Project5 Verilog 完成流水线处理器开发

一、顶层设计

1、顶层视图：

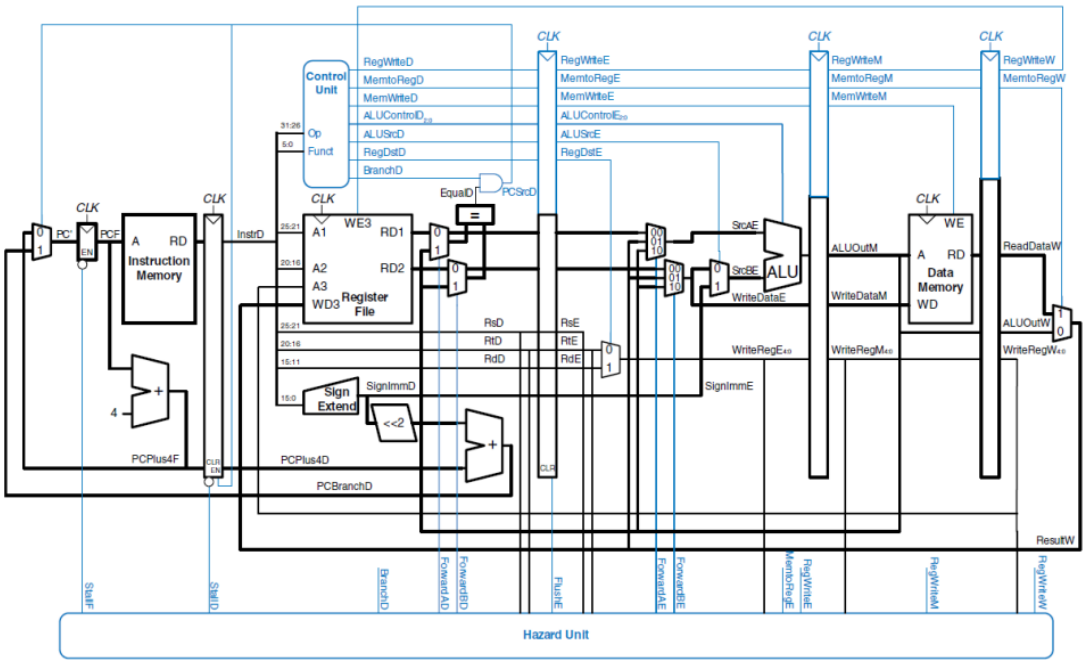

Figure 7.58 Pipelined processor with full hazard handling

2、支持指令集：

MIPS-lite2={ addu, subu, ori, lw, sw, beq, lui, j, jal, jr, nop }

3、顶层文件接口定义：

文件	模块接口定义

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

二、数据通路设计

1、数据通路架构：

	部件	描述	输入	输入来源				MUX	MUX控制信号	lw	sw	addu	subu	ori	lui	beq	j	jal	jr
F级功能部件	PC	程序计数器																	
	ADD4	完成PC+4		PC						PC	PC	PC	PC	PC	PC	PC	PC	PC	PC
	IM	指令存储器		PC						PC	PC	PC	PC	PC	PC	PC	PC	PC	PC
D级更新PC	PC			ADD4						ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	
D级流水线寄存器	IR@D	传递指令		IM						IM	IM	IM	IM	IM	IM	IM	IM	IM	IM
	PC4@D	下一条指令地址		ADD4												ADD4	ADD4	ADD4	
D级功能部件	RF	寄存器堆	A1	IR@D[rs]						IR@D[rs]	IR@D[rs]	IR@D[rs]	IR@D[rs]	IR@D[rs]	IR@D[rs]	IR@D[rs]			IR@D[rs]
			A2	IR@D[rt]							IR@D[rt]	IR@D[rt]	IR@D[rt]			IR@D[rt]			
	EXT	立即数扩展		IR@D[i16]						IR@D[i16]	IR@D[i16]			IR@D[i16]	IR@D[i16]				
	CMP	比较2个数	D1	RF.RD1												RF.RD1			
			D2	RF.RD2												RF.RD2			
	NPC	计算下条地址	PC4	PC4@D												PC4@D	PC4@D	PC4@D	
			I26	IR@D[i16]	IR@D[i26]											IR@D[i16]	IR@D[i26]	IR@D[i26]	
E级更新PC	PC			NPC	RF.RD1			MUX_PC	Pcsel							NPC	NPC	NPC	RF.RD1
E级流水线寄存器	IR@E	传递指令		IR@D						IR@D	IR@D	IR@D	IR@D	IR@D	IR@D			IR@D	
	PC4@E	下一条指令地址		PC4@D														PC4@D	
	RS@E	RF的RS值		RF.RD1						RF.RD1	RF.RD1	RF.RD1	RF.RD1	RF.RD1	RF.RD1				
	RT@E	RF的RT值		RF.RD2							RF.RD2	RF.RD2	RF.RD2						
	EXT@E	扩展后的立即数		EXT						EXT	EXT			EXT	EXT				
E级功能部件	ALU	算数逻辑运算	A	RS@E						RS@E	RS@E	RS@E	RS@E	RS@E	RS@E				
			B	RT@E	EXT@E			MUX_ALUb	ALU_bsel	EXT@E	EXT@E	RT@E	RT@E	EXT@E	EXT@E				
M级流水线寄存器	IR@M	传递指令		IR@E						IR@E	IR@E	IR@E	IR@E	IR@E	IR@E			IR@E	
	PC4@M	下一条指令地址		PC4@E														PC4@E	
	AO@M	ALU读出结果		ALU						ALU	ALU	ALU	ALU	ALU	ALU				
	RT@M	RF的RT值		RT@E							RT@E								
M级功能部件	DM	数据存储器	A	AO@M						AO@M	AO@M								
			WD	RT@M							RT@M								
W级流水线寄存器	IR@W	传递指令		IR@M						IR@M		IR@M	IR@M	IR@M	IR@M			IR@M	
	PC4@W	下一条指令地址		PC4@M														PC4@M	
	AO@W	ALU读出结果		AO@M								AO@M	AO@M	AO@M	AO@M				
	DR@W	DM读出结果		DM						DM									
W级功能部件	RF	寄存器堆	A3	IR@W[rd]	IR@W[rt]	0x1F		MUX_GPRwa	RF_WAsel	IR@W[rt]		IR@W[rd]	IR@W[rd]	IR@W[rt]	IR@W[rt]			0x1F	
			WD	AO@W	DR@W	EXT	PC4@W	MUX_GPRwd	RF_WDsel	DR@W		AO@W	AO@W	AO@W	AO@W			PC4@W	

		描述	lw	sw	addu	subu	ori	lui	beq	j	jal	jr	MUX	0	1	2	MUX控制信号
PC		程序计数器	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4|NPC	NPC	NPC	RF.RD1	MPC	ADD4	NPC	RF.RD1	PCsel
IM		指令存储器	PC	PC	PC	PC	PC	PC	PC	PC	PC	PC		PC			
ADD4		完成PC+4	PC	PC	PC	PC	PC	PC	PC	PC	PC			PC			
D级	IR	传递指令	IM	IM	IM	IM	IM	IM	IM	IM	IM	IM		IM			
	NPC	下一条指令地址	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4			ADD4			
RF	A1	第1个源寄存器编号	IR[rs]@D	IR[rs]@D	IR[rs]@D	IR[rs]@D	IR[rs]@D	IR[rs]@D	IR[rs]@D			IR[rs]@D		IR[rs]@D			
	A2	第2个源寄存器编号		IR[rt]@D	IR@D[rt]	IR@D[rt]			IR[rt]@D					IR[rt]@D			
EXT		立即数扩展	IR[i16]@D	IR[i16]@D			IR[i16]@D	IR[i16]@D						IR[i16]@D			
NPC	PC4	计算下条地址							PC4@D	PC4@D	PC4@D			PC4@D			
	I26								IR[i16]@D	IR[i26]@D	IR[i26]@D			IR[i26]@D			
CMP	D1	比较2个数							RF.RD1					RF.RD1			
	D2								RF.RD2					RF.RD2			
E级	V1	RF的第1个寄存器输出值	RF.RD1	RF.RD1	RF.RD1	RF.RD1	RF.RD1	RF.RD1						RF.RD1			
	V2	RF的第2个寄存器输出值		RF.RD2	RF.RD2	RF.RD2								RF.RD2			
	A1	第1个源寄存器编号	IR[rs]@D	IR[rs]@D	IR[rs]@D	IR[rs]@D	IR[rs]@D	IR[rs]@D						IR[rs]@D			
	A2	第2个源寄存器编号		IR[rt]@D	IR[rt]@D	IR[rt]@D								IR[rt]@D			
	A3	目的寄存器编号	IR[rt]@D		IR[rd]@D	IR[rd]@D	IR[rt]@D	IR[rt]@D			0x1F		MA3E	IR[rt]@D	IR[rd]@D	0x1F	RFA3sel
	E32	EXT的32位扩展结果	EXT	EXT			EXT	EXT						EXT			
	PC4	下一条指令地址									PC4@D			PC4@D			
ALU	A	算数逻辑运算	V1@E	V1@E	V1@E	V1@E	V1@E	V1@E						V1@E			
	B		E32@E	E32@E	V2@E	V2@E	E32@E	E32@E					MALUB	V2@E	E32@E		ALUBsel
M级	V2	RF的第2个寄存器输出值		V2@E										V2@E			
	A2	第2个源寄存器编号		A2@E										A2@E			
	AO	ALU读出结果	ALU	ALU	ALU	ALU	ALU	ALU						ALU			
	A3		A3@E		A3@E	A3@E	A3@E	A3@E			A3@E			A3@E			
	PC4	下一条指令地址									PC4@E			PC4@E			
DM	A	写入地址	AO@M	AO@M	AO@M	AO@M	AO@M	AO@M						AO@M			
	WD	写入值		V2@M										V2@M			
W级	A3	传递指令	A3@M		A3@M	A3@M	A3@M	A3@M			A3@M			A3@M			
	PC4	下一条指令地址									PC4@M			PC4@M			
	AO	ALU读出结果	AO@M		AO@M	AO@M	AO@M	AO@M						AO@M			
	DR	DM输出值	DM											DM			
RF	A3	写入寄存器编号	A3@W		A3@W	A3@W	A3@W	A3@W			A3@W			A3@W			
	WD	写入值	DR@W		AO@W	AO@W	AO@W	AO@M			PC4@W		MRFWD	AO@W	DR@W	PC4@W	RFWDsel

		lw	sw	addu	subu	ori	lui	beq	j	jal	jr	MUX	0	1	2
PC		ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4|NPC	NPC	NPC	RF.RD1	MPC	ADD4	NPC	MF_PC
CMP	D1							RF.RD1				MF_CMP_D1			
	D2							RF.RD2				MF_CMP_D2			
E级	A3	IR[rt]@D		IR[rd]@D	IR[rd]@D	IR[rt]@D	IR[rt]@D			0x1F		MA3E	IR[rt]@D	IR[rd]@D	0x1F
ALU	B	E32@E	E32@E	V2@E	V2@E	E32@E	E32@E					MALUB	MF_ALU_B	E32@E	
M级	V2		V2@E									MF_M_V2			
DM	WD		V2@M									MF_DM_WD			
RF	WD	DR@W		AO@W	AO@W	AO@W	AO@M			PC4@W		MRFWD	AO@W	DR@W	PC4@W

2、PC

（1）基本描述：

PC 用寄存器实现，具有复位功能。

起始地址：0x0000_3000。

（2）文件接口定义：

文件	模块接口定义
PC.v	module PC(Clock,Reset,Hold,NextPC,PC); input Clock, //clock input Reset, //reset input Hold, //stay the same input [31:0] NextPC, //next PC input output reg[31:0] PC //new PC

（3）功能定义：

序号	功能名称	功能描述
1	复位	当Reset有效且时钟上升沿来临时,PC被设置为0x0000_3000
2	更新PC的值	在时钟上升沿来临时，将NextPC写入到PC中
3	保持PC的值	当Reset无效、Hold有效且时钟上升沿来临时，PC保持不变

3、IM

（1）基本描述：

IM 容量为 4KB（32bit×1024 字）。

生成相应数量的 32 位寄存器的阵列。

采用$readmemh 指令将指令读入到 IM 中去。

（2）文件接口定义：

文件	模块接口定义
IM.v	module IM(Address,Instr); input [9:0] Address, //Instruction address output [31:0] Instr //Instruction

（3）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

4、GRF

（1）基本描述：

用具有写使能的寄存器实现，寄存器总数为 32 。

0 号寄存器的值始终保持为 0。其他寄存器初始值均为 0。

（2）文件接口定义：

文件	模块接口定义
IM.v	module GRF(Clock,Reset, RegWrite, ReadReg1, ReadReg2, WriteReg, WriteData, WPC, ReadData1, ReadData2); input Clock, //clock input Reset, //reset input RegWrite, //write enable input [4:0] ReadReg1, //read reg1 input [4:0] ReadReg2, //read reg2 input [4:0] WriteReg, //write reg input [31:0] WriteData , //write data input [31:0] WPC, // PC output [31:0] ReadData1, //read data1 output [31:0] ReadData2 //read data2

（3）功能定义：

序号	功能名称	功能描述
1	复位	Reset 信号有效时，所有寄存器存储的数值清零
2	读数据	读出 ReadReg1， ReadReg2 地址对应寄存器中所存储的数据到 ReadData1， ReadData2
3	写数据	当 RegWrite 有效且时钟上升沿来临时，将 WriteData 写入 WriteReg 所对应的寄存器中

5、EXT

（1）基本描述：

扩展

（2）文件接口定义：

文件	模块接口定义
EXT.v	module EXT(Imm_16,ExtOp,Imm_32); input [15:0] Imm_16, //input 16 bit Immediate input [1:0] ExtOp, //control the extension

	output reg [31:0] Imm_32 //output 32 bit Immediate

（3）功能定义：

序号	功能名称	功能描述
1	符号扩展	将 16 位输入数据符号扩展为 32 位
2	零扩展	将 16 位输入数据符号置为低 16 位，高 16 位置 0
3	加载至高位	将 16 位输入数据符号置为高 16 位，低 16 位置 0
4	符号扩展之后，左移两位	将 16 位输入数据符号扩展为 32 位之后，左移两位

6、NPC

（1）基本描述：

计算下一个 PC 的值

（2）文件接口定义：

文件	模块接口定义
NPC.v	module NPC(Instr,pc,rs,Zero,nPCOp,npc,pc_4); input [25:0] Instr, // Instruction input [31:0] pc, // PC input [31:0] rs, // $rs input Zero, // ALU Zero input [2:0] nPCOp, // control the operation output [31:0] npc, // next PC output [31:0] pc_4 // PC + 4

（3）功能定义：

序号	功能名称	功能描述
1	计算下一 PC	计算下一 PC 的值

7、ALU

（1）基本描述：

提供 32 位加、减、或运算等功能。

可以不支持溢出（不检测溢出）。

（2）文件接口定义：

文件	模块接口定义
ALU.v	module ALU(A,B,ALUOp,Zero.Result);

	input [31:0] A, // input data A input [31:0] B, // input data B input [3:0] ALUOp, // control the ALU output Zero, // Result == 0 output [31:0] Result // result

（3）功能定义：

序号	功能名称	功能描述
1	加运算	C = A + B
2	减运算	C = A - B
3	与运算	C = A & B

8、DM

（1）基本描述：

DM 容量为 4KB（32bit×1024 字）。

起始地址：0x00000000

（2）文件接口定义：

文件	模块接口定义
DM.v	module DM(Clock,Reset, Memwrite, Address, WriteData ,pc,addr, ReadData); input Clock, //clock input Reset, //reset input Memwrite, //memory write enable input [9:0] Address, // address input [31:0] WriteData, //write data input [31:0] pc, // PC input [31:0] addr, // 32-bit addr output[31:0] ReadData //read data

（3）功能定义：

序号	功能名称	功能描述
1	复位	Reset 信号有效时，所有储存器存储的数值清零
2	读数据	读出地址 Address 中所存储的数据到 ReadData
3	写数据	当 Memwrite 有效且时钟上升沿来临时，将 WriteData 写入地址 Address

三、控制器设计

1、基本描述：

控制单元基于指令的 opcode 字段（$Instr_{31:26}$）和 funct 字段（$Instr_{5:0}$）计算控制信号。采用分布式译码，可以只把指令在流水级之间传递，然后在不同流水级分别实例化一个同样的控制模块进行译码即可。

2、模块定义：

信号名	方向	描述
Op [5:0]	I	用于识别指令的功能
Func [5:0]	I	用于辅助 op 来识别指令
RegDst [1:0]	O	控制写入端地址选择 00：寄存器堆写入端地址选择 Rt 字段 01：寄存器堆写入端地址选择 Rd 字段 10：寄存器堆写入端地址选择 31 号寄存器
RegWrite	O	GRF 的写使能信号 0：无效 1：把数据写入寄存器堆中对应寄存器
ALUSrc	O	控制 ALU 的操作 0：ALU 输入端 B 选择寄存器堆输出 R[rt] 1：ALU 输入端 B 选择 extend(immediate)
MemWrite	O	DM 的写使能信号 0：无效 1：数据存储器 DM 写数据（输入）
MemtoReg [1:0]	O	控制数据从 ALU 读出还是从 DM 读出 00：寄存器堆写入端数据来自 ALU 输出 01：寄存器堆写入端数据来自 DM 输出 10：寄存器堆写入端数据来自 PC+4
ExtOp [1:0]	O	EXT 功能的选择信号 00：符号扩展 01：零扩展 10：加载至高位 11：符号扩展之后，左移两位
NPCOp [2:0]	O	下一 PC 的选择信号 000：PC = PC+4 001：执行 beq 分支指令 010：执行 j/jal 跳转指令 011：执行 jr 跳转指令
ALUOp [3:0]	O	ALU 功能的选择信号 0000：加法运算 0001：减法运算 0010：或运算

3、文件接口定义：

文件	模块接口定义
ctrl.v	module ctrl(Op,Func,RegDst,RegWrite,ALUSrc,MemWrite,MemReg, ExtOp,ALUOp,NPCOp); input [5:0] Op, input [5:0] Func, output [1:0] RegDst, output RegWrite, output ALUSrc, output MemWrite, output [1:0]MemtoReg, output [1:0] ExtOp, output [3:0] ALUOp, output [2:0] NPCOp

4 支持指令集：

(1)addu 指令：

① 功能：无符号加法，不考虑溢出

② 操作：GPR[rd] ← GPR[rs] + GPR[rt]

③ 编码： 000000[31:26] rs[25:11] rt[20:16] rd[15:11] 00000[10:6] 100001[5:0]

④ 控制信号：

RegDst	RegWrite	ALUSrc	MemWrite	MemtoReg	ExtOp	ALUOp	NPCOp
01	1	0	0	00	xx	0000	000

(2)subu 指令：

① 功能：无符号减法，不考虑溢出

② 操作：GPR[rd] ← GPR[rs] - GPR[rt]

③ 编码： 000000[31:26] rs[25:11] rt[20:16] rd[15:11] 00000[10:6]

100010[5:0]

④ 控制信号：

RegDst	RegWrite	ALUSrc	MemWrite	MemtoReg	ExtOp	ALUOp	NPCOp
01	1	0	0	00	xx	0001	000

(3)ori 指令：

① 功能：或立即数

② 操作：GPR[rt] ← GPR[rs] OR zero_extend(immediate)

③ 编码：001101[31:26] rs[25:11] rt[20:16] immediate[15:0]

④ 控制信号：

RegDst	RegWrite	ALUSrc	MemWrite	MemtoReg	ExtOp	ALUOp	NPCOp
00	1	1	0	00	01	0010	000

(4)lw 指令：

① 功能：加载字，从内存中读取 4 个字节

② 操作：GPR[rt] ← GPR[base] + sign_extend(immediate)

③ 编码：100011[31:26] base[25:11] rt[20:16] offset[15:0]

④ 控制信号：

RegDst	RegWrite	ALUSrc	MemWrite	MemtoReg	ExtOp	ALUOp	NPCOp
00	1	1	0	01	00	0000	000

(5)sw 指令：

①功能：存储字，向内存中写入 4 个字节

②操作：Addr ← GPR[base] + sign_extend(immediate)

Memory[Addr] ← GPR[rt]

③编码：101011[31:26] base[25:11] rt[20:16] offset[15:0]

④控制信号：

RegDst	RegWrite	ALUSrc	MemWrite	MemtoReg	ExtOp	ALUOp	NPCOp
xx	0	1	1	00	00	0000	000

(6)lui 指令：

①功能：立即数加载至最高位

②操作：GPR[rt] ← GPR[zero] + immediate || 0^{16}

③编码：001111[31:26] 00000[25:11] rt[20:16] immediate[15:0]

④控制信号：

RegDst	RegWrite	ALUSrc	MemWrite	MemtoReg	ExtOp	ALUOp	NPCOp
00	1	1	0	00	10	0000	000

(7)beq 指令：

①功能：当两个待比较寄存器相等时，跳转到分支地址

②操作：if (GPR[rs] == GPR[rt])

PC ← PC + 1 + sign_extend(offset || 0^{2})

else

PC ← PC + 1

③编码：000100[31:26] rs[25:11] rt[20:16] offset[15:0]

④控制信号：

RegDst	RegWrite	ALUSrc	MemWrite	MemtoReg	ExtOp	ALUOp	NPCOp
xx	0	0	0	00	00	0001	001

(8)j 指令：

①功能：跳转到地址

②操作：PC←PC31…28||instr_index||0^{2}

③编码：000011[31:26] instr_index[25:0]

④控制信号：

RegDst	RegWrite	ALUSrc	MemWrite	MemtoReg	ExtOp	ALUOp	NPCOp
xx	0	x	0	xx	xx	xxxx	010

(9)jal 指令：

①功能：跳转到地址，并将 PC+4 保存在 GPR[31]中

②操作：PC←PC31…28||instr_index||0^{2}

GPR[31]←PC+4

③编码：000011[31:26] insrt_index[25:0]

④控制信号：

RegDst	RegWrite	ALUSrc	MemWrite	MemtoReg	ExtOp	ALUOp	NPCOp
00	0	0	0	00	00	xxxx	000

(10) jr 指令：

①功能：跳转到寄存器

②操作：PC←GPR[rs]

③编码：000000[31:26] rs[25:11] 0[20:11] 00000[10:6] 001000[5:0]

④控制信号：

RegDst	RegWrite	ALUSrc	MemWrite	MemtoReg	ExtOp	ALUOp	NPCOp
00	0	0	0	00	00	xxxx	000

(11) nop 指令：

①功能：不执行任何操作

②操作：无

③编码：00000000000000000000000000000000 [31:0]

④控制信号：

RegDst	RegWrite	ALUSrc	MemWrite	MemtoReg	ExtOp	ALUOp	NPCOp
00	0	0	0	00	00	xxxx	000

四、数据冒险

Tuse：指令进入 D 级后，其后的某个功能部件再经过多少 cycle 就必须要使用寄存器的值

Tnew：位于 E 级及其后各级的指令，再经过多少周期能够产生要写入寄存器的结果

暂停：Tnew＞Tuse

转发：Tnew=0 && 指令不在 W 级 ||

	Tuse	
	rs	rt
addu	1	1
subu	1	1
ori	1	
lui		
lw	1	
sw	1	2
beq	0	0
j		
jr	0	
jal		
	{0,1}	{0,1,2}

指令	功能部件	Tuse		
		E	M	W
addu	ALU	1	0	0
subu	ALU	1	0	0
ori	ALU	1	0	0
lui	ALU	1	0	0
lw	DM	2	1	0
sw				
beq				
j				
jr				
jal	PC	0	0	0

Tnew≤Tuse

rs策略矩阵

Tnew \ Tuse	E			M			W		
	ALU	DM	PC	ALU	DM	PC	ALU	DM	PC
	1	2	0	0	1	0	0	0	0
0	S	S	F	F	S	F	F	F	F
1	F	S	F	F	F	F	F	F	F

rt策略矩阵

Tnew \ Tuse	E			M			W		
	ALU	DM	PC	ALU	DM	PC	ALU	DM	PC
	1	2	0	0	1	0	0	0	0
0	S	S	F	F	S	F	F	F	F
1	F	S	F	F	F	F	F	F	F
2	F	F	F	F	F	F	F	F	F

转发表格

流水级	源寄存器	指令	转发MUX	控制信号		E级	M级			W级			
						jal 0/31	cal_r 0/rd	cal_i 0/rt	jal 0/31	cal_r 0/rd	cal_i 0/rt	load 0/rt	jal 0/31
D	rs	beq,jr	MFRSD	RSDsel	RF.RD1	PC_E+8	AO	AO	PC_M+8	WD	WD	WD	WD
	rt	beq	MFRTD	RTDsel	RF.RD2	PC_E+8	AO	AO	PC_M+8	WD	WD	WD	WD
E	rs	cal_r,cal_i, load,store	MFRSE	RSEsel	RS@E		AO	AO	PC_M+8	WD	WD	WD	WD
	rt	cal_r,store	MFRTE	RTEsel	RT@E		AO	AO	PC_M+8	WD	WD	WD	WD
M	rt	strore	MFRTM	RTMsel	RT@M					WD	WD	WD	WD

暂停表格

D级指令			E级(Tnew)			M级(Tnew)			W级(Tnew)		
指令	源寄存器	Tuse	cal_r 1/rd	cal_i 1/rt	load 2/rt	cal_r 0/rd	cal_i 0/rt	load 1/rt	cal_r 0/rd	cal_i 0/rt	load 0/rt
beq	rs/rt	0	S	S	S			S			
cal_r	rs/rt	1			S						
cal_i	rs	1			S						
load	rs	1			S						
strore	rs	1			S						
	rt	2			S						
jr	rs	0	S	S	S			S			

五、测试

1、首先测试 ori、lui、addu、subu、nop

```
ori $0,$0,100
ori $1,$0,1
ori $2,$1,2
ori $3,$2,3
ori $4,$2,-4
ori $5,$4,-5
ori $6,$5,6
ori $7,$6,7
nop

lui $8,8
lui $9,9
lui $10,10
lui $11,0xa
lui $12,0xb
lui $13,0xc
nop

addu   $14,$1,$2
addu   $15,$2,$3
addu   $16,$14,$15
addu   $17,$15,$16
addu   $18,$16,$17
addu   $19,$19,$7
addu   $20,$20,$8
nop

subu   $21,$9,$10
subu   $22,$10,$11
subu   $23,$21,$22
subu   $24,$22,$23
subu   $25,$25,$14
subu   $26,$26,$15
nop
34000064
34010001
34220002
34430003
34440004
34850005
34a60006
```

0	0x00000000
1	0x00000001
2	0x00000003
3	0x00000003
4	0x00000007
5	0x00000007
6	0x00000007
7	0x00000007
8	0x00080000
9	0x00090000
10	0x000a0000
11	0x000a0000
12	0x000b0000
13	0x000c0000
14	0x00000004
15	0x00000006
16	0x0000000a
17	0x00000010
18	0x0000001a
19	0x00000007
20	0x00080000
21	0xffff0000
22	0x00000000
23	0xffff0000
24	0x00010000
25	0xfffffffc
26	0xfffffffa
27	0x00000000

```
34c70007
00000000
3c080008
3c090009
3c0a000a
3c0b000a
3c0c000b
3c0d000c
00000000
00227021
00437821
01cf8021
01f08821
02119021
02679821
0288a021
00000000
012aa823
014bb023
02b6b823
02d7c023
032ec823
034fd023
00000000
```

2、然后测试全部指令

```
.data
arr:   .space 200

.text
ori $1,$0,1
ori $2,$0,2
ori $3,$0,0
ori $4,$0,4
ori $5,$0,10
jal swlw
ori $20,$0,20

j   jump
ori $22,$0,22

swlw:  beq $5,$0,end
    ori $21,$0,21
    sw  $2,arr($3)
```

0	0x00000000
1	0x00000001
2	0x00000800
3	0x00000028
4	0x00000004
5	0x00000000
6	0x00000000
7	0x00000000
8	0x00000000
9	0x00000000
10	0x00000400
11	0x00000000
12	0x00000000
13	0x00000000
14	0x00000000
15	0x00000000
16	0x00000000
17	0x00000000
18	0x00000000
19	0x00000000
20	0x00000014
21	0x00000015
22	0x00000016
23	0x00003054
24	0x00000000
25	0x00000019
26	0x0000001a
27	0x0000001b
28	0x00001800
29	0x00002ffc
30	0x00000000
31	0x0000301c

```
    lw  $10,arr($3)
    addu   $2,$2,$2
    addu   $3,$3,$4
    subu   $5,$5,$1
    j   swlw
    ori $27,$0,27

end:   jr  $31

jump:  ori $23,$0,0x3054
    jr  $23
    ori $25,$0,25

jrjr:  ori $26,$0,26


34010001
34020002
34030000
34040004
3405000a
0c000c09
34140014
08000c12
34160016
10a00007
34150015
ac620000
8c6a0000
00421021
00641821
00a12823
08000c09
341b001b
03e00008
34173054
02e00008
34190019
341a001a
```

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x00000000	0x00000002	0x00000004	0x00000008	0x00000010	0x00000020	0x00000040	0x00000080	0x00000100
0x00000020	0x00000200	0x00000400	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

六、思考题

1、在本实验中你遇到了哪些不同指令组合产生的冲突？你又是如何解决的？相应的测试样例是什么样的？请有条理的罗列出来。（非常重要）

用例编号	测试类型	前序指令	冲突位置	冲突寄存器	测试序列
1	R-M-RS	R	MEM	rs	addu $3,$1,$2 addu $5,$3,$4
2	R-M-RT	R	MEM	rt	addu $3,$1,$2 addu $5,$4,$3

3	R-W-RS	R	WB	rs	addu $3,$1,$2
					nop
					addu $5,$3,$4

4	R-W-RT	R	WB	rt	addu $3,$1,$2
					nop
					addu $5,$4,$3

5	I-M-RS	I	MEM	rs	ori $1,$0,20 addu $3,$1,$2
6	I-M-RT	I	MEM	rt	ori $1,$0,20 addu $3,$2,$1
7	I-W-RS	I	WB	rs	ori $1,$0,20 nop addu $3,$2,$2
8	I-W-RT	I	WB	rt	ori $1,$0,20 nop addu $3,$2,$3
9	lui-M-RS	lui	MEM	rs	lui $1,20 addu $3,$1,$2
10	lui-M-RT	lui	MEM	rt	lui $1,20 addu $3,$1,$2
11	lui-W-RS	lui	WB	rs	lui $1,20 nop addu $3,$1,$2
12	lui-W-RT	lui	WB	rt	lui $1,20 nop addu $3,$2,$1
13	L-M-RS	load	MEM	rs	lw $1,0($2) addu $3,$1,$2
14	L-M-RT	load	MEM	rt	lw $1,0($2) addu $3,$2,$1

15	L-W-RS	load	WB	rs	lw $1,0($2)
					nop
					addu $3,$1,$2

16	L-W-RT	load	WB	rt	lw $1,0($2)
					nop
					addu $3,$2,$1

17	J-M-RS	jal	MEM	rs	jal loop addu $2,$31,$1
18	J-M-RT	jal	MEM	rt	jal loop addu $2,$1,$31
19	J-W-RS	jal	WB	rs	jal loop nop addu $2,$31,$1
20	J-W-RT	jal	WB	rt	jal loop nop addu $2,$1,$31
21	R-E-RS	R	EXE	rs	addu $3,$1,$2 jr $3
22	R-M-RS	R	MEM	rs	addu $3,$1,$2 nop jr $3
23	R-W-RS	R	WB	rs	addu $3,$1,$2 nop nop jr $3
24	I-E-RS	I	EXE	rs	ori $1,$0,20 jr $1
25	I-M-RS	I	MEM	rs	ori $1,$0,20 nop jr $1
26	I-W-RS	I	WB	rs	ori $1,$0,20 nop nop jr $1
27	lui-E-RS	lui	EXE	rs	lui $1,20 jr $1
28	lui-M-RS	lui	MEM	rs	lui $1,20 nop jr $1
29	lui-W-RS	lui	WB	rs	lui $1,20 nop nop jr $1
30	L-E-RS	load	EXE	rs	lw $1,0($2) jr $1

31	L-M-RS	load	MEM	rs	lw $1,0($2) noo jr $1
32	L-W-RS	load	WB	rs	lw $1,0($2) noo nop jr $1

	部件	描述	输入	输入来源				MUX	MUX控制信号	lw	sw	addu	subu	ori	lui	beq	j	jal	jr
F级功能部件	PC	程序计数器																	
	ADD4	完成PC+4		PC						PC	PC	PC	PC	PC	PC	PC	PC	PC	PC
	IM	指令存储器		PC						PC	PC	PC	PC	PC	PC	PC	PC	PC	PC
D级更新PC	PC			ADD4						ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	
D级流水线寄存器	IR@D	传递指令		IM						IM	IM	IM	IM	IM	IM	IM	IM	IM	IM
	PC4@D	下一条指令地址		ADD4												ADD4	ADD4	ADD4	
D级功能部件	RF	寄存器堆	A1	IR@D[rs]						IR@D[rs]	IR@D[rs]	IR@D[rs]	IR@D[rs]	IR@D[rs]	IR@D[rs]	IR@D[rs]			IR@D[rs]
			A2	IR@D[rt]							IR@D[rt]	IR@D[rt]	IR@D[rt]			IR@D[rt]			
	EXT	立即数扩展		IR@D[i16]						IR@D[i16]	IR@D[i16]			IR@D[i16]	IR@D[i16]				
	CMP	比较2个数	D1	RF.RD1												RF.RD1			
			D2	RF.RD2												RF.RD2			
	NPC	计算下条地址	PC4	PC4@D												PC4@D	PC4@D	PC4@D	
			I26	IR@D[i16]	IR@D[i26]											IR@D[i16]	IR@D[i26]	IR@D[i26]	
E级更新PC	PC			NPC	RF.RD1			MUX_PC	Pcsel							NPC	NPC	NPC	RF.RD1
E级流水线寄存器	IR@E	传递指令		IR@D						IR@D	IR@D	IR@D	IR@D	IR@D	IR@D			IR@D	
	PC4@E	下一条指令地址		PC4@D														PC4@D	
	RS@E	RF的RS值		RF.RD1						RF.RD1	RF.RD1	RF.RD1	RF.RD1	RF.RD1	RF.RD1				
	RT@E	RF的RT值		RF.RD2							RF.RD2	RF.RD2	RF.RD2						
	EXT@E	扩展后的立即数		EXT						EXT	EXT			EXT	EXT				
E级功能部件	ALU	算数逻辑运算	A	RS@E						RS@E	RS@E	RS@E	RS@E	RS@E	RS@E				
			B	RT@E	EXT@E			MUX_ALUb	ALU_bsel	EXT@E	EXT@E	RT@E	RT@E	EXT@E	EXT@E				
M级流水线寄存器	IR@M	传递指令		IR@E						IR@E	IR@E	IR@E	IR@E	IR@E	IR@E			IR@E	
	PC4@M	下一条指令地址		PC4@E														PC4@E	
	AO@M	ALU读出结果		ALU						ALU	ALU	ALU	ALU	ALU	ALU				
	RT@M	RF的RT值		RT@E							RT@E								
M级功能部件	DM	数据存储器	A	AO@M						AO@M	AO@M								
			WD	RT@M							RT@M								
W级流水线寄存器	IR@W	传递指令		IR@M						IR@M		IR@M	IR@M	IR@M	IR@M			IR@M	
	PC4@W	下一条指令地址		PC4@M														PC4@M	
	AO@W	ALU读出结果		AO@M								AO@M	AO@M	AO@M	AO@M				
	DR@W	DM读出结果		DM						DM									
W级功能部件	RF	寄存器堆	A3	IR@W[rd]	IR@W[rt]	0x1F		MUX_GPRwa	RF_WAsel	IR@W[rt]		IR@W[rd]	IR@W[rd]	IR@W[rt]	IR@W[rt]			0x1F	
			WD	AO@W	DR@W	EXT	PC4@W	MUX_GPRwd	RF_WDsel	DR@W		AO@W	AO@W	AO@W	AO@W			PC4@W	

		描述	lw	sw	addu	subu	ori	lui	beq	j	jal	jr	MUX	0	1	2	MUX控制信号
PC		程序计数器	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4|NPC	NPC	NPC	RF.RD1	MPC	ADD4	NPC	RF.RD1	PCsel
IM		指令存储器	PC	PC	PC	PC	PC	PC	PC	PC	PC	PC		PC			
ADD4		完成PC+4	PC	PC	PC	PC	PC	PC	PC	PC	PC			PC			
D级	IR	传递指令	IM	IM	IM	IM	IM	IM	IM	IM	IM	IM		IM			
	NPC	下一条指令地址	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4			ADD4			
RF	A1	第1个源寄存器编号	IR[rs]@D	IR[rs]@D	IR[rs]@D	IR[rs]@D	IR[rs]@D	IR[rs]@D	IR[rs]@D			IR[rs]@D		IR[rs]@D			
	A2	第2个源寄存器编号		IR[rt]@D	IR@D[rt]	IR@D[rt]			IR[rt]@D					IR[rt]@D			
EXT		立即数扩展	IR[i16]@D	IR[i16]@D			IR[i16]@D	IR[i16]@D						IR[i16]@D			
NPC	PC4	计算下条地址							PC4@D	PC4@D	PC4@D			PC4@D			
	I26								IR[i16]@D	IR[i26]@D	IR[i26]@D			IR[i26]@D			
CMP	D1	比较2个数							RF.RD1					RF.RD1			
	D2								RF.RD2					RF.RD2			
E级	V1	RF的第1个寄存器输出值	RF.RD1	RF.RD1	RF.RD1	RF.RD1	RF.RD1	RF.RD1						RF.RD1			
	V2	RF的第2个寄存器输出值		RF.RD2	RF.RD2	RF.RD2								RF.RD2			
	A1	第1个源寄存器编号	IR[rs]@D	IR[rs]@D	IR[rs]@D	IR[rs]@D	IR[rs]@D	IR[rs]@D						IR[rs]@D			
	A2	第2个源寄存器编号		IR[rt]@D	IR[rt]@D	IR[rt]@D								IR[rt]@D			
	A3	目的寄存器编号	IR[rt]@D		IR[rd]@D	IR[rd]@D	IR[rt]@D	IR[rt]@D			0x1F		MA3E	IR[rt]@D	IR[rd]@D	0x1F	RFA3sel
	E32	EXT的32位扩展结果	EXT	EXT			EXT	EXT						EXT			
	PC4	下一条指令地址									PC4@D			PC4@D			
ALU	A	算数逻辑运算	V1@E	V1@E	V1@E	V1@E	V1@E	V1@E						V1@E			
	B		E32@E	E32@E	V2@E	V2@E	E32@E	E32@E					MALUB	V2@E	E32@E		ALUBsel
M级	V2	RF的第2个寄存器输出值		V2@E										V2@E			
	A2	第2个源寄存器编号		A2@E										A2@E			
	AO	ALU读出结果	ALU	ALU	ALU	ALU	ALU	ALU						ALU			
	A3		A3@E		A3@E	A3@E	A3@E	A3@E			A3@E			A3@E			
	PC4	下一条指令地址									PC4@E			PC4@E			
DM	A	写入地址	AO@M	AO@M	AO@M	AO@M	AO@M	AO@M						AO@M			
	WD	写入值		V2@M										V2@M			
W级	A3	传递指令	A3@M		A3@M	A3@M	A3@M	A3@M			A3@M			A3@M			
	PC4	下一条指令地址									PC4@M			PC4@M			
	AO	ALU读出结果	AO@M		AO@M	AO@M	AO@M	AO@M						AO@M			
	DR	DM输出值	DM											DM			
RF	A3	写入寄存器编号	A3@W		A3@W	A3@W	A3@W	A3@W			A3@W			A3@W			
	WD	写入值	DR@W		AO@W	AO@W	AO@W	AO@M			PC4@W		MRFWD	AO@W	DR@W	PC4@W	RFWDsel