc(ALUBSrc),.multdivOp(multdivOp),.start(start),.if\_mthi(if\_mthi),.if\_mtlo(if\_mtlo),.if\_mfhi(if\_mfhi),.if\_mflo(if\_mflo));

controller my\_controllerM(.op(IR\_M[`op]),.func(IR\_M[`func]),.rt(IR\_D[`rt]),.MemRead(MemRead),.MemWrite(MemWrite),.if\_sh(if\_sh),.if\_sb(if\_sb));

controller my\_controllerW(.op(IR\_W[`op]),.func(IR\_W[`func]),.rt(IR\_D[`rt]),.RegDst(RegDst),.RegWrite(RegWrite),.MemtoReg(MemtoReg),.dataOp(dataOp));

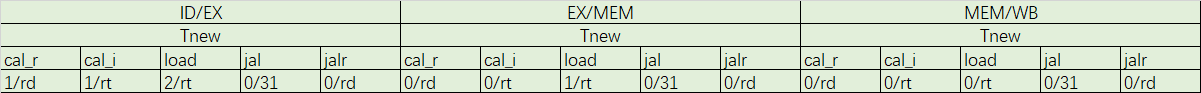
#### 四．冒险处理单元设计

**需求时间——供给时间模型**。

Tuse



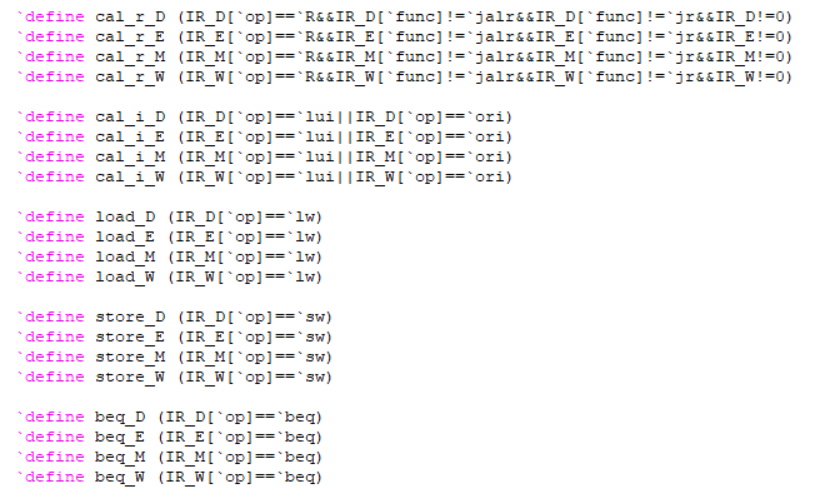
Tnew

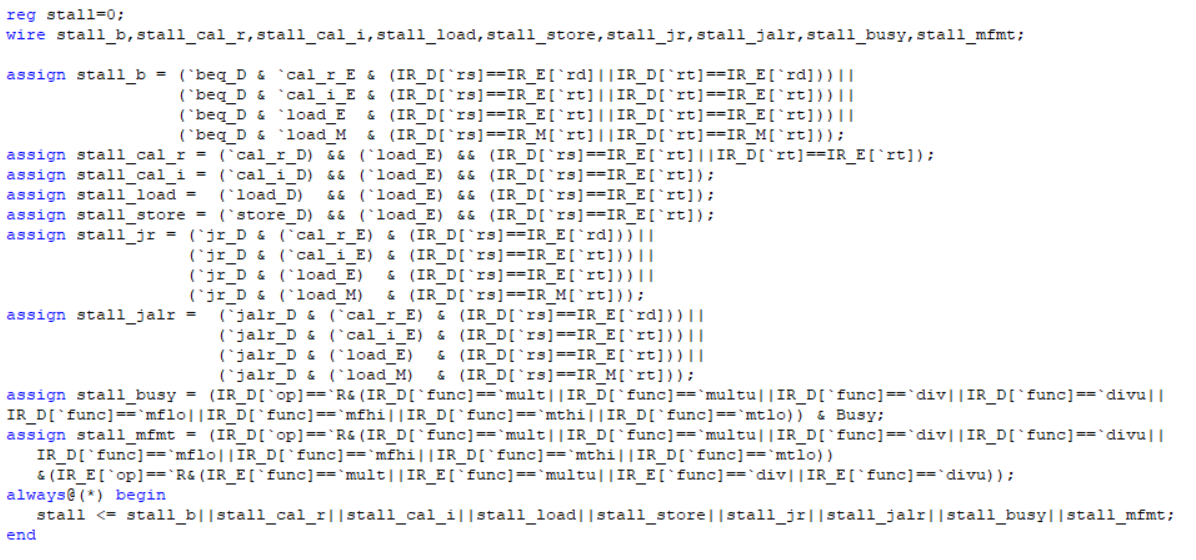


暂停

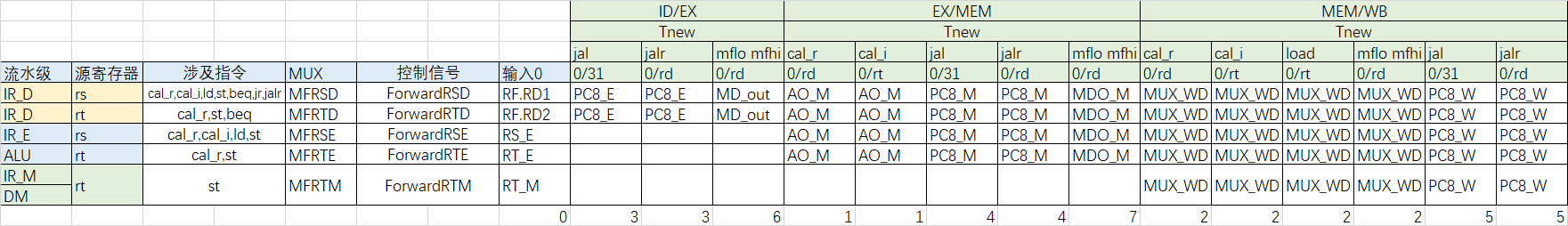
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| IF/ID当前指令 | | | ID/EX | | | EX/MEM |
| 指令类型 | 源寄存器 | Tuse | Tnew | | | Tnew |
| cal\_r | cal\_i | load | load |
| 1/rd | 1/rt | 2/rt | 1/rt |
| beq | rs/rt | 0 | 暂停 | 暂停 | 暂停 | 暂停 |
| cal\_r | rs/rt | 1 |  |  | 暂停 |  |
| cal\_i | rs | 1 |  |  | 暂停 |  |
| load | rs | 1 |  |  | 暂停 |  |
| store | rs | 1 |  |  | 暂停 |  |
| store | rt | 2 |  |  |  |  |
| jr | rs | 0 | 暂停 | 暂停 | 暂停 | 暂停 |
| jalr | rs | 0 | 暂停 | 暂停 | 暂停 | 暂停 |

由此可以写出各种控制信号的表达式如下

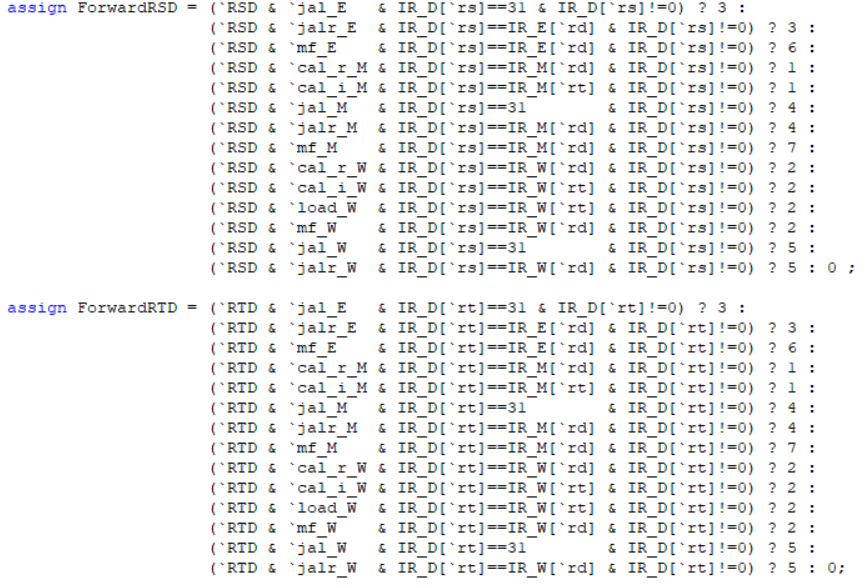


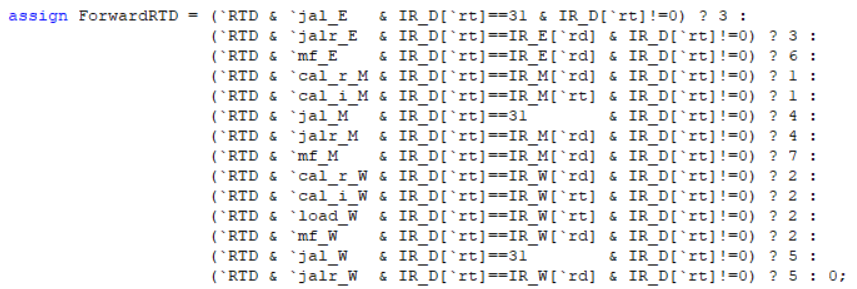


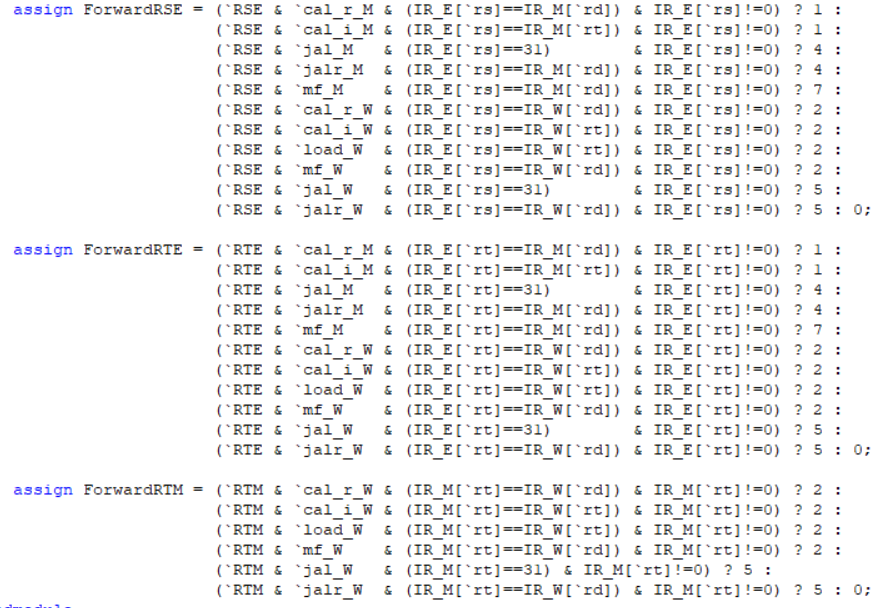
转发

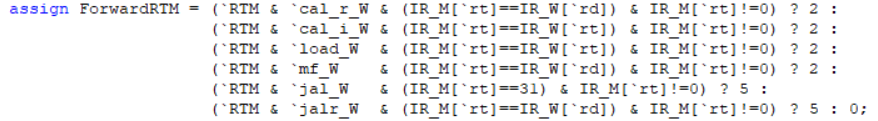


由此可以写出各种控制信号的表达式如下



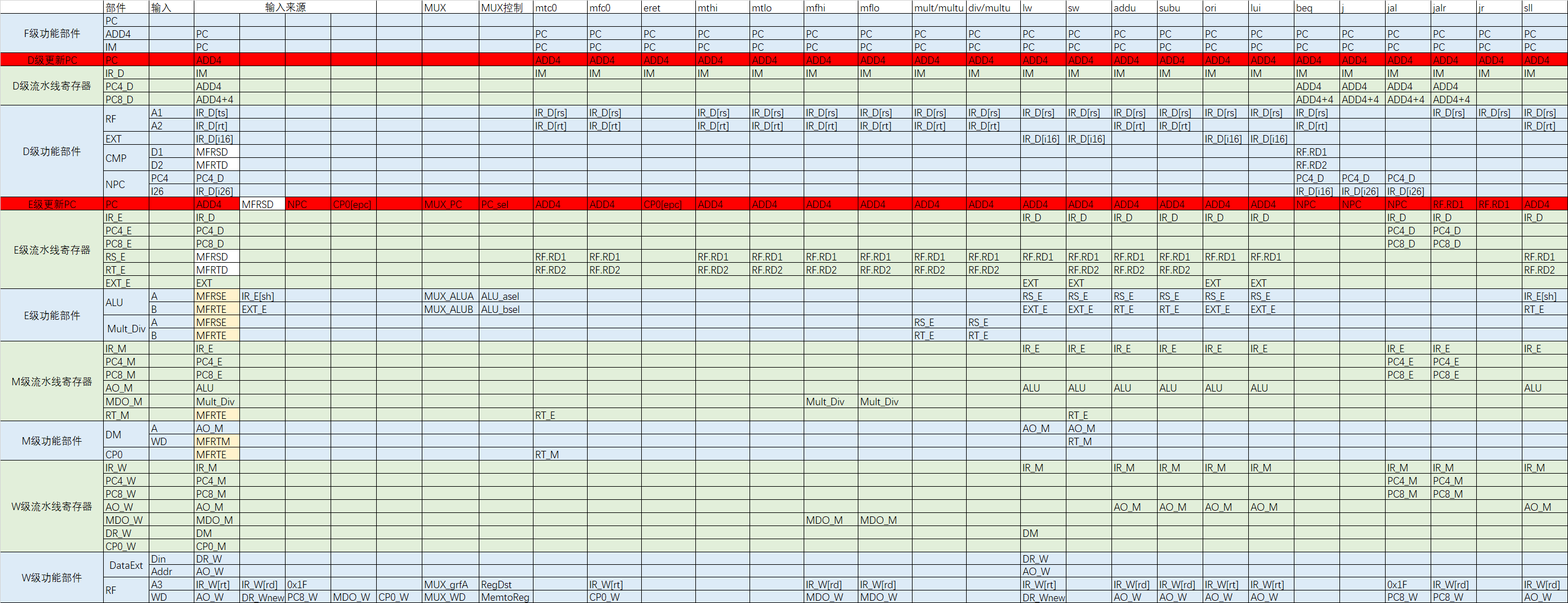








更新后的数据通路 加入了转发MUX



Forward\_mux代码如下

always@(\*) begin

case(ForwardRSD)

3'b000 : MFRSD <= RF\_RD1;

3'b001 : MFRSD <= AO\_M;

3'b010 : MFRSD <= WD;

3'b011 : MFRSD <= PC8\_E;

3'b100 : MFRSD <= PC8\_M;

3'b101 : MFRSD <= CP0\_M;

3'b110 : MFRSD <= MD\_out;

3'b111 : MFRSD <= MDO\_M;

default : MFRSD <= 0;

endcase

case(ForwardRTD)

0: MFRTD <= RF\_RD2;

1: MFRTD <= AO\_M;

2: MFRTD <= WD;

3: MFRTD <= PC8\_E;

4: MFRTD <= PC8\_M;

5: MFRTD <= CP0\_M;

6: MFRTD <= MD\_out;

7: MFRTD <= MDO\_M;

default: MFRTD <= 0;

endcase

case(ForwardRSE)

0:MFRSE <= RS\_E;

1:MFRSE <= AO\_M;

2:MFRSE <= WD;

3:MFRSE <= 0;

4:MFRSE <= PC8\_M;

5:MFRSE <= CP0\_M;

6:MFRSE <= 0;

7:MFRSE <= MDO\_M;

default:MFRSE <= 0;

endcase

case(ForwardRTE)

0:MFRTE <= RT\_E;

1:MFRTE <= AO\_M;

2:MFRTE <= WD;

3:MFRTE <= 0;

4:MFRTE <= PC8\_M;

5:MFRTE <= CP0\_M;

6:MFRTE <= 0;

7:MFRTE <= MDO\_M;

default:MFRTE <= 0;

endcase

case(ForwardRTM)

0:MFRTM <= RT\_M;

1:MFRTM <= 0;

2:MFRTM <= WD;

3:MFRTM <= 0;

4:MFRTM <= 0;

5:MFRTM <= 0;

6:MFRTM <= 0;

7:MFRTM <= 0;

default:MFRTM <= 0;

endcase

case(ForwardERET)

0:MFERET <= EPCout;

1:MFERET <= MFRTM;

default:MFERET <= 0;

endcase

end

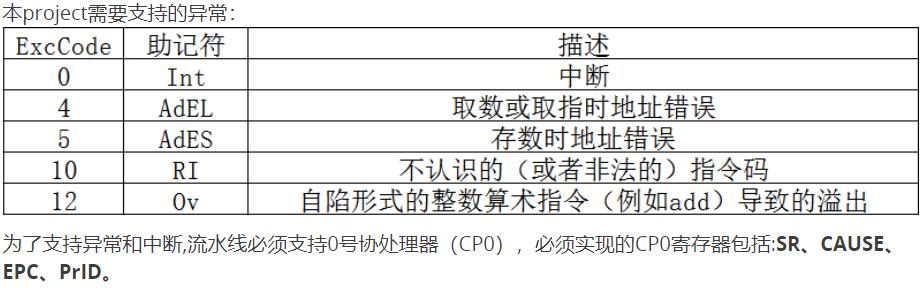
#### 五．中断异常设计

ExcCode.v

模块接口

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| ExcCode.v | module ExcCode(  input clk,  input reset,  input en,  input [4:0] ExcCodeIn,  output reg[4:0] ExcCodeOut  ); |

ExcCode用来传递每一级的异常编码



CP0 行为规范

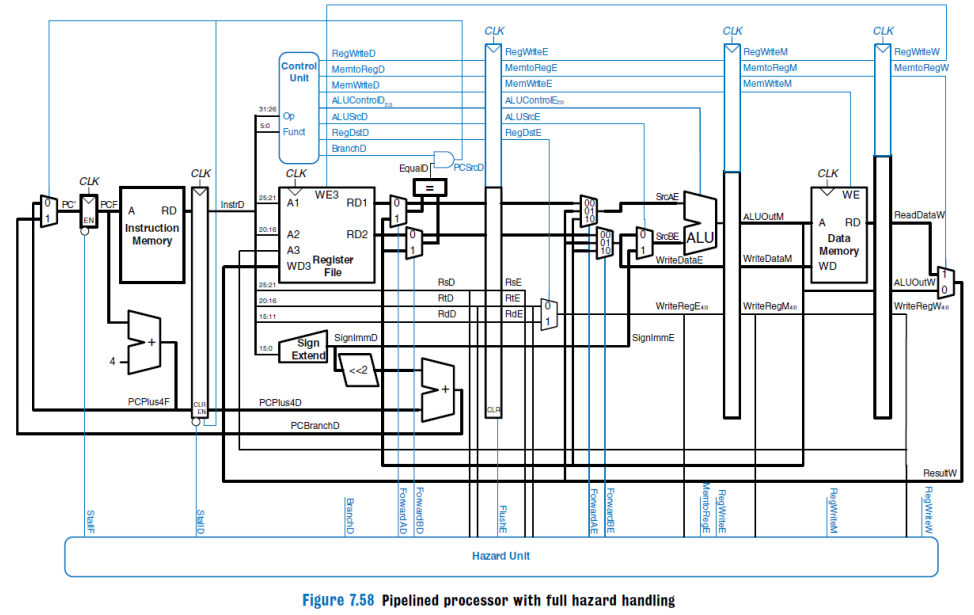
1. 本实验要求支持 SR、Cause、EPC、PRId 四个 CP0 寄存器。
2. CP0的位置不做明确要求，需要自行设计
3. 模块规格可以参考课件上的设计，不作硬性要求。
4. SR 寄存器行为需要与课件上行为保持一致。
5. Cause 寄存器在课件的基础上需要增加 BD 位，和 ExcCode 位（因此需要对顶层模块做相应的修改）。行为规范参考 《See MIPS Run Linux》。
6. EPC 寄存器架构与课件上保持一致。异常发生时 EPC 存入的值参考 《See MIPS Run Linux》，中断发生时 EPC 存入的值可以自行设计。
7. PRId 寄存器的值不作要求。

CP0.v

模块接口

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| CP0.v | module CP0(  input[4:0] A1,//读CP0寄存器编号 执行MFC0指令时产生  input[4:0] A2,//写CP0寄存器编号 执行MTC0指令时产生  input[31:0] DIn,//CP0寄存器的写入数据 执行MTC0指令时产生 数据来自GPR  input[31:0] PC,//中断/异常时的PC  input[31:0] IR\_M,//指令  input Zero,  input more,  input less,  input if\_bd,  input[6:2] ExcCode,//中断/异常的类型  input[5:0] HWInt,//6个设备中断  input We,//CP0写使能 执行MTC0指令时产生  input EXLSet,//用于置位SR的EXL(EXL为1) 流水线在M阶段产生  input EXLClr,//用于清除SR的EXL(EXL为0) 执行ERET指令时产生  input clk,  input reset,  output Interrupt,//中断和异常 是HWInt/IM/EXL/IE的函数  output[31:0] EPC,//EPC寄存器输出至NPC  output[31:0] DOut //CP0寄存器的输出数据 执行MFC0指令时产生，输出数据至GRF  ); |

#### 六.主程序



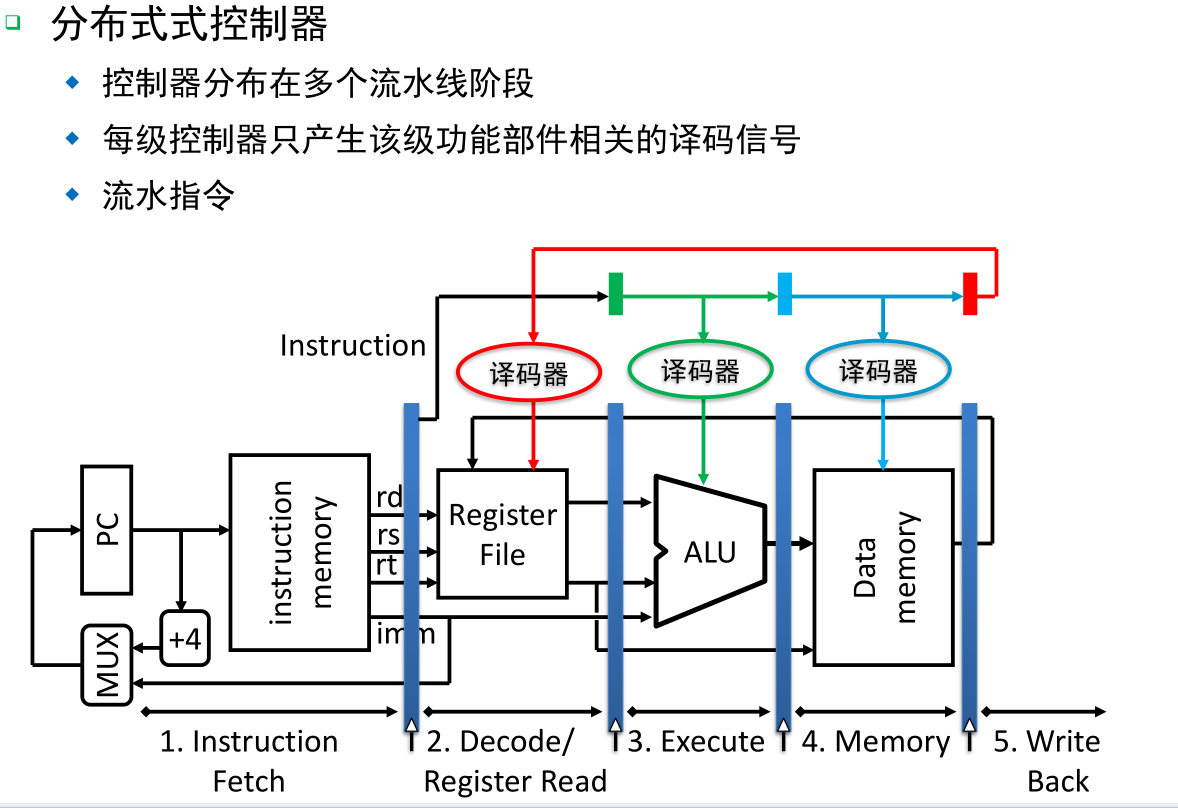
数据通路主要采用如上架构 区别是分布式译码

1. 流水线的设计以追求性能为第一目标，因此必须尽最大可能**支持转发**以解决数据冒险。这一点在本project的最终成绩中所占比重较大，课上测试时会通过测试程序所跑的**总周期数**进行判定，望大家慎重对待。

2. 对于 b 类和 j 类指令， 流水线设计必须**支持延迟槽**，因此设计需要注意使用 **PC+8**。

3. 为了解决数据冒险而设计的转发数据来源必须是**某级流水线寄存器**，**不允许**对功能部件的输出直接进行转发。

4.分布式译码



需要以下几个MUX多路选择器

1.GRF的WA端选择Rd,Rt需要一个MUX，控制信号RegDst[1:0]

2.GRF的WD输入端，有三种选择：RF.RD2，ALU的输出，lui指令直接对imm16后边补16位0，需要2选4MUX,选择信号MemToReg[1:0]

3. ALU的A端两种选择，RF.RD1或IR\_E[sh]的输出，选择信号ALUASrc

4.ALU的B端两种选择，RF.RD2或EXT的输出，选择信号ALUBSrc

1.mux.v

模块接口

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| mux.v | module mux(  input [31:0] EXT\_E,  input [31:0] IR\_E,  input [31:0] IR\_W,  input [31:0] DR\_Wnew,  input [31:0] AO\_W,  input [31:0] MDO\_W,  input [31:0] PC8\_W,  input [31:0] CP0\_W,  input [31:0] MFRSE,  input [31:0] MFRTE,  input [31:0] High,  input [31:0] Low,  input ALUasel,  input ALUbsel,  input if\_mfhi,  input if\_mflo,  input [1:0] RegDst,  input [2:0] MemtoReg,  output reg[31:0] ALU\_A,  output reg[31:0] ALU\_B,  output reg[4:0] MUX\_A3,  output reg[31:0] MUX\_WD,  output reg[31:0] MD\_out  ); |

2.CPU.v

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| CPU.v | module CPU(  input clk,  input reset,  input [7:2] HWInt,  input [31:0] PrRD,  input [6:2] ExcCode,  output [31:0] PrAddr,  output [31:0] PrWD,  output PrWe  ); |

3.tb

module test;

// Inputs

reg clk;

reg reset;

// Instantiate the Unit Under Test (UUT)

mips uut (

.clk(clk),

.reset(reset)

);

initial begin

clk = 0;

reset = 1;

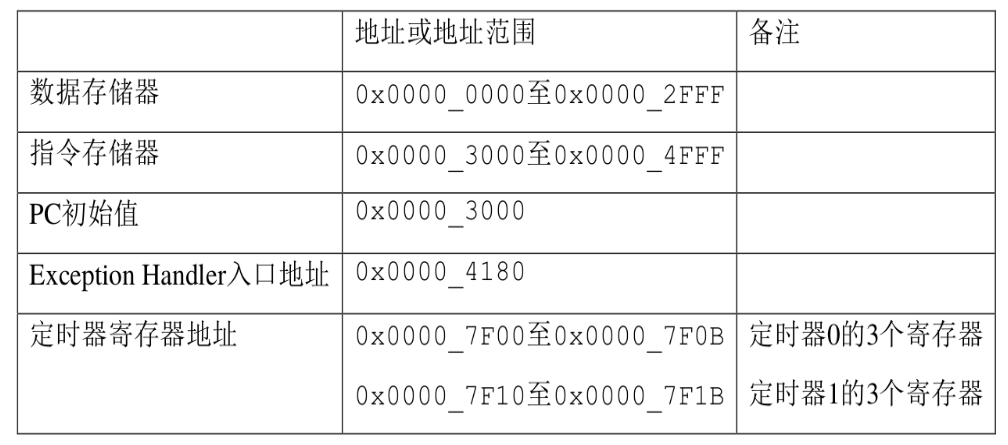
#12 reset = 0;

end

always #10 clk = ~clk;

endmodule

4.外设 计时器



功能描述及内部结构

TC 的内部基本结构如图 1-1 所示。TC 由控制寄存器、初值寄存器、32 位

计数器及中断产生逻辑构成。

1) 控制寄存器决定该计数起停控制等。

2) 初值寄存器为 32 位计数器提供初始值。

3) 根据不同的计数模式，在计数为 0 后，计数器或者自动装填初值并重新倒计数，或者保持在 0 值直至计数器使能再次被设置为 1。

4) 使用 store 类指令修改 TC 寄存器值的优先级高于 TC 自修改的优先级。

5) 当计数器计数时，若计数器使能被 store 类指令修改为 0 则停止计数。

6) 当计数器工作在模式 0 并且在中断允许的前提下，当计数器计数值为 0 时，

中断产生逻辑产生中断请求(IRQ 为 1)。

计数模式

2.1. 模式 0

当计数器倒计数为 0 后，计数器停止计数，此时控制寄存器中的使能 Enable

自动变为 0。当使能 Enable 被设置为 1 后，初值寄存器值再次被加载至计数器，

计数器重新启动倒计数。模式 0 通常用于产生定时中断。例如，为操作系统的时间片调度机制提供定时。模式 0 下的中断信号将持续有效，直至控制寄存器中的中断屏蔽位被设置为0。

2.2. 模式 1

当计数器倒计数为 0 后，初值寄存器值被自动加载至计数器，计数器继续倒数计数。模式 1 通常用于产生周期性脉冲。例如，可以用模式 1 产生步进电机所需的步进控制信号。不同于模式 0，模式 1 下计数器每次计数循环中只产生一周期的

中断信号。





1.在允许计数器计数前，应首先停止计数；然后加载初值寄存器；再允许计数。2.无论哪种模式，如果不需要产生中断，则应屏蔽中断。

DEV0.v

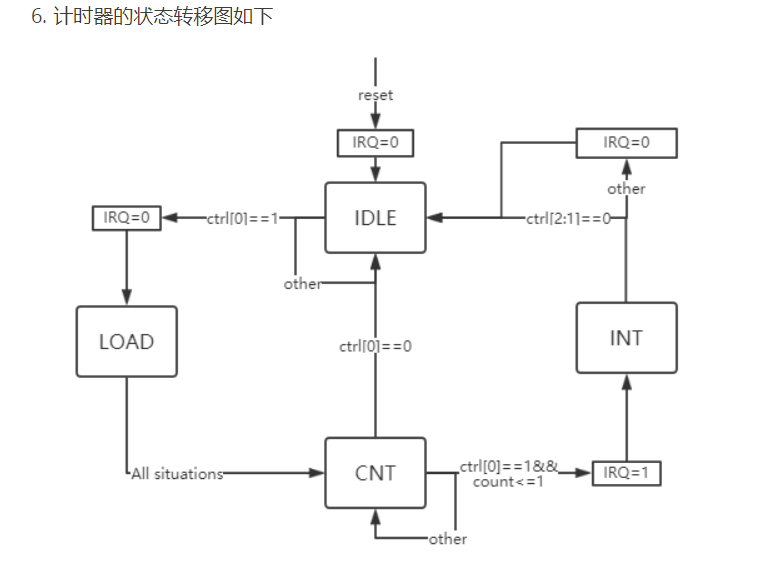
模块接口

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| DEV0.v | module DEV0(  input clk,  input reset,  input [31:0] Addr,  input WE,  input [31:0] DataIn,  output [31:0] DataOut,  output IRQ  ); |

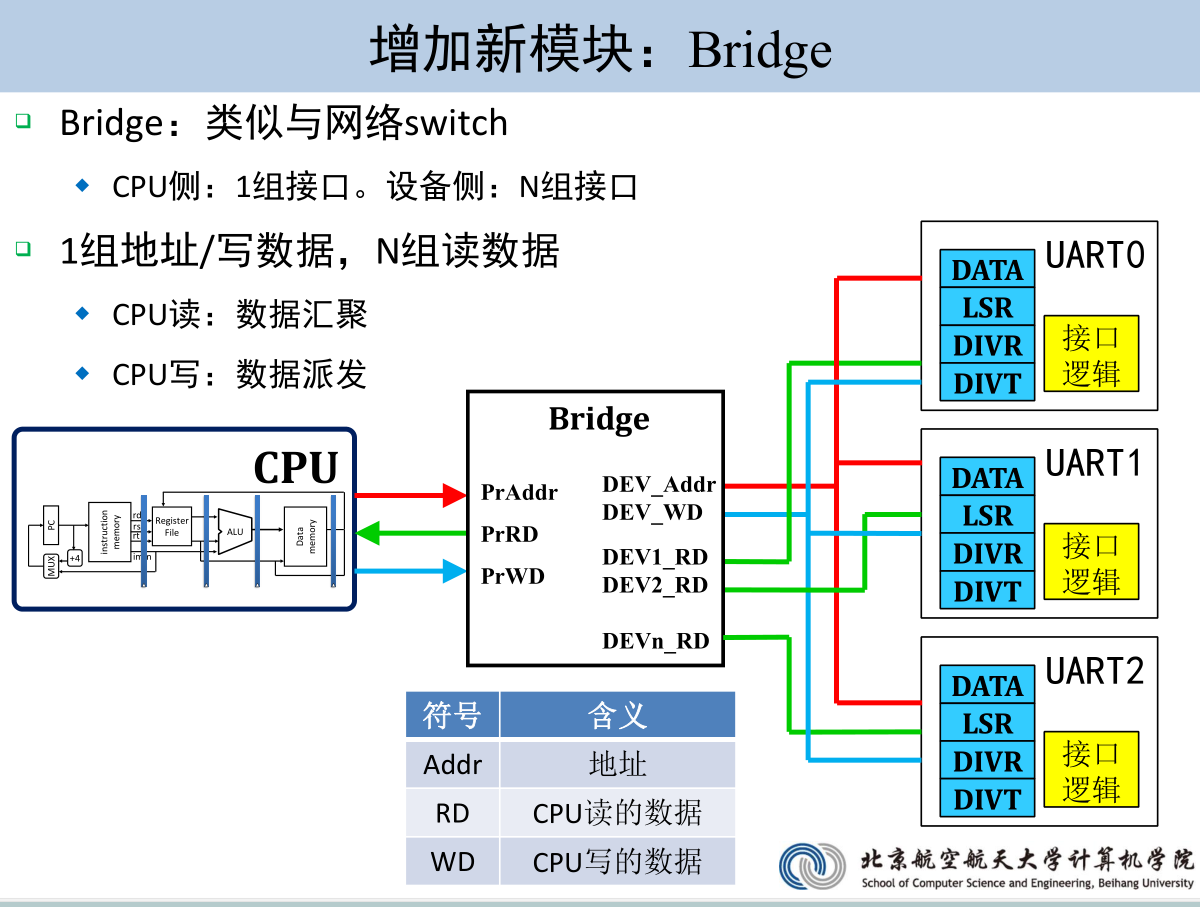
DEV1.v

模块接口

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| DEV1.v | module DEV1(  input clk,  input reset,  input [31:0] Addr,  input WE,  input [31:0] DataIn,  output [31:0] DataOut,  output IRQ  ); |



5.Bridge



Bridge.v

模块接口

|  |  |
| --- | --- |
| 文件 | 模块接口定义 |
| Bridge.v | module Bridge(  input [31:0] PrAddr,  input PrWE,  output [31:0] PrRD,  input [31:0] PrWD,  output [31:0] DEV\_Addr,  output [31:0] DEV\_WD,  input [31:0] DEV0\_RD,  input [31:0] DEV1\_RD,  output WeDEV0,  output WeDEV1  ); |

#### 七．测试程序

命令行导出：java -jar E:\Mars4\_5.jar a db mc CompactDataAtZero dump 0x00004180-0x00004ffc HexText E:\code\_handler.txt E:\mips4.asm

##### (1)转发测试

**测试目录：**

一．D级rs

1.D级Rs与E级jal

2.D级Rs与E级jalr

3.D级Rs与E级 mflo/mfhi

4.D级Rs与M级cal\_r

5.D级Rs与M级cal\_i

6.D级Rs与M级jal

7.D级Rs与M级jalr

8.D级Rs与M级mflo/mfhi

9.D级Rs与W级cal\_r

10.D级Rs与W级cal\_i

11.D级Rs与W级load rt

12.D级Rs与W级jal

13.D级Rs与W级jalr

14.D级Rs与W级jalr

每一项又分为：cal\_r,cal\_i,ld,st,beq,jr,jalr

二．D级Rt

1.D级Rt与E级jal

2.D级Rt与E级jalr

3.D级Rt与E级 mflo/mfhi

4.D级Rt与M级cal\_r

5.D级Rt与M级cal\_i

6.D级Rt与M级jal

7.D级Rt与M级jalr

8.D级Rt与M级mflo/mfhi

9.D级Rt与W级cal\_r

10.D级Rt与W级cal\_i

11.D级Rt与W级load rt

12.D级Rt与W级jal

13.D级Rt与W级jalr

14.D级Rt与W级jalr

每一项又分为：cal\_r,st,beq

三．E级Rs

1.E级Rs与M级cal\_r

2.E级Rs与M级cal\_i

3.E级Rs与M级jal

4.E级Rs与M级jalr

5.E级Rs与M级mflo/mfhi

6.E级Rs与W级cal\_r

7.E级Rs与W级cal\_i

8.E级Rs与W级load

9.E级Rs与W级jal

10.E级Rs与W级jalr

11.E级Rs与W级mflo/mfhi

每一项又分为：cal\_r,cal\_i,ld,st

四．E级Rt

1.E级Rt与M级cal\_r

2.E级Rt与M级cal\_i

3.E级Rt与M级jal

4.E级Rt与M级jalr

5.E级Rt与M级mflo/mfhi

6.E级Rt与W级cal\_r

7.E级Rt与W级cal\_i

8.E级Rt与W级load

9.E级Rt与W级jal

10.E级Rt与W级jalr

11.E级Rt与W级mflo/mfhi

每一项又分为：cal\_r,st

五．M级Rt

1.M级Rt与W级cal\_r

2.M级Rt与W级cal\_i

3.M级Rt与W级load

4.M级Rt与W级jal

5.M级Rt与W级jalr

6.M级Rt与W级mflo/mfhi

每一项又分为：st

一．D级rs

1.D级rs与E级jal

(1)Rs----cal\_r

ori $t0,11

jal eee

addu $t0,$ra,$0

eee:

110@00003000: $ 8 <= 0000000b

130@00003004: $31 <= 0000300c

150@00003008: $ 8 <= 0000300c

(2)Rs---cal\_i

ori $t0,11

jal eee

ori $t0,$ra,11

eee:

110@00003000: $ 8 <= 0000000b

130@00003004: $31 <= 0000300c

150@00003008: $ 8 <= 0000300f

2.D级Rs与E级jalr

(1)Rs----cal\_r

ori $t0,0x0000300c

jalr $t1,$t0

addu $t2,$t1,$t0

nop

110@00003000: $ 8 <= 0000300c

150@00003004: $ 9 <= 0000300c

170@00003008: $10 <= 00006018

(2)Rs---cal\_i

ori $t0,0x0000300c

jalr $t1,$t0

xori $t2,$t1,111

nop

110@00003000: $ 8 <= 0000300c

150@00003004: $ 9 <= 0000300c

170@00003008: $10 <= 00003063

3.D级Rs与E级 mflo/mfhi

(1)Rs----cal\_r

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

addu $t1,$t2,$t0

mfhi $t2

and $t1,$t2,$t0

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

310@00003010: $ 9 <= 0000008f

330@00003014: $10 <= 00000000

350@00003018: $ 9 <= 00000000

(2)Rs---cal\_i

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

lui $t1,$t2,100

mfhi $t2

addiu $t1,$t2,99

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

310@00003010: $ 9 <= 000000e8

330@00003014: $10 <= 00000000

350@00003018: $ 9 <= 00000063

(3)Rs---load

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

lb $t1,0($t2)

mfhi $t2

lbu $t1,0($t2)

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

310@00003010: $ 9 <= 00000000

330@00003014: $10 <= 00000000

350@00003018: $ 9 <= 00000000

(4)Rs---store

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

sh $t1,0($t2)

mfhi $t2

sb $t1,0($t2)

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

290@00003010: \*00000084 <= 0000000c

330@00003014: $10 <= 00000000

330@00003018: \*00000000 <= 0000000c

(5)Rs---beq

ori $t0,11

ori $t1,12

multu $t0,$t1

mfhi $t2

beq $t2,$t1,out

nop

out:

nop

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000000

(6)Rs---jr

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

jr $t2

nop

out:

nop

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

(7) Rs---jalr

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

jalr $t1,$t2

nop

out:

nop

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

310@00003010: $ 9 <= 00003018

4.D级Rs与M级cal\_r

(1)Rs----cal\_r

ori $t0,11

addu $t1,$t2,$t0

nop

addu $t2,$t1,$t0

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000b

170@0000300c: $10 <= 00000016

(2)Rs---cal\_i

ori $t0,11

addu $t1,$t2,$t0

nop

ori $t0,$t1,1

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000b

170@0000300c: $ 8 <= 0000000b

(3)Rs---load

addu $a0,$0,$0

nop

lw $t0,0($a0)

110@00003000: $ 4 <= 00000000

150@00003008: $ 8 <= 00000000

(4)Rs---store

ori $t0,111

addu $a0,$0,$0

nop

sw $t0,0($a0)

110@00003000: $ 8 <= 0000006f

130@00003004: $ 4 <= 00000000

150@0000300c: \*00000000 <= 0000006f

(5)Rs---beq

ori $t1,1

addu $t2,$t1,$0

nop

beq $t2,$t1,out

nop

out:

nop

110@00003000: $ 9 <= 00000001

130@00003004: $10 <= 00000001

(6)Rs---jr

ori $t1,0x00003014

addu $t2,$t1,$0

nop

jr $t2

nop

out:

nop

110@00003000: $ 9 <= 00003014

130@00003004: $10 <= 00003014

(7)Rs-jalr

ori $t1,0x00003014

or $t2,$t1,$0

nop

jalr $t1,$t2

nop

out:

nop

110@00003000: $ 9 <= 00003014

130@00003004: $10 <= 00003014

170@0000300c: $ 9 <= 00003014

5.D级Rs与M级cal\_i

(1)Rs----cal\_r

ori $t0,11

ori $t1,$t2,0

nop

addu $t2,$t1,$t0

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 00000000

170@0000300c: $10 <= 0000000b

(2)Rs---cal\_i

ori $t0,11

ori $t1,$t2,0

nop

ori $t0,$t1,1

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 00000000

170@0000300c: $ 8 <= 00000001

190@00003010: $ 8 <= 0000000b

210@00003014: $ 9 <= 00000000

250@0000301c: $ 8 <= 00000001

(3)Rs---load

ori $a0,$0,0

nop

lw $t0,0($a0)

110@00003000: $ 4 <= 00000000

150@00003008: $ 8 <= 00000000

(4)Rs---store

ori $t0,111

ori $a0,$0,0

nop

sw $t0,0($a0)

110@00003000: $ 8 <= 0000006f

130@00003004: $ 4 <= 00000000

150@0000300c: \*00000000 <= 0000006f

(5)Rs---beq

ori $t1,1

ori $t2,$t1,0

nop

beq $t2,$t1,out

nop

out:

nop

110@00003000: $ 9 <= 00000001

130@00003004: $10 <= 00000001

(6)Rs---jr

ori $t1,0x00003014

ori $t2,$t1,0

nop

jr $t2

nop

out:

nop

110@00003000: $ 9 <= 00003014

130@00003004: $10 <= 00003014

(7)Rs-jalr

ori $t1,0x00003014

ori $t2,$t1,0

nop

jalr $t1,$t2

nop

out:

nop

110@00003000: $ 9 <= 00003014

130@00003004: $10 <= 00003014

170@0000300c: $ 9 <= 00003014

6.D级Rs与M级jal

(1)Rs----cal\_r

ori $t0,11

jal eee

nop

nor $t0,$ra,$0

eee:

110@00003000: $ 8 <= 0000000b

130@00003004: $31 <= 0000300c

(2)Rs---cal\_i

ori $t0,11

jal eee

nop

ori $t0,$ra,11

eee:

110@00003000: $ 8 <= 0000000b

130@00003004: $31 <= 0000300c

(3)Rs---load

ori $a0,$0,0x00003000

jal eee

subu $ra,$ra,$a0

eee:

lw $t1,0($ra)

110@00003000: $ 4 <= 00003000

130@00003004: $31 <= 0000300c

150@00003008: $31 <= 0000000c

170@0000300c: $ 9 <= 00000000

(4)Rs---store

ori $t1,1

ori $a0,$0,0x00003000

jal eee

subu $ra,$ra,$a0

eee:

sw $t1,0($ra)

110@00003000: $ 9 <= 00000001

130@00003004: $ 4 <= 00003000

150@00003008: $31 <= 00003010

170@0000300c: $31 <= 00000010

170@00003010: \*00000010 <= 00000001

7.D级Rs与M级jalr

(1)Rs----cal\_r

ori $t0,0x0000300c

jalr $t1,$t0

nop

addu $t2,$t1,$t0

nop

110@00003000: $ 8 <= 0000300c

150@00003004: $ 9 <= 0000300c

170@0000300c: $10 <= 00006018

(2)Rs---cal\_i

ori $t0,0x0000300c

jalr $t1,$t0

nop

xori $t2,$t1,111

nop

110@00003000: $ 8 <= 0000300c

150@00003004: $ 9 <= 0000300c

170@0000300c: $10 <= 00003063

8.D级Rs与M级 mflo/mfhi

(1)Rs----cal\_r

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

addu $t1,$t2,$t0

mfhi $t2

nop

and $t1,$t2,$t0

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

310@00003010: $ 9 <= 0000008f

330@00003014: $10 <= 00000000

350@00003018: $ 9 <= 00000000

(2)Rs---cal\_i

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

lui $t1,$t2,100

mfhi $t2

nop

addiu $t1,$t2,99

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

310@00003010: $ 9 <= 000000e8

330@00003014: $10 <= 00000000

350@00003018: $ 9 <= 00000063

(3)Rs---load

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

lb $t1,0($t2)

mfhi $t2

nop

lbu $t1,0($t2)

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

310@00003010: $ 9 <= 00000000

330@00003014: $10 <= 00000000

350@00003018: $ 9 <= 00000000

(4)Rs---store

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

sh $t1,0($t2)

mfhi $t2

nop

sb $t1,0($t2)

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

290@00003010: \*00000084 <= 0000000c

330@00003014: $10 <= 00000000

330@00003018: \*00000000 <= 0000000c

(5)Rs---beq

ori $t0,11

ori $t1,12

multu $t0,$t1

mfhi $t2

nop

beq $t2,$t1,out

nop

out:

nop

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000000

(6)Rs---jr

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

nop

jr $t2

nop

out:

nop

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

(7) Rs---jalr

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

nop

jalr $t1,$t2

nop

out:

nop

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

310@00003010: $ 9 <= 00003018

9.D级Rs与W级cal\_r

(1)Rs----cal\_r

ori $t0,11

addu $t1,$t2,$t0

nop

nop

addu $t2,$t1,$t0

(2)Rs---cal\_i

ori $t0,11

addu $t1,$t2,$t0

nop

nop

ori $t0,$t1,1

(3)Rs---load

addu $a0,$0,$0

nop

nop

lw $t0,0($a0)

(4)Rs---store

ori $t0,111

addu $a0,$0,$0

nop

nop

sw $t0,0($a0)

(5)Rs---beq

ori $t1,1

addu $t2,$t1,$0

nop

nop

beq $t2,$t1,out

nop

out:

nop

(6)Rs---jr

ori $t1,0x00003014

addu $t2,$t1,$0

nop

nop

jr $t2

nop

out:

nop

(7)Rs---jalr

ori $t1,0x00003014

addu $t2,$t1,$0

nop

nop

jalr $t1,$t2

nop

out:

nop

10.D级Rs与W级cal\_i

(1)Rs----cal\_r

ori $t0,11

ori $t1,$t2,0

nop

nop

addu $t2,$t1,$t0

(2)Rs---cal\_i

ori $t0,11

ori $t1,$t2,0

nop

nop

ori $t0,$t1,1

(3)Rs---load

ori $a0,$0,0

nop

nop

lw $t0,0($a0)

(4)Rs---store

ori $t0,111

ori $a0,$0,0

nop

nop

sw $t0,0($a0)

(5)Rs---beq

ori $t1,1

ori $t2,$t1,0

nop

nop

beq $t2,$t1,out

nop

out:

nop

(6)Rs---jr

ori $t1,0x00003014

ori $t2,$t1,0

nop

nop

jr $t2

nop

out:

nop

(7)Rs---jalr

ori $t1,0x00003014

ori $t2,$t1,0

nop

nop

jalr $t1,$t2

nop

out:

nop

11.D级Rs与W级load rt

(1)Rs----cal\_r

ori $t1,1

ori $t0,0x00000000

lw $t1,0($t0)

nop

nop

addu $t2,$t1,$t0

110@00003000: $ 9 <= 00000001

130@00003004: $ 8 <= 00000000

150@00003008: $ 9 <= 00000000

210@00003014: $10 <= 00000000

(2)Rs---cal\_i

ori $t1,1

ori $t0,0x00000000

lw $t1,0($t0)

nop

nop

ori $t0,$t1,1

110@00003000: $ 9 <= 00000001

130@00003004: $ 8 <= 00000000

150@00003008: $ 9 <= 00000000

210@00003014: $ 8 <= 00000001

(3)Rs---load

ori $t0,0x00000000

lw $t1,0($t0)

nop

nop

lw $t0,0($t1)

110@00003000: $ 8 <= 00000000

130@00003004: $ 9 <= 00000000

190@00003010: $ 8 <= 00000000

(4)Rs---store

ori $t0,0x00000000

lw $t1,0($t0)

nop

nop

sw $t0,0($t1)

110@00003000: $ 8 <= 00000000

130@00003004: $ 9 <= 00000000

170@00003010: \*00000000 <= 00000000

(5)Rs---beq

ori $t1,1

ori $t0,0x00000000

lw $t1,0($t0)

ori $t2,$t1,0

nop

beq $t1,$t2,out

nop

out:

nop

110@00003000: $ 9 <= 00000001

130@00003004: $ 8 <= 00000000

150@00003008: $ 9 <= 00000000

190@0000300c: $10 <= 00000000

(6)Rs---jr

ori $t0,0x00000000

ori $t2,$0,0x00003020

sw $t2,0($t0)

lw $t1,0($t0)

nop

nop

jr $t1

nop

out:

nop

110@00003000: $ 8 <= 00000000

130@00003004: $10 <= 00003020

130@00003008: \*00000000 <= 00003020

170@0000300c: $ 9 <= 00003020

(7)Rs---jalr

ori $t0,0x00000000

ori $t2,$0,0x00003020

sw $t2,0($t0)

lw $t1,0($t0)

nop

nop

jalr $t2,$t1

nop

out:

nop

110@00003000: $ 8 <= 00000000

130@00003004: $10 <= 00003020

130@00003008: \*00000000 <= 00003020

170@0000300c: $ 9 <= 00003020

230@00003018: $10 <= 00003020

12.D级Rs与W级jal

(1)Rs----cal\_r

ori $t0,11

jal eee

nop

nop

nor $t0,$ra,$0

eee:

110@00003000: $ 8 <= 0000000b

130@00003004: $31 <= 0000300c

(2)Rs---cal\_i

ori $t0,11

jal eee

nop

nop

ori $t0,$ra,11

eee:

110@00003000: $ 8 <= 0000000b

130@00003004: $31 <= 0000300c

(3)Rs---load

ori $a0,$0,0x00003000

jal eee

subu $ra,$ra,$a0

eee:

nop

lw $t1,0($ra)

110@00003000: $ 4 <= 00003000

130@00003004: $31 <= 0000300c

150@00003008: $31 <= 0000000c

170@0000300c: $ 9 <= 00000000

(4)Rs---store

ori $t1,1

ori $a0,$0,0x00003000

jal eee

subu $ra,$ra,$a0

eee:

nop

sw $t1,0($ra)

110@00003000: $ 9 <= 00000001

130@00003004: $ 4 <= 00003000

150@00003008: $31 <= 00003010

170@0000300c: $31 <= 00000010

170@00003010: \*00000010 <= 00000001

13.D级Rs与M级jalr

(1)Rs----cal\_r

ori $t0,0x0000300c

jalr $t1,$t0

nop

nop

addu $t2,$t1,$t0

nop

110@00003000: $ 8 <= 0000300c

150@00003004: $ 9 <= 0000300c

210@00003010: $10 <= 00006018

(2)Rs---cal\_i

ori $t0,0x0000300c

jalr $t1,$t0

nop

nop

xori $t2,$t1,111

nop

110@00003000: $ 8 <= 0000300c

150@00003004: $ 9 <= 0000300c

170@0000300c: $10 <= 00003063

14.D级Rs与M级 mflo/mfhi

(1)Rs----cal\_r

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

addu $t1,$t2,$t0

mfhi $t2

nop

nop

and $t1,$t2,$t0

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

310@00003010: $ 9 <= 0000008f

330@00003014: $10 <= 00000000

350@00003018: $ 9 <= 00000000

(2)Rs---cal\_i

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

lui $t1,$t2,100

mfhi $t2

nop

nop

addiu $t1,$t2,99

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

310@00003010: $ 9 <= 000000e8

330@00003014: $10 <= 00000000

350@00003018: $ 9 <= 00000063

(3)Rs---load

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

lb $t1,0($t2)

mfhi $t2

nop

nop

lbu $t1,0($t2)

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

310@00003010: $ 9 <= 00000000

330@00003014: $10 <= 00000000

350@00003018: $ 9 <= 00000000

(4)Rs---store

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

sh $t1,0($t2)

mfhi $t2

nop

nop

sb $t1,0($t2)

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

290@00003010: \*00000084 <= 0000000c

330@00003014: $10 <= 00000000

330@00003018: \*00000000 <= 0000000c

(5)Rs---beq

ori $t0,11

ori $t1,12

multu $t0,$t1

mfhi $t2

nop

nop

beq $t2,$t1,out

nop

out:

nop

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000000

(6)Rs---jr

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

nop

nop

jr $t2

nop

out:

nop

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

(7) Rs---jalr

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

nop

nop

jalr $t1,$t2

nop

out:

nop

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

350@00003018: $ 9 <= 00003020

二．D级Rt

1.D级rt与E级jal

(1)Rt----cal\_r

ori $t0,11

jal eee

addu $t0,$0,$ra

eee:

110@00003000: $ 8 <= 0000000b

130@00003004: $31 <= 0000300c

150@00003008: $ 8 <= 0000300c

2.D级Rt与E级jalr

(1)Rt----cal\_r

ori $t0,0x0000300c

jalr $t1,$t0

addu $t2,$t0,$t1

nop

110@00003000: $ 8 <= 0000300c

150@00003004: $ 9 <= 0000300c

170@00003008: $10 <= 00006018

3.D级Rt与E级 mflo/mfhi

(1)Rt----cal\_r

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

addu $t1,$t0,$t2

mfhi $t2

and $t1,$t2,$t0

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

310@00003010: $ 9 <= 0000008f

330@00003014: $10 <= 00000000

350@00003018: $ 9 <= 00000000

(2)Rt---store

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

sh $t2,0($t2)

mfhi $t2

sb $t2,0($t2)

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

290@00003010: \*00000084 <= 0000000c

330@00003014: $10 <= 00000000

330@00003018: \*00000000 <= 0000000c

(3)Rt---beq

ori $t0,11

ori $t1,12

multu $t0,$t1

mfhi $t2

beq $t1,$t2,out

nop

out:

nop

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000000

4.D级Rt与M级cal\_r

(1)Rt----cal\_r

ori $t0,11

addu $t1,$t2,$t0

nop

addu $t2,$t0,$t1

(2)Rt---store

ori $t0,111

addu $a0,$t0,$0

nop

sw $a0,0($0)

(3)Rt---beq

ori $t1,1

addu $t2,$t1,$0

nop

beq $t1,$t2,out

nop

out:

nop

5.D级Rt与M级cal\_i

(1)Rt----cal\_r

ori $t0,11

ori $t1,$t2,0

nop

addu $t2,$t0,$t1

(2)Rt---store

ori $a0,$0,111

nop

sw $a0,0($0)

(3)Rt---beq

ori $t1,1

ori $t2,$t1,0

nop

beq $t1,$t2,out

nop

out:

nop

6.D级Rt与M级jal

(1)Rt----cal\_r

ori $t0,11

jal eee

nop

addu $t0,$0,$ra

eee:

(2)Rt---store

ori $t1,1

ori $a0,$0,0x00003000

jal eee

subu $ra,$ra,$a0

eee:

sw $ra,0($0)

7.D级Rt与M级jalr

(1)Rt----cal\_r

ori $t0,0x0000300c

jalr $t1,$t0

nop

addu $t2,$t0,$t1

nop

110@00003000: $ 8 <= 0000300c

150@00003004: $ 9 <= 0000300c

170@00003008: $10 <= 00006018

8.D级Rt与M级 mflo/mfhi

(1)Rt----cal\_r

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

addu $t1,$t0,$t2

mfhi $t2

nop

and $t1,$t2,$t0

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

310@00003010: $ 9 <= 0000008f

330@00003014: $10 <= 00000000

370@0000301c: $ 9 <= 00000000

(2)Rt---store

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

sh $t2,0($t2)

mfhi $t2

nop

sb $t2,0($t2)

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

290@00003010: \*00000084 <= 00000084

330@00003014: $10 <= 00000000

350@0000301c: \*00000000 <= 00000000

(3)Rt---beq

ori $t0,11

ori $t1,12

multu $t0,$t1

mfhi $t2

nop

beq $t1,$t2,out

nop

out:

nop

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000000

9.D级Rt与W级cal\_r

(1)Rt----cal\_r

ori $t0,11

addu $t1,$t2,$t0

nop

nop

addu $t2,$t0,$t1

(2)Rt---store

ori $t0,111

addu $a0,$0,$0

nop

nop

sw $a0,0($0)

(3)Rt---beq

ori $t1,1

addu $t2,$t1,$0

nop

nop

beq $t1,$t2,out

nop

out:

nop

10.D级Rt与W级cal\_i

(1)Rt----cal\_r

ori $t0,11

ori $t1,$t2,0

nop

nop

addu $t2,$t0,$t1

(4)Rt---store

ori $t0,111

ori $a0,$0,0

nop

nop

sw $a0,0($0)

(5)Rt---beq

ori $t1,1

ori $t2,$t1,0

nop

nop

beq $t1,$t2,out

nop

out:

nop

11.D级Rt与W级load rt

(1)Rt----cal\_r

ori $t1,1

ori $t0,0x00000000

lw $t1,0($t0)

nop

nop

addu $t2,$t0,$t1

(2)Rt---store

ori $t0,0x00000000

lw $t1,0($t0)

nop

nop

sw $t1,0($0)

(3)Rs---beq

ori $t1,1

ori $t0,0x00000000

lw $t1,0($t0)

ori $t2,$t1,0

nop

beq $t2,$t1,out

nop

out:

nop

12.D级Rt与W级jal

(1)Rt----cal\_r

ori $t0,11

jal eee

nop

nop

addu $t0,$0,$ra

eee:

(2)Rt---store

ori $t1,1

ori $a0,$0,0x00003000

jal eee

subu $ra,$ra,$a0

eee:

nop

sw $ra,0($0)

13. D级Rt与W级jalr

(1)Rt----cal\_r

ori $t0,0x0000300c

jalr $t1,$t0

nop

nop

addu $t2,$t0,$t1

nop

110@00003000: $ 8 <= 0000300c

150@00003004: $ 9 <= 0000300c

170@00003008: $10 <= 00006018

14.D级Rt与W级 mflo/mfhi

(1)Rt----cal\_r

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

addu $t1,$t0,$t2

mfhi $t2

nop

nop

and $t1,$t2,$t0

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

310@00003010: $ 9 <= 0000008f

330@00003014: $10 <= 00000000

370@0000301c: $ 9 <= 00000000

(2)Rt---store

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

sh $t2,0($t2)

mfhi $t2

nop

nop

sb $t2,0($t2)

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

290@00003010: \*00000084 <= 00000084

330@00003014: $10 <= 00000000

350@0000301c: \*00000000 <= 00000000

(3)Rt---beq

ori $t0,11

ori $t1,12

multu $t0,$t1

mfhi $t2

nop

nop

beq $t1,$t2,out

nop

out:

nop

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000000

三．E级Rs

1.E级Rs与M级cal\_r

(1)Rs----cal\_r

ori $t2,$0,111

addu $t1,$t2,$t3

subu $t4,$t1,$0

110@00003000: $10 <= 0000006f

130@00003004: $ 9 <= 0000006f

150@00003008: $12 <= 0000006f

(2)Rs----cal\_i

ori $t2,$0,111

subu $t4,$t2,$0

ori $t2,$t4,111

(3)Rs----load

addu $t1,$t2,$t3

lw $t2,0($t1)

110@00003000: $ 9 <= 00000000

130@00003004: $10 <= 00000000

(4)Rs----store

addu $t1,$t2,$t3

sw $t2,0($t1)

2.E级Rs与M级cal\_i

(1)Rs----cal\_r

ori $t2,$0,111

ori $t1,$t2,0

subu $t4,$t1,$0

(2)Rs----cal\_i

ori $t2,$0,111

ori $t4,$t2,0

(3)Rs----load

ori $t1,$t2,$t3

lw $t2,0($t1)

(4)Rs----store

ori $t1,$t2,$t3

sw $t2,0($t1)

3.E级Rs与M级jal

(1)Rs----cal\_r

jal eee

subu $t1,$ra,$t2

eee:

(2)Rs----cal\_i

jal eee

lui $ra,111

eee:

(3)Rs----load

jal eee

lw $0,0($ra)

eee:

(4)Rs----store

jal eee

sw $0,0($ra)

eee:

4.E级Rs与M级jalr

(1)Rs----cal\_r

ori $t1,0x0000300c

jalr $t2,$t1

subu $t1,$t1,$t2

nop

110@00003000: $ 9 <= 0000300c

150@00003004: $10 <= 0000300c

170@00003008: $ 9 <= 00000000

(2)Rs----cal\_i

ori $t1,0x0000300c

jalr $t2,$t1

addi $t2,$t2,1

nop

(3)Rs----load

ori $t1,0x0000300c

jalr $t2,$t1

lw $0,0($t2)

(4)Rs----store

ori $t1,0x0000300c

jalr $t2,$t1

sw $0,0($t2)

5.E级Rs与M级mflo/mfhi

(1)Rs----cal\_r

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

sub $t3,$t2,$t2

(2)Rs----cal\_i

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

or $t3,$t2,1

(3)Rs----load

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

lw $t2,0($t2)

(4)Rs----store

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

sw $t2,0($t2)

6.E级Rs与W级cal\_r

(1)Rs----cal\_r

ori $t2,$0,111

addu $t1,$t2,$t3

nop

subu $t4,$t1,$0

(2)Rs----cal\_i

ori $t2,$0,111

subu $t4,$t2,$0

nop

ori $t2,$t4,111

(3)Rs----load

addu $t1,$t2,$t3

nop

lw $t2,0($t1)

(4)Rs----store

addu $t1,$t2,$t3

nop

sw $t2,0($t1)

7.E级Rs与W级cal\_i

(1)Rs----cal\_r

ori $t2,$0,111

ori $t1,$t2,0

nop

subu $t4,$t1,$0

(2)Rs----cal\_i

ori $t2,$0,111

nop

ori $t4,$t2,0

(3)Rs----load

ori $t1,$t2,$t3

nop

lw $t2,0($t1)

(4)Rs----store

ori $t1,$t2,$t3

nop

sw $t2,0($t1)

8.E级Rs与W级load

(1)Rs----cal\_r

ori $t1,1

ori $t0,0x00000000

lw $t1,0($t0)

nop

addu $t2,$t1,$t0

(2)Rs---cal\_i

ori $t1,1

ori $t0,0x00000000

lw $t1,0($t0)

nop

ori $t0,$t1,1

(3)Rs---load

ori $t0,0x00000000

lw $t1,0($t0)

nop

lw $t0,0($t1)

(4)Rs---store

ori $t0,0x00000000

lw $t1,0($t0)

nop

sw $t0,0($t1)

9.E级Rs与W级jal

(1)Rs----cal\_r

jal eee

nop

eee:

subu $t1,$ra,$t2

(2)Rs----cal\_i

jal eee

nop

eee:

lui $ra,111

(3)Rs----load

ori $t1,0x00003000

jal eee

eee:

subu $ra,$ra,$t1

lw $0,0($ra)

(4)Rs----store

ori $t1,0x00003000

jal eee

eee:

subu $ra,$ra,$t1

sw $0,0($ra)

10. E级Rs与M级jalr

(1)Rs----cal\_r

ori $t1,0x0000300c

jalr $t2,$t1

nop

subu $t1,$t1,$t2

nop

110@00003000: $ 9 <= 0000300c

150@00003004: $10 <= 0000300c

170@00003008: $ 9 <= 00000000

(2)Rs----cal\_i

ori $t1,0x0000300c

jalr $t2,$t1

nop

addi $t2,$t2,1

nop

(3)Rs----load

ori $t1,0x0000300c

jalr $t2,$t1

nop

lw $0,0($t2)

(4)Rs----store

ori $t1,0x0000300c

jalr $t2,$t1

nop

sw $0,0($t2)

11.E级Rs与M级mflo/mfhi

(1)Rs----cal\_r

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

nop

sub $t3,$t2,$t2

(2)Rs----cal\_i

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

nop

or $t3,$t2,1

(3)Rs----load

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

nop

lw $t2,0($t2)

(4)Rs----store

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

nop

sw $t2,0($t2)

四．E级Rt

1.E级Rt与M级cal\_r

(1)Rt----cal\_r

ori $t2,$0,111

addu $t1,$t2,$t3

subu $t4,$0,$t1

110@00003000: $10 <= 0000006f

130@00003004: $ 9 <= 0000006f

150@00003008: $12 <= ffffff91

170@0000300c: $10 <= 0000006f

190@00003010: $ 9 <= 0000006f

210@00003014: $12 <= ffffff91

(2)Rt----store

addu $t1,$t2,$t3

sw $t1,0($t2)

110@00003000: $ 9 <= 00000000

110@00003004: \*00000000 <= 00000000

2.E级Rt与M级cal\_i

(1)Rt----cal\_r

ori $t2,$0,111

ori $t1,$t2,0

subu $t4,$0,$t1

110@00003000: $10 <= 0000006f

130@00003004: $ 9 <= 0000006f

150@00003008: $12 <= ffffff91

(2)Rt----store

ori $t1,$t2,100

sw $t1,0($t1)

110@00003000: $ 9 <= 00000064

110@00003004: \*00000064 <= 00000064

3.E级Rt与M级jal

(1)Rt----cal\_r

jal eee

subu $t1,$t2,$ra

eee:

110@00003000: $31 <= 00003008

130@00003004: $ 9 <= ffffcff8

(2)Rt----store

jal eee

sw $ra,0($0)

eee:

110@00003000: $31 <= 00003008

110@00003004: \*00000000 <= 00003008

4. E级Rt与M级jalr

(1)Rt----cal\_r

jalr $t3,$t2

subu $t1,$t2,$t2

110@00003000: $11 <= 00003008

130@00003004: $ 9 <= 00000000

(2)Rt----store

jalr $t3,$t2

sw $t2,0($0)

110@00003000: $11 <= 00003008

110@00003004: \*00000000 <= 00000000

5. E级Rt与M级mflo mfhi

(1)Rt----cal\_r

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

sub $t3,$t2,$t2

(2)Rt----store

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

sw $t2,0($t2)

6.E级Rt与W级cal\_r

(1)Rt----cal\_r

ori $t2,$0,111

addu $t1,$t2,$t3

nop

subu $t4,$0,$t1

110@00003000: $10 <= 0000006f

130@00003004: $ 9 <= 0000006f

170@0000300c: $12 <= ffffff91

(2)Rt----store

addu $t1,$t2,$t3

nop

sw $t1,0($t2)

110@00003000: $ 9 <= 00000000

130@00003008: \*00000000 <= 00000000

7.E级Rt与W级cal\_i

(1)Rt----cal\_r

ori $t2,$0,111

ori $t1,$t2,0

nop

subu $t4,$0,$t1

110@00003000: $10 <= 0000006f

130@00003004: $ 9 <= 0000006f

170@0000300c: $12 <= ffffff91

(2)Rt----store

ori $t1,$t2,$t3

nop

sw $t1,0($t2)

110@00003000: $ 9 <= 00000064

130@00003008: \*00000000 <= 00000064

8.E级Rt与W级load

(1)Rt----cal\_r

ori $t1,1

ori $t0,0x00000000

lw $t1,0($t0)

nop

addu $t2,$t0,$t1

110@00003000: $ 9 <= 00000001

130@00003004: $ 8 <= 00000000

150@00003008: $ 9 <= 00000000

190@00003010: $10 <= 00000000

(2)Rt---store

ori $t0,0x00000000

lw $t1,0($t0)

nop

sw $t1,0($t0)

110@00003000: $ 8 <= 00000000

130@00003004: $ 9 <= 00000000

150@0000300c: \*00000000 <= 00000000

9.E级Rt与W级jal

(1)Rt----cal\_r

jal eee

nop

eee:

subu $t1,$t2,$ra

110@00003000: $31 <= 00003008

150@00003008: $ 9 <= ffffcff8

(2)Rt----store

ori $t1,0x00003000

jal eee

subu $ra,$ra,$t1

eee:

sw $ra,0($0)

110@00003000: $ 9 <= 00003000

130@00003004: $31 <= 0000300c

150@00003008: $31 <= 0000000c

150@0000300c: \*00000000 <= 0000000c

10. E级Rt与W级jalr

(1)Rt----cal\_r

jalr $t3,$t2

nop

subu $t1,$t2,$t2

110@00003000: $11 <= 00003008

130@00003004: $ 9 <= 00000000

(2)Rt----store

jalr $t3,$t2

nop

sw $t2,0($0)

110@00003000: $11 <= 00003008

110@00003004: \*00000000 <= 00000000

11. E级Rt与W级mflo mfhi

(1)Rt----cal\_r

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

nop

sub $t3,$t2,$t2

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

330@00003014: $11 <= 00000000

(2)Rt----store

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

nop

sw $t2,0($t2)

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

310@00003014: \*00000084 <= 00000084

五．M级Rt

1.M级Rt与W级cal\_r

ori $t2,10

addu $t1,$t2,$t3

sw $t1,0($0)

110@00003000: $10 <= 0000000a

130@00003004: $ 9 <= 0000000a

130@00003008: \*00000000 <= 0000000a

2.M级Rt与W级cal\_i

ori $t2,10

ori $t1,$t2,10

sw $t1,0($0)

110@00003000: $10 <= 0000000a

130@00003004: $ 9 <= 0000000a

130@00003008: \*00000000 <= 0000000a

3.M级Rt与W级load

ori $t2,10

lw $t2,0($0)

sw $t2,0($0)

110@00003000: $10 <= 0000000a

130@00003004: $10 <= 00000000

130@00003008: \*00000000 <= 00000000

4.M级Rt与W级jal

ori $t2,10

jal eee

sw $ra,0($0)

eee:

110@00003000: $10 <= 0000000a

130@00003004: $31 <= 0000300c

130@00003008: \*00000000 <= 0000300c

5. M级Rt与W级jalr

jalr $t3,$t2

sw $t3,0($0)

110@00003000: $11 <= 00003008

110@00003004: \*00000000 <= 00003008

6. M级Rt与W级mflo mfhi

ori $t0,11

ori $t1,12

multu $t0,$t1

mflo $t2

sb $t2,0($0)

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@0000300c: $10 <= 00000084

290@00003010: \*00000000 <= 00000084

##### (2)暂停测试

测试目录：

一．Beq\_rs/rt

(1)E级cal\_r\_rd

(2) E级cal\_i\_rt

(3) E级load\_rt

(4) M级load\_rt

二．Cal\_r\_rs/rt

E级load\_rt

三．Cal\_i\_rs

E级load\_rt

四．load\_rs

E级load\_rt

五．store\_rs

E级load\_rt

六．jr\_rs

(1)E级cal\_r\_rd

(2) E级cal\_i\_rt

(3) E级load\_rt

(4) M级load\_rt

七. mult multu div divu

mflo mfhi mtlo mthi导致的暂停

一．Beq

(1) E段cal\_r

ori $s0,1

addu $s1,$s0,$0

beq $s0,$s1,eee

nop

eee:

nop

110@00003000: $16 <= 00000001

130@00003004: $17 <= 00000001

(2) E段cal\_i

ori $s0,1

ori $s1,$s0,2

beq $s1,$s0,eee

nop

eee:

nop

110@00003000: $16 <= 00000001

130@00003004: $17 <= 00000003

(3) E段load

ori $s0,$0,10

lw $s1,0($0)

beq $s1,$s0,eee

nop

eee:

nop

110@00003000: $16 <= 0000000a

130@00003004: $17 <= 00000000

(4) M段load

ori $s0,$0,10

lw $s1,0($0)

nop

beq $s1,$s0,eee

nop

eee:

addu $t0,$t0,$t0

110@00003000: $16 <= 0000000a

130@00003004: $17 <= 00000000

230@00003014: $ 8 <= 00000000

二．Cal\_r

E段load

lw $t2,0($0)

addu $t2,$t2,$t2

110@00003000: $10 <= 00000000

150@00003004: $10 <= 00000000

三．Cal\_i

E段load

lw $t2,0($0)

ori $t2,$t2,100

110@00003000: $10 <= 00000000

150@00003004: $10 <= 00000064

四．Load

E段load

lw $t2,0($0)

lw $t3,0($t2)

110@00003000: $10 <= 00000000

150@00003004: $11 <= 00000000

五．store

E段load

lw $t2,0($0)

sw $t3,0($t2)

110@00003000: $10 <= 00000000

130@00003004: \*00000000 <= 00000000

六．Jr

(1) E段cal\_r

ori $t3,$0,0x0000300c

addu $t2,$t2,$t3

jr $t2

nop

110@00003000: $11 <= 0000300c

130@00003004: $10 <= 0000300c

(2) E段cal\_i

ori $t3,$0,0x0000300c

ori $t2,$t3,0

jr $t2

nop

110@00003000: $11 <= 0000300c

130@00003004: $10 <= 0000300c

(3) E段load

ori $t3,$0,0x0000300c

sw $t3,0($0)

lw $t2,0($0)

jr $t2

nop

110@00003000: $11 <= 0000300c

110@00003004: \*00000000 <= 0000300c

150@00003008: $10 <= 0000300c

(4) M段load

ori $t3,$0,0x0000300c

sw $t3,0($0)

lw $t2,0($0)

nop

jr $t2

nop

110@00003000: $11 <= 0000300c

110@00003004: \*00000000 <= 0000300c

150@00003008: $10 <= 0000300c

七. 乘除相关

(1) busy

ori $t0,11

ori $t1,12

multu $t0,$t1

nop

mflo $t1

mfhi $t2

mtlo $t2

mthi $t1

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

290@00003010: $ 9 <= 00000084

310@00003014: $10 <= 00000000

(2)start

ori $t0,11

ori $t1,12

divu $t0,$t1

mflo $t1

mfhi $t2

mtlo $t2

mthi $t1

110@00003000: $ 8 <= 0000000b

130@00003004: $ 9 <= 0000000c

390@0000300c: $ 9 <= 00000000

410@00003010: $10 <= 0000000b

(3)测试乘除法

ori $t0,2

ori $t1,-3

mult $t0,$t1

mfhi $a0

mflo $a1

multu $t0,$t1

mthi $t0

mfhi $a2

mflo $a3

sw $a2,0($0)

sb $a3,2($0)

ori $t0,4

ori $t1,5

mtlo $t0

mult $t0,$t1

mfhi $a0

mflo $a1

multu $t0,$t1

mfhi $a2

mflo $a3

sh $a2,2($0)

sb $a3,13($0)

ori $t0,-3

ori $t1,-5

mult $t0,$t1

mfhi $a0

mflo $a1

multu $t0,$t1

mfhi $a2

mflo $a3

sw $a2,16($0)

sb $a3,15($0)

ori $t0,25

ori $t1,-5

div $t0,$t1

mfhi $a0

mflo $a1

divu $t0,$t1

mfhi $a2

mflo $a3

mthi $a3

sw $a2,4($0)

sw $a3,8($0)

ori $t0,2

ori $t1,5

div $t0,$t1

mfhi $a0

mflo $a1

divu $t0,$t1

mthi $a1

mtlo $a2

mfhi $a2

mflo $a3

sb $a2,1($0)

sw $a3,12($0)

ori $t0,-999

ori $t1,-5

div $t0,$t1

mfhi $a0

mthi $a0

mflo $a1

mtlo $a0

divu $t0,$t1

mfhi $a2

mflo $a3

sw $a2,0($0)

sb $a3,4($0)

110@00003000: $ 8 <= 00000002

130@00003004: $ 1 <= ffff0000

150@00003008: $ 1 <= fffffffd

170@0000300c: $ 9 <= fffffffd

330@00003014: $ 4 <= ffffffff

350@00003018: $ 5 <= fffffffa

530@00003024: $ 6 <= 00000002

550@00003028: $ 7 <= fffffffa

550@0000302c: \*00000000 <= 00000002

570@00003030: \*00000000 <= 00fa0002

610@00003034: $ 8 <= 00000006

630@00003038: $ 9 <= fffffffd

810@00003044: $ 4 <= ffffffff

830@00003048: $ 5 <= ffffffee

990@00003050: $ 6 <= 00000005

1010@00003054: $ 7 <= ffffffee

1010@00003058: \*00000000 <= 00050002

1030@0000305c: \*0000000c <= 0000ee00

1070@00003060: $ 1 <= ffff0000

1090@00003064: $ 1 <= fffffffd

1110@00003068: $ 8 <= ffffffff

1130@0000306c: $ 1 <= ffff0000

1150@00003070: $ 1 <= fffffffb

1170@00003074: $ 9 <= ffffffff

1330@0000307c: $ 4 <= 00000000

1350@00003080: $ 5 <= 00000001

1510@00003088: $ 6 <= fffffffe

1530@0000308c: $ 7 <= 00000001

1530@00003090: \*00000010 <= fffffffe

1550@00003094: \*0000000c <= 0100ee00

1590@00003098: $ 8 <= ffffffff

1610@0000309c: $ 1 <= ffff0000

1630@000030a0: $ 1 <= fffffffb

1650@000030a4: $ 9 <= ffffffff

1910@000030ac: $ 4 <= 00000000

1930@000030b0: $ 5 <= 00000001

2190@000030b8: $ 6 <= 00000000

2210@000030bc: $ 7 <= 00000001

2230@000030c4: \*00000004 <= 00000000

2250@000030c8: \*00000008 <= 00000001

2290@000030cc: $ 8 <= ffffffff

2310@000030d0: $ 9 <= ffffffff

2570@000030d8: $ 4 <= 00000000

2590@000030dc: $ 5 <= 00000001

2890@000030ec: $ 6 <= 00000001

2910@000030f0: $ 7 <= 00000000

2910@000030f4: \*00000000 <= 00050102

2930@000030f8: \*0000000c <= 00000000

2970@000030fc: $ 1 <= ffff0000

2990@00003100: $ 1 <= fffffc19

3010@00003104: $ 8 <= ffffffff

3030@00003108: $ 1 <= ffff0000

3050@0000310c: $ 1 <= fffffffb

3070@00003110: $ 9 <= ffffffff

3330@00003118: $ 4 <= 00000000

3370@00003120: $ 5 <= 00000001

3650@0000312c: $ 6 <= 00000000

3670@00003130: $ 7 <= 00000001

3670@00003134: \*00000000 <= 00000000

3690@00003138: \*00000004 <= 00000001

##### (3)异常中断测试

1.ADEL

(1).ktext 0x4180

mfc0 $k0, $14

addu $k0, $k0, 4

mtc0 $k0, $14

eret

.text

ori $28, $0, 0x0000

ori $29, $0, 0x0000

lui $8, 0x7000

lui $9, 0xf000

**lw $9,3($0)**

sub $10, $8,$9

or $10, $8, $9

(2).ktext 0x4180

mfc0 $k0, $14

addu $k0, $k0, 4

mtc0 $k0, $14

eret

.text

ori $28, $0, 0x0000

ori $29, $0, 0x0000

lui $8, 0x7000

lui $9, 0xf000

**lh $9,3($0)**

sub $10, $8,$9

or $10, $8, $9

(3).text

ori $28, $0, 0x0000

ori $29, $0, 0x0000

ori $8, 0x7fffffff

lui $9, 0x1

**add $10, $8,$9**

**lw $a0,0x1000($8)**

or $10, $8, $9

2.ADES

(1).text

ori $28, $0, 0x0000

ori $29, $0, 0x0000

ori $8, 0x7f00

lui $9, 0xf000

**sw $9,3($0)**

sub $10, $8,$9

or $10, $8, $9

(2).text

ori $28, $0, 0x0000

ori $29, $0, 0x0000

ori $8, 0x7f00

lui $9, 0xf000

**sh $9,1($0)**

sub $10, $8,$9

or $10, $8, $9

3.RI

在其他测试的机器码中插入ffffffff

4.Ov

.ktext 0x4180

mfc0 $k0, $14

sub $8,$8,$8

mtc0 $k0, $14

eret

.text

ori $28, $0, 0x0000

ori $29, $0, 0x0000

lui $8, 0x7fff

lui $9, 0x7fff

**add $10, $8,$9**

or $10, $8, $9

.text

ori $28, $0, 0x0000

ori $29, $0, 0x0000

lui $8, 0x7fff

lui $9, 0x7fff

**addi $10, $8,0x7fff0000**

or $10, $8, $9

.text

ori $28, $0, 0x0000

ori $29, $0, 0x0000

lui $8, 0x7fff

ori $8, $8, 0xffff

**addi $10, $8, 1**

ori $a0,$0,100

.text

ori $28, $0, 0x0000

ori $29, $0, 0x0000

lui $8, 0x7000

lui $9, 0xf000

**sub $10, $8,$9**

or $10, $8, $9

.text

ori $28, $0, 0x0000

ori $29, $0, 0x0000

ori $8, 0x7f00

lui $9, 0xf000

lw $9,0($8)

**sub $10, $8,$9**

or $10, $8, $9

.text

ori $28, $0, 0x0000

ori $29, $0, 0x0000

ori $8, 0x7f00

lui $9, 0xf000

sh $9,4($8)

**sub $10, $8,$9**

or $10, $8, $9

5.延迟槽

（1）.ktext 0x4180

mfc0 $1,$13

sub $9,$9,$9

eret

.text

ori $28, $0, 0x0000

ori $29, $0, 0x0000

ori $8, 0x7fffffff

ori $9, 0x1000

**j eee**

**add $10, $8,$9**

lw $a0,0x1000($8)

eee:

or $10, $8, $9

(2) .ktext 0x4180

mfc0 $1,$13

sub $9,$9,$9

eret

.text

ori $8, 0x7fffffff

ori $9, 0x1000

ori $t1 0x00007f00

ori $a0,0x0009

ori $a3,0xfc01

**beq $9,$8,eee**

**add $10,$8,$9**

ori $a1,2

sw $a1,4($t1)

eee:

sw $a0,0($t1)

（3）.ktext 0x4180

mfc0 $1,$13

sub $9,$9,$9

eret

.text

ori $8, 0x7fff0000

ori $9, 0x7fff0000

ori $a0,0x0009

ori $a3,0xfc01

**beq $9,$8,eee**

**add $10,$8,$9**

ori $a1,2

sw $a1,4($t1)

eee:

sw $a0,0($t1)

（4）.ktext 0x4180

mfc0 $1,$13

sub $8,$8,$8

sub $9,$9,$9

eret

.text

ori $8, 0x7fff0000

ori $9, 0x7fff0000

ori $a0,0x0009

ori $a3,0xfc01

**beq $9,$8,eee**

**add $10,$8,$9**

ori $a1,2

sw $a1,4($t1)

eee:

sw $a0,0($t1)

（5）.ktext 0x4180

mfc0 $1,$13

sub $9,$9,$9

eret

.text

ori $28, $0, 0x0000

ori $29, $0, 0x0000

ori $8, 0x7fffffff

ori $9, 0x1000

j eee

add $10, $8,$0

lw $a0,0x1000($8)

eee:

or $10, $8, $9

**j end**

**add $10,$8,$9**

end:

ori $a0,$0,0

(6) .ktext 0x4180

mfc0 $1,$13

sub $9,$9,$9

eret

.text

ori $28, $0, 0x0000

ori $29, $0, 0x0000

ori $8, 0x7fffffff

ori $9, 0x1000

j eee

add $10, $8,$0

lw $a0,0x1000($8)

eee:

or $10, $8, $9

**j end**

**sh $1,1($0)**

end:

ori $a0,$0,0

(7) .text

ori $28, $0, 0x0000

ori $29, $0, 0x0000

ori $8, 0x7fffffff

ori $9, 0x1000

j eee

add $10, $8,$0

lw $a0,0x1000($8)

eee:

or $10, $8, $9

**j end**

**sw $1,1($0)**

end:

ori $a0,$0,0

(8) .text

ori $28, $0, 0x0000

ori $29, $0, 0x0000

ori $8, 0x7fffffff

ori $9, 0x1000

j eee

add $10, $8,$0

lw $a0,0x1000($8)

eee:

or $10, $8, $9

**j end**

**lh $1,1($0)**

end:

ori $a0,$0,0

(9) .text

ori $28, $0, 0x0000

ori $29, $0, 0x0000

ori $8, 0x7fffffff

ori $9, 0x1000

j eee

add $10, $8,$0

lw $a0,0x1000($8)

eee:

or $10, $8, $9

**j end**

**lw $1,1($0)**

end:

ori $a0,$0,0

(10).text

ori $28, $0, 0x0000

ori $29, $0, 0x0000

ori $8, 0x7fffffff

ori $9, 0x1000

j eee

add $10, $8,$0

lw $a0,0x1000($8)

eee:

or $10, $8, $9

**j end**

**lhu $1,1($0)**

end:

ori $a0,$0,0

6.中断

（1）.ktext 0x4180

mfc0 $1,$13

sub $9,$9,$9

mtc0 $0,$12

eret

.text

ori $8, 0x7fffffff

ori $9, 0x1000

ori $t1 0x00007f00

ori $a0,0x0009

ori $a3,0xfc01

ori $a1,2

sw $a1,4($t1)

sw $a0,0($t1)

mtc0 $a3,$12

or $10, $8, $9

ori $28, $0, 0x0000

ori $29, $0, 0x0000

ori $28, $0, 0x0010

ori $29, $0, 0x0111

nop

nop

nop

（2）.ktext 0x4180

mfc0 $1,$13

sub $9,$9,$9

mtc0 $0,$12

eret

.text

ori $8, 0x7fffffff

ori $9, 0x1000

ori $t1 0x00007f00

ori $a0,0x0009

ori $a3,0xfc01

ori $a1,2

sw $a1,4($t1)

sw $a0,0($t1)

**add $11,$8,$9**

**mtc0 $a3,$12**

or $10, $8, $9

ori $28, $0, 0x0000

ori $29, $0, 0x0000

ori $28, $0, 0x0010

ori $29, $0, 0x0111

nop

nop

nop

（3）.ktext 0x4180

mfc0 $1,$13

sub $9,$9,$9

sub $29,$29,$29

mtc0 $0,$12

ori $v1,$0,0x00007f00

sw $a3,0($v1)

eret

.text

ori $8, 0x7fffffff

ori $9, 0x1000

add $11,$8,$9

ori $t1 0x00007f00

ori $a0,0x0009

ori $a3,0xfc01

ori $a1,1

sw $a1,4($t1)

sw $a0,0($t1)

mtc0 $a3,$12

nop

ori $a0,1111

j eee

ori $29, $0, 0x0111

eee:

or $10, $8, $9

ori $28, $0, 0x0000

ori $29, $0, 0x0000

ori $29,0x1000

lui $v1,1

j end

add $10,$8,$29

end:

lui $a0,1

lui $a2,2

7.乘除相关

（1）.ktext 0x4180

mfc0 $1,$13

sub $a0,$a0,$a0

eret

.text

ori $8, 0x7fffffff

ori $9, 0x1000

div $8,$9

lui $a0,0x7f00

add $a0,$a0,$a0

mthi $a0

mflo $s7

ori $a1,1

ori $v0,1

addu $a1,$v0,$v0

nop

ori $s0,11

（2）.ktext 0x4180

mfc0 $1,$13

mtc0 $0,$12

mflo $s2

mfhi $s1

ori $v1,$0,0x00007f00

sw $a3,0($v1)

eret

.text

ori $t1,0x00007f00

ori $a0,0x0009

ori $a3,0xfc01

ori $a1,3

sw $a1,4($t1)

sw $a0,0($t1)

mtc0 $a3,$12

mult $t1,$a3

mthi $a3

mfhi $a2

ori $29,0x1000

lui $v1,1

（3）mult 中断

.ktext 0x4180

mfc0 $1,$13

mtc0 $0,$12

mflo $s2

mfhi $s1

ori $v1,$0,0x00007f00

sw $a3,0($v1)

eret

.text

ori $t1,0x00007f00

ori $a0,0x0009

ori $a3,0xfc01

ori $a1,3

sw $a1,4($t1)

sw $a0,0($t1)

mtc0 $a3,$12

mult $t1,$a3

mthi $a3

mfhi $a2

ori $29,0x1000

lui $v1,1

(4) .ktext 0x4180

mfc0 $1,$13

mtc0 $0,$12

mflo $s2

mfhi $s1

ori $v1,$0,0x00007f00

sw $a3,0($v1)

eret

.text

ori $t1,0x00007f00

ori $a0,0x0009

ori $a3,0xfc01

ori $a1,4

sw $a1,4($t1)

sw $a0,0($t1)

mtc0 $a3,$12

mult $t1,$a3

mthi $a3

mfhi $a2

ori $29,0x1000

lui $v1,1

(5) .ktext 0x4180

mfc0 $1,$13

mtc0 $0,$12

mflo $s2

mfhi $s1

ori $v1,$0,0x00007f00

sw $a3,0($v1)

eret

.text

ori $t1,0x00007f00

mult $t1,$t1

ori $a0,0x0009

ori $a3,0xfc01

ori $a1,3

sw $a1,4($t1)

sw $a0,0($t1)

mtc0 $a3,$12

mult $t1,$a3

#mthi $a3

#mfhi $a2

ori $29,0x1000

lui $v1,1

（6）.ktext 0x4180

mfc0 $1,$13

mtc0 $0,$12

mflo $s2

mfhi $s1

ori $v1,$0,0x00007f00

sw $a3,0($v1)

eret

.text

ori $t1,0x00007f00

mult $t1,$t1

ori $a0,0x0009

ori $a3,0xfc01

ori $a1,8

sw $a1,4($t1)

sw $a0,0($t1)

mtc0 $a3,$12

ori $29,0x1000

mthi $a3

ori $a0,0x000b

sw $a0,0($t1)

mthi $a3

mfhi $a2

ori $29,0x1000

mult $t1,$t1

div $a2,$a2

lui $a0,1

110@00003000: $ 9 <= 00007f00

150@00003008: $ 4 <= 00000009

170@0000300c: $ 7 <= 0000fc01

190@00003010: $ 5 <= 00000008

270@00003020: $29 <= 00001000

310@00003028: $ 4 <= 0000000b

370@00003034: $ 6 <= 0000fc01

390@00003038: $29 <= 00001000

550@00004180: $ 1 <= 00000000

590@00004188: $18 <= 3f010000

610@0000418c: $17 <= 00000000

630@00004190: $ 3 <= 00007f00

870@00003044: $ 4 <= 00010000

#### 八．思考题

1. 我们计组课程一本参考书目标题中有“硬件/软件接口”接口字样，那么到底什么是“硬件/软件接口”？

硬件

目标：CPU与外设(被控对象)在硬件上连接构成一个有机整体

方法：I/0接口电路(接口、接口控制器)

软件

目标：控制设备工作方式，完成信息传送

方法：接口控制程序(或驱动程序)

硬软件接口就是计算机硬件与软件的交互手段，人类与电脑等信息机器或人类与程序之间的接口称为用户界面。电脑等信息机器硬件组件间的接口叫硬件接口。电脑等信息机器软件组件间的接口叫软件接口。硬件接口指的是两个硬件设备之间的连接方式。硬件接口既包括物理上的接口，还包括逻辑上的数据传送协议。软件不同部分之间的交互接口。通常就是所谓的API――应用程序编程接口，其表现的形式是源代码。

1. 在我们设计的流水线中，DM 处于 CPU 内部，请你考虑现代计算机中它的位置应该在何处。

在主存中，CPU通过高速缓存cache与主存交换数据。

1. BE 部件对所有的外设都是必要的吗？

不是，本次加的timer只能用lw,sw,用不到BE了。

1. 请开发一个主程序以及定时器的exception handler。整个系统完成如下功能：

定时器在主程序中被初始化为模式0；定时器倒计数至0产生中断；handler设置使能Enable为1从而再次启动定时器的计数器。2及3被无限重复。主程序在初始化时将定时器初始化为模式0，设定初值寄存器的初值为某个值，如100或1000。（注意，主程序可能需要涉及对CP0.SR的编程，推荐阅读过后文后再进行。）

.ktext 0x4180

mfc0 $1,$13

mflo $s2

mfhi $s1

ori $v1,$0,0x00007f00

sw $a0,0($v1)

eret

.text

ori $t1,0x00007f00

mult $t1,$t1

ori $a0,0x0009

ori $a3,0xfc01

ori $a1,2

sw $a1,4($t1)

sw $a0,0($t1)

mtc0 $a3,$12

mult $t1,$a3

ori $29,0x1000

lui $v1,1

lui $v0,2

lui $s1,3

lui $s0,4

1. 请查阅相关资料，说明鼠标和键盘的输入信号是如何被CPU知晓的？

设备实际上包括两部分接口控制器（也称为接口芯片）和设备主体，设备主体不直接与主机连接，而是通过接口控制器与主机连接。鼠标和键盘的输入信号相当于中断，当键盘、鼠标有信息时，产生一个中断然后中断例程会从端口读入数据到寄存器。CPU接收到中断请求之后进入中断处理程序获得鼠标键盘的信息。