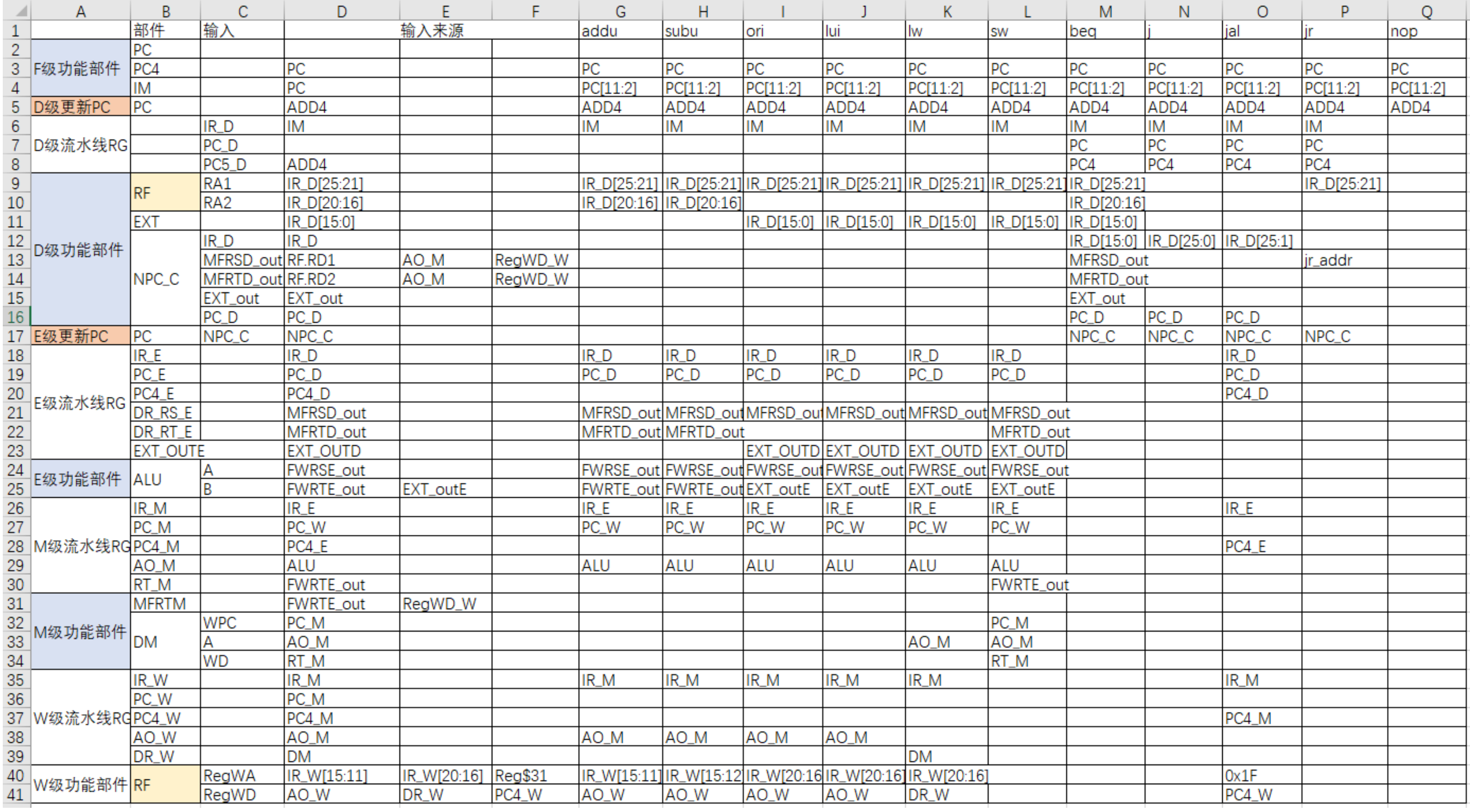
verilog 单周期 CPU 设计文档

一、 设计与测试说明

1. 处理器应支持的指令集为：MIPS-lite2={ addu, subu, ori, lw, sw, beq, lui, j, jal, jr, nop }

2. 处理器为流水线设计。

二、 数据通路设计



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1		部件	输入		输入来源		addu	subu	ori	lui	lw	sw	beq	j	jal	jr	nop
2	F级功能部件	PC															
3		PC4		PC			PC	PC	PC	PC	PC	PC	PC	PC	PC	PC	PC
4		IM		PC			PC[11:2]	PC[11:2]	PC[11:2]	PC[11:2]	PC[11:2]	PC[11:2]	PC[11:2]	PC[11:2]	PC[11:2]	PC[11:2]	PC[11:2]
5	D级更新PC	PC		ADD4			ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4	ADD4
6	D级流水线RG		IR_D	IM			IM	IM	IM	IM	IM	IM	IM	IM	IM	IM	
7			PC_D										PC	PC	PC	PC	
8			PC5_D	ADD4									PC4	PC4	PC4	PC4	
9	D级功能部件	RF	RA1	IR_D[25:21]			IR_D[25:21]	IR_D[25:21]	IR_D[25:21]	IR_D[25:21]	IR_D[25:21]	IR_D[25:21]	IR_D[25:21]			IR_D[25:21]	
10			RA2	IR_D[20:16]			IR_D[20:16]	IR_D[20:16]					IR_D[20:16]				
11		EXT		IR_D[15:0]					IR_D[15:0]	IR_D[15:0]	IR_D[15:0]	IR_D[15:0]	IR_D[15:0]				
12		NPC_C	IR_D	IR_D									IR_D[15:0]	IR_D[25:0]	IR_D[25:1]		
13			MFRSD_out	RF.RD1	AO_M	RegWD_W							MFRSD_out			jr_addr	
14			MFRTD_out	RF.RD2	AO_M	RegWD_W							MFRTD_out				
15			EXT_out	EXT_out									EXT_out				
16			PC_D	PC_D									PC_D	PC_D	PC_D		
17	E级更新PC	PC	NPC_C	NPC_C									NPC_C	NPC_C	NPC_C	NPC_C	
18	E级流水线RG	IR_E		IR_D			IR_D	IR_D	IR_D	IR_D	IR_D	IR_D			IR_D		
19		PC_E		PC_D			PC_D	PC_D	PC_D	PC_D	PC_D	PC_D			PC_D		
20		PC4_E		PC4_D											PC4_D		
21		DR_RS_E		MFRSD_out			MFRSD_out	MFRSD_out	MFRSD_out	MFRSD_out	MFRSD_out	MFRSD_out					
22		DR_RT_E		MFRTD_out			MFRTD_out	MFRTD_out				MFRTD_out					
23		EXT_OUTE		EXT_OUTD					EXT_OUTD	EXT_OUTD	EXT_OUTD	EXT_OUTD					
24	E级功能部件	ALU	A	FWRSE_out			FWRSE_out	FWRSE_out	FWRSE_out	FWRSE_out	FWRSE_out	FWRSE_out					
25			B	FWRTE_out	EXT_outE		FWRTE_out	FWRTE_out	EXT_outE	EXT_outE	EXT_outE	EXT_outE					
26	M级流水线RG	IR_M		IR_E			IR_E	IR_E	IR_E	IR_E	IR_E	IR_E			IR_E		
27		PC_M		PC_W			PC_W	PC_W	PC_W	PC_W	PC_W	PC_W					
28		PC4_M		PC4_E											PC4_E		
29		AO_M		ALU			ALU	ALU	ALU	ALU	ALU	ALU					
30		RT_M		FWRTE_out								FWRTE_out					
31	M级功能部件	MFRTM		FWRTE_out	RegWD_W												
32		DM	WPC	PC_M								PC_M					
33			A	AO_M							AO_M	AO_M					
34			WD	RT_M								RT_M					
35	W级流水线RG	IR_W		IR_M			IR_M	IR_M	IR_M	IR_M	IR_M				IR_M		
36		PC_W		PC_M													
37		PC4_W		PC4_M											PC4_M		
38		AO_W		AO_M			AO_M	AO_M	AO_M	AO_M							
39		DR_W		DM							DM						
40	W级功能部件	RF	RegWA	IR_W[15:11]	IR_W[20:16]	Reg$31	IR_W[15:11]	IR_W[15:12	IR_W[20:16	IR_W[20:16]	IR_W[20:16]				0x1F		
41			RegWD	AO_W	DR_W	PC4_W	AO_W	AO_W	AO_W	AO_W	DR_W				PC4_W		

三、 控制器设计

Controller 模块整体进行设计，输入指令，输出指令相应的信号。mips.v 中进行模块实例化和连接时，在每个流水级分别输出该流水级需要的信号。STALL 和 FORWARD 单独设计。

1. Controller 模块

```
CONTROLLER(INSTR,ALU_OP,ALU_Src,EXT_OP,RegDst,RegWrite,MemtoReg,b_pc,j_pc,NPC_sel,MemWrite,cal_r,cal_i,load,store,jal,jr);
```

其中 ALU_OP，RegWrite，MemWrite 等信号控制 ALU，GRF，DM 等功能部件的功能以及多选器对于输入数据和地址的选择。b_pc，j_pc，cal_r 等信号用于识别指令的类型。jal，jr 信号识别具体指令 jal，jr.

Table 1 指令对应信号值

	addu	subu	ori	sw	lw	lui	beq	j	jal	jr
IR[31:26]	000000	000000	001101	101011	100011	001111	000100	000010	000011	000000
IR[5:0]	100001	100011	x	x	x	x	x	x	x	001000
ALU_Src	0	0	1	1	1	1	x	x	x	x
ALU_OP	000	001	010	000	000	000	x	x	x	x
RegDst	01	01	00	x	00	00	x	x	10	xx
MemtoReg	00	00	00	xx	01	00	xx	xx	10	Xx
EXT_op	x	x	00	10	10	01	11	x	x	x
ALU_Src	00	00	01	01	01	01	xx	xx	xx	xx
NPC_sel	x	x	x	x	x	x	0	01	01	10
RegWrite	1	1	1	0	1	1	0	0	1	0
MemWrite	0	0	0	1	0	0	0	0	0	0
b_pc	0	0	0	0	0	0	1	0	0	0
j_pc	0	0	0	0	0	0	0	1	1	1
cal_i	0	0	1	1	1	1	0	0	0	0
cal_r	1	1	0	0	0	0	0	0	0	0
load	0	0	0	0	1	0	0	0	0	0
store	0	0	0	1	0	0	0	0	0	0
jal	0	0	0	0	0	0	0	0	1	0
jr	0	0	0	0	0	0	0	0	0	1

Table 2 部分信号

信号	含义	值	选择/功能	对应指令
RegDst	选择写入寄存器的地址	0	rt	lui, ori, lw
		1	rd	addu, subu
		2	$31	jal
MemtoReg	选择写入寄存器的数据	0	ALU_out	lui, ori, addu, subu
		1	DM_RD	lw
		2	PC+4(PC+8)	jal
ALU_OP	ALU 运算符	0	加	addu, lw, sw, lui
		1	减	subu
		2	或	ori
ALU_Src	选择输入 ALU_B 数据	0	RF_RD2	addu, subu
		1	EXT_out	lui, ori, lw, sw
EXT_OP	扩展符号	0	zero_ext	ori
		1	imm 移至高位	lui
		2	sign_ext	lw, sw
		3	beq_ext	beq
NPC_sel	选择 NPC	0	PC+4+EXT_out	beq
		1	PC[31:28] || IR[25:21] ||00	jal, j
		2	RF_RD1	jr

2. STALL 模块

根据 Tnew 和 Tuse 分析需要暂停的情况，暂停序列有： 1.

1. beq_D(rs/rt)-cal_rD(rd)/cal_iD(rt)/loadE(rt)/loadM(rt)

2. cal_rD(rs/rt)-loadM(rt)， cal_iD(rs)-loadE(rt)

3. loadD(rs)-loadE(rt)

4. storeD(rs)-loadE(rt)

5. jr_D(rs)-cal_rD(rd)/cal_iD(rt)/loadE(rt)/loadM(rt)

3. FORWARD 模块

直接根据流水级指令由分布式控制器输出的 RegWrite， RegDst， MemtoReg 信号，再设置流水级多选器 RegWA_M 和 RegWA_W，输出流水级写寄存器地址 M_WA

和 W_WA.根据寄存器写入地址和读地址来判断转发情况，避免转发多选器信号定义过于复杂。

Table 3 转发信号

转发信号	取值	条件	数据
FWRSD	0	else	RF_RD1
	1	RegWriteM && M_WA!=0 && D_rs==M_WA	AO_M
	2	RegWriteW && W_WA!=0 && D_rs==W_WA	RegWD_W
FWRTD	0	else	RF_RD2
	1	RegWriteM && M_WA!=0 && D_rt==M_WA	AO_M
	2	RegWriteW && W_WA!=0 && D_rt==W_WA	RegWD_W
FWRSE	0	else	RS_E
	1	RegWriteM && M_WA!=0 && E_rs==M_WA	AO_M
	2	RegWriteW && W_WA!=0 && E_rs==W_WA	RegWD_W
FWRTE	0	else	RT_E
	1	RegWriteM && M_WA!=0 && E_rt==M_WA	AO_M
	2	RegWriteW && W_WA!=0 && E_rt==W_WA	RegWD_W
FWRTM	0	else	RT_M
	1	loadW && W_WA!=0 && M_rt==W_WA	RegWD_W

除了 ID 级需要用到寄存器数据的 beq， jr 指令，对于下一级需要用到寄存器值的指令比如 cal_r 也进行了转发，再将转发的结果传入流水线寄存器作为 RF_RD1 和 RF_RD2. 此处考虑到以下特殊的几点：

1. loadM 且与 D 级指令数据冲突，以 cal_rD 为例，转发到 ID 的数据为 AO_M，并不符合 cal_r 读寄存器的要求，但当 cal_r 到达 E 级时，对于 ALU_A 和 ALU_B 会进行重新选择，发生冲突的寄存器转发信号为 2，实际参加运算的数据为将要写入寄存器的数据。
2. 在一些满足暂停条件的情况下也满足转发条件，但因 pc_en 的优先级较高，并不会以错误的转发数据进行运算，不影响最终结果。以 jr_D 和 loadM 且数据冲突为例，转发到 ID 的 RF_RD1 的数据为 AO_M， 但此时满足了暂停条件，暂停条件的优先级较高，PC 停止计数，下一时钟周期 jr_D && loadW，转发到 ID 的数据为 RegWD_W.
3. FWRTM 写寄存器只考虑 loadW，因为其他写寄存器的指令都在前一时钟周期经 ALU 得出了结果，且转发至 EX 做为 RF_RD2E.之后在下一时钟周期存入 M 流水级寄存器。

四、 测试程序

Table 4 addu **冲突测试**

测试类型	前序指令	冲突寄存器	测试序列	实际结果	初始化
r-M-rs	subu	rs	subu $1, $2, $3	430@00003044: $ 1 <= ffff0000	lui $1,1
			addu $4, $1, $2	450@00003048: $ 4 <= 00010000	lui $2,2
r-M-rs	subu	rt	subu $5, $6, $7	470@0000304c: $ 5 <= ffff0000	lui $3,3
			addu $8, $6, $5	490@00003050: $ 8 <= 00050000	lui $4,4
r-W-rs	subu	rs	subu $9, $10, $11	510@00003054: $ 9 <= ffff0000	lui $5,5
			lui $12 1	530@00003058: $12 <= 00010000	lui $6,6
			addu $13, $9, $10	550@0000305c: $13 <= 00090000	lui $7,7
r-W-rt	subu	rt	subu $14, $15, $16	570@00003060: $14 <= ffff0000	lui $8,8
			lui $17 1	590@00003064: $17 <= 00010000	lui $9,9
			addu $17,$15,$14	610@00003068: $17 <= 000e0000	lui $10,10
i-M-rs	ori	rs	ori $1, $2, 1	630@0000306c: $ 1 <= 00020001	lui $11,11
			addu $4, $1, $2	650@00003070: $ 4 <= 00040001	lui $12,12
i-M-rt	ori	rt	ori $5, $6, 1	670@00003074: $ 5 <= 00060001	lui $13,13
			addu $7, $6, $5	690@00003078: $ 7 <= 000c0001	lui $14,14
i-W-rs	ori	rs	ori $3, $8, 1	710@0000307c: $ 3 <= 00050001	lui $15,15
			lui $9, 9	730@00003080: $ 9 <= 00090000	lui $16,16
			addu $10, $3, $8	750@00003084: $10 <= 000a0001	lui $17,17
i-W-rs	ori	rt	ori $11, $8, 2	770@00003088: $11 <= 00050002	
			lui $9, 10	790@0000308c: $ 9 <= 000a0000	
			addu $10, $11, $8	810@00003090: $10 <= 000a0002	
ld-W-rs	lw	rs	lw $6,0($1)	270@00003024: $ 6 <= 00030000	lui $2,1
			addu $6,$6,$2	310@00003028: $ 6 <= 00040000	lui $3,3
ld-W-rt	lw	rt	lw $8,4($1)	330@0000302c: $ 8 <= 00040000	lui $4,4
			addu $9,$7,$8	370@00003030: $ 9 <= 000b0000	lui $5,5
ld-M-rs	lw	rs	lw $10,8($1)	390@00003034: $10 <= 00050000	lui $7,7
			lui $17,17	410@00003038: $17 <= 00110000	sw $3,0($0)
			addu $11,$10,$7	430@0000303c: $11 <= 000c0000	sw $4,4($0)
ld-M-rt	lw	rt	lw $12,0($1)	450@00003040: $12 <= 00030000	sw $5,8($0)
			lui $17,18	470@00003044: $17 <= 00120000	sw $2,12($0)
			addu $13,$7,$12	490@00003048: $13 <= 000a0000	

Table 5　ori 冲突测试

测试类型	前序指令	冲突寄存器	测试序列	实际结果	初始化
i-M-rs	ori	rs	`ori $2,$1,1`	250@00003020: $ 2 <= 00010001	`lui $1,1`
			`ori $3,$2,1300`	270@00003024: $ 3 <= 00010515	`lui $2,2`
i-W-rs	ori	rs	`ori $4,$5,100`	290@00003028: $ 4 <= 00050064	`lui $3,3`
			`lui $9,9`	310@0000302c: $ 9 <= 00090000	`lui $4,4`
			`ori $5,$6,0x1ff`	330@00003030: $ 5 <= 000601ff	`lui $5,5`
r-M-rs	addu	rs	`addu $1,$6,$2`	350@00003034: $ 1 <= 00070001	`lui $6,6`
			`ori $7,$1,0xfff`	370@00003038: $ 7 <= 00070fff	`lui $7,7`
r-W-rs	subu	rs	`subu $2,$3,$1`	390@0000303c: $ 2 <= fffa0514	`lui $8,8`
			`lui $9,10`	410@00003040: $ 9 <= 000a0000	
			`ori $4,$9,0xfff`	430@00003044: $ 4 <= 000a0fff	`lui $2,1`
ld-M-rs	lw	rs	`lw $6,0($1)`	270@00003024: $ 6 <= 00030000	`lui $3,3`
			`ori $6,$6,0xf`	310@00003028: $ 6 <= 0003000f	`lui $4,4`
ld-W-rs	lw	rs	`lw $10,8($1)`	330@0000302c: $10 <= 00050000	`lui $5,5`
			`lui $17,17`	350@00003030: $17 <= 00110000	`lui $7,7`
			`ori $11,$10,0xff`	370@00003034: $11 <= 000500ff	`sw $3,0($0)`
					`sw $4,4($0)`
					`sw $5,8($0)`
					`sw $2,12($0)`

Table 6　lw 冲突测试

测试类型	前序指令	冲突寄存器	测试序列	实际结果	初始化
i-M-rs	ori	rs	`ori $1,$0,4`	310@0000302c: $ 1 <= 00000004	`lui $1,1`
			`lw $2,0($1)`	330@00003030: $ 2 <= 00020000	`lui $2,2`
r-M-rs	addu	rs	`addu $2,$0,$0`	350@00003034: $ 2 <= 00000000	`lui $3,3`
			`lw $3,4($2)`	370@00003038: $ 3 <= 00020000	`lui $4,4`
i-W-rs	ori	rs	`ori $3,$0,0`	390@0000303c: $ 3 <= 00000000	`lui $5,5`
			`lui $8,8`	410@00003040: $ 8 <= 00080000	`lui $6,6`
			`lw $7,8($3)`	430@00003044: $ 7 <= 00030000	`lui $7,7`
r-W-rs	subu	rs	`subu $5,$4,$4`	450@00003048: $ 5 <= 00000000	`sw $1,0($0)`
			`lui $9,9`	470@0000304c: $ 9 <= 00090000	`sw $2,4($0)`
			`lw $9,12($5)`	490@00003050: $ 9 <= 00040000	`sw $3,8($0)`
					`sw $4,12($0)`
lw-M-rs	lw	rs	`lw $5,12($0)`	130@00003008: $ 5 <= 00000000	`lui $5,5`
			`sw $7,0($5)`	170@0000300c: $ 6 <= 00000000	`lui $7,7`
ld-W-rs	lw	rs	`lw $6,12($0)`	190@00003010: $ 7 <= 00000000	
			`lui $7,7`	210@00003014: $ 8 <= 00080000	
			`sw $2,0($6)`	230@00003018: $ 9 <= 00000000	

Table 7 sw 冲突测试

测试类型	前序指令	冲突寄存器	测试序列	实际结果	初始化
i-M-rs	ori	rs	ori $1,$0,4	90@00003000: $ 2 <= 00010000	lui $2,1
			sw $2,0($1)	110@00003004: $ 3 <= 00030000	lui $3,3
r-M-rs	addu	rs	addu $2,$0,$1	130@00003008: $ 4 <= 00040000	lui $4,4
			sw $3,0($2)	150@0000300c: $ 5 <= 00050000	lui $5,5
i-W-rs	ori	rs	ori $3,$0,0	170@00003010: $ 7 <= 00070000	lui $7,7
			lui $8,8	190@00003014: $ 1 <= 00000004	
			sw $7,0($3)	190@00003018: *00000004 <= 00010000	
r-W-rs	subu	rs	subu $5,$4,$4	230@0000301c: $ 2 <= 00000004	
			lui $9,9	230@00003020: *00000004 <= 00030000	
			sw $9,4($5)	270@00003024: $ 3 <= 00000000	
i-M-rt	ori	rt	ori $7,$6,0xff	290@00003028: $ 8 <= 00080000	
			sw $7,4($0)	290@0000302c: *00000000 <= 00070000	
r-M-rt	addu	rt	addu $5,$4,$6	330@00003030: $ 5 <= 00000000	
			sw $5,0($0)	350@00003034: $ 9 <= 00090000	
ld-M-rs	lw	rs	lw $5,12($0)	270@00003024: $ 5 <= 00000004	ori $1,$0,1
			sw $7,0($5)	290@00003028: *00000004 <= 00070000	ori $2,$0,2
ld-W-rs	lw	rs	lw $6,12($0)	330@0000302c: $ 6 <= 00000004	ori $3,$0,3
			lui $7,7	350@00003030: $ 7 <= 00070000	ori $4,$0,4
			sw $2,0($6)	350@00003034: *00000004 <= 00000002	lui $7,7
lw-M-rt	lw	rt	lw $9,8($0)	390@00003038: $ 9 <= 00000003	sw $1,0($0)
			sw $9,12($0)	390@0000303c: *0000000c <= 00000003	sw $2,4($0)
lw-W-rt	lw	rt	lw $8,0($0)	430@00003040: $ 8 <= 00000001	sw $3,8($0)
			lui $9,9	450@00003044: $ 9 <= 00090000	sw $4,12($0)
			sw $8,4($0)	450@00003048: *00000004 <= 00000001	

Table 8 beq 冲突测试-1

测试类型	前序指令	冲突寄存器	测试序列	实际结果	测试代码	
i-E-rs	ori	rs	ori $2,$1,0	跳转	label0:	label6:
			beq $2,$1,label1		lui $1,1	ori $7,$1,0
r-E-rs	addu	rs	addu $4,$1,$0	跳转	ori $2,$1,0	nop
			beq $4,$1,label2		beq $2,$1,label1	beq $1,$7,label7
i-M-rs	ori	rs	ori $3,$1,0	跳转	nop	nop
			nop		label2:	label8:
			beq $3,$1,label3		ori $3,$1,0	lw $7,0($0)
r-M-rs	subu	rs	subu $4,$1,$3	跳转	nop	beq $7,$0,label9
			nop		beq $3,$1,label3	nop
			beq $4,$0,label4		nop	label7:
i-E-rt	ori	rt	ori $4,$1,0	跳转	label1:	addu $8,$1,$0
			beq $1,$4,label5		addu $4,$1,$0	nop
r-E-rt	addu	rt	addu $5,$1,$0	跳转	beq $4,$1,label2	beq $1,$8,label8
			beq $1,$5,label6		nop	nop
i-M-rt	ori	rt	ori $7,$1,0	跳转	label3:	label9:
			nop		subu $4,$1,$3	lw $8,0($0)
			beq $1,$7,label7		nop	beq $0,$8,label10
r-M-rt	addu	rt	addu $8,$1,$0	跳转	beq $4,$0,label4	nop
			nop		nop	label11:
			beq $1,$8,label8		label5:	lw $4,0($0)
lw-E-rs	lw	rs	lw $7,0($0)	跳转	addu $5,$1,$0	nop
			beq $7,$0,label9		beq $1,$5,label6	beq $0,$4,label12
lw-E-rt	lw	rt	lw $8,0($0)	跳转	nop	nop
			beq $0,$8,label10		label4:	label10:
lw-M-rs	lw	rs	lw $5,0($0)	跳转	ori $4,$1,0	lw $5,0($0)
			nop		beq $1,$4,label5	nop
			beq $5,$0,label11		nop	beq $5,$0,label11
lw-M-rt	lw	rt	lw $4,0($0)	跳转		nop
			nop			label12:
			beq $0,$4,label12			lui $12,12

Table 9 beq 冲突测试-2

测试类型	前序指令	冲突寄存器	测试序列	实际结果
i-W-rs	ori	rs	ori $2,$1,0	跳转正确
			nop	
			nop	
			beq $2,$1,label1	
r-W-rs	addu	rs	addu $3,$0,$1	跳转正确
			nop	
			nop	
			beq $3,$1,label2	
lw-W-rs	lw	rs	lw $4,0($0)	跳转正确
			nop	
			nop	
			beq $4,$1,label3	
i-W-rs	ori	rt	ori $5,$1,0	跳转正确
			nop	
			nop	
			beq $1,$5,label4	
r-W-rs	addu	rt	lw $7,4($0)	跳转正确
			nop	
			nop	
			beq $1,$7,end	
lw-W-rs	lw	rt	subu $6,$1,$0	跳转正确
			nop	
			nop	
			beq $1,$6,label5	

测试程序:

```
lui $1,1
ori $2,$1,0
nop
nop
beq $2,$1,label1
nop

label2:
sw $1,0($0)
lw $4,0($0)
nop
```

```
nop
beq $4,$1,label3
nop

label1:
addu $3,$0,$1
nop
nop
beq $3,$1,label2
nop

label3:
ori $5,$1,0
nop
nop
beq $1,$5,label4
nop
```

```
label5:
sw $1,4($0)
lw $7,4($0)
nop
nop
beq $1,$7,end
nop

label4:
subu $6,$1,$0
nop
nop
beq $1,$6,label5
nop

end:
lui $8,8
```

Table 10 jr 冲突测试

测试类型	前序指令	冲突寄存器	测试序列	实际结果
i-E-rs	ori	rs	ori $4,$31,0 jr $4	跳转正确
r-E-rs	addu	rs	addu $5,$31,$0 jr $5	跳转正确
i-M-rs	ori	rs	ori $6,$31,0 nop jr $6	跳转正确
r-M-rs	subu	rs	subu $12,$31,$0 nop jr $12	跳转正确
lw-E-rs	lw	rs	lw $7,0($0) jr $7	跳转正确
lw-M-rs	lw	rs	lw $8,4($0) nop jr $8	跳转正确
i-W-rs	ori	rs	ori $9,$31,0 nop nop jr $9	跳转正确
r-W-rs	addu	rs	subu $10,$31,$0 nop nop jr $10	跳转正确
lw-W-rs	lw	rs	lw $11,8($0) nop nop jr $11	跳转正确

```
lui $1,1
lui $2,2
lui $3,3
lui $4,4
jal label1
nop
addu
$2,$1,$0

jal label2
nop
addu
$3,$2,$1

jal label3
nop
ori
$4,$0,100

jal label31
nop
ori $18,1

jal label4
nop
lui $5,5

jal label5
nop
lui $6,6

jal label6
nop
lui $7,7

jal label7
nop
lui $8,8

jal label8
nop
lui $9,9
j end

label1:
ori
```

```
label2:
addu
$5,$31,$0
jr $5
nop

label3:
ori $6,$31,0
nop
jr $6
nop

label31:
subu
$12,$31,$0
nop
jr $12
nop

label4:
sw $31,0($0)
lw $7,0($0)
jr $7
nop

label5:
sw $31,4($0)
lw $8,4($0)
nop
jr $8
nop

label6:
ori $9,$31,0
nop
nop
jr $9
nop

label7:
subu
$10,$31,$0
nop
nop
jr $10
nop
```

```
label8:
sw $31,8($0)
lw $11,8($0)
nop
nop
jr $11
nop

end:
ori $31,$0,0
```

五、 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**)