Verilog 流水线 CPU 实验报告

一、 设计说明

本处理器支持 MIPS 指令集中的 addu, subu, ori, lw, sw, beq, lui, nop，j 指令，其中 addu 和 subu 不支持进位功能。支持延迟槽。使用暴力转发法。

二、流水线 CPU 顶层设计

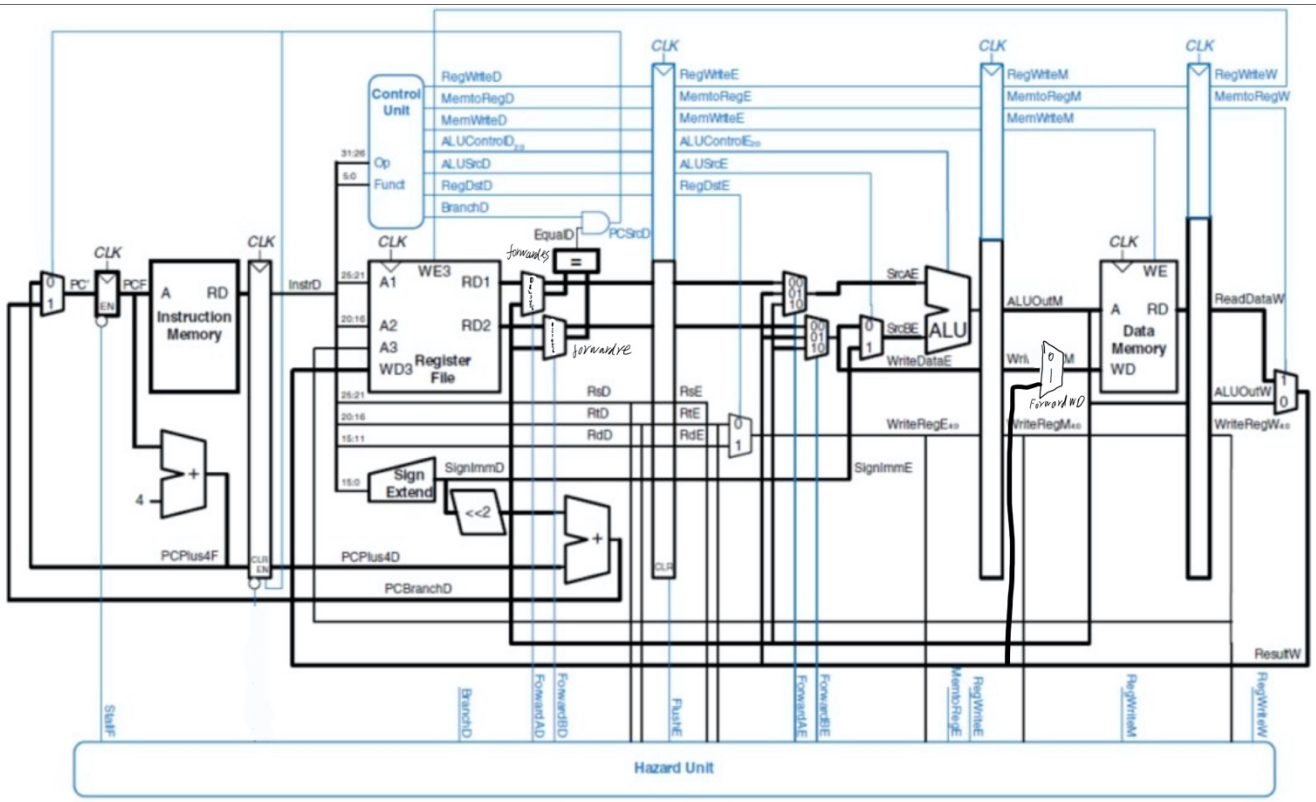
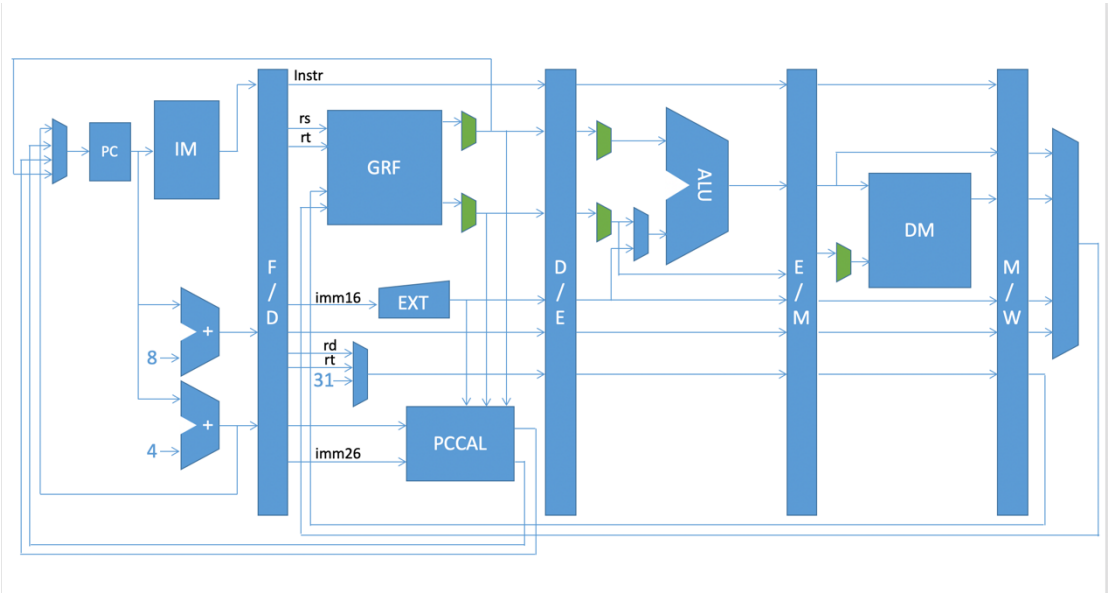


Figure 7.58 Pipelined processor with full hazard handling



三、模块规格

1. PC（程序计数器）

端口说明：

信号名	方向	描述
clk	I	时钟信号
reset	I	复位信号，将 PC 置为 0x00003000 1：复位 0：无效
En	I	使能信号 1：可写入 0：不可写入
Npc[31:0]	I	下一条指令地址
Pc[31:0]	O	当前指令地址

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	计数	当 En 有效且时钟上升沿到来时，PC 更新为 nPC 的输入值

2. Im_4kb（指令存储器）

端口说明：

信号名	方向	描述
Pc[31:0]	I	当前指令地址
IR[31:0]	O	A 地址的当前指令

功能定义：

表 4 IM 功能定义

序号	功能名称	功能描述
1	取指令	输出 A 指定的当前指令

3. GRF（通用寄存器文件）

端口说明：

表 5 GRF 端口说明

信号名	方向	描述
clk	I	时钟信号
reset	I	复位信号，将 32 个寄存器中的值全部清零 1：复位 0：无效
regWrite	I	写使能信号 1：可向 GRF 中写入数据 0：不能向 GRF 中写入数据
A1[4:0]	I	5 位地址输入信号，指定 32 个寄存器中的一个，将其中存储的数据读出到 RD1
A2[4:0]	I	5 位地址输入信号，指定 32 个寄存器中的一个，将其中存储的数据读出到 RD2
A3[4:0]	I	5 位地址输入信号，指定 32 个寄存器中的一个，作为写入的目标寄存器
WD[31:0]	I	32 位数据输入信号
RD1[31:0]	O	输出 A1 指定的寄存器中的 32 位数据
RD2[31:0]	O	输出 A2 指定的寄存器中的 32 位数据
WPC[31:0]	O	当前 PC 值

功能定义：

表 6 GRF 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 A1,A2 地址对应寄存器中所存储的数据到 RD1,RD2
3	写数据	当 WE 有效且时钟上升沿来临时，将 WD 写入 A3 所对应的寄存器中

4. ALU（算术逻辑单元）

端口说明：

信号名	方向	描述
A[31:0]	I	参与 ALU 计算的第一个值
B[31:0]	I	参与 ALU 计算的第二个值
Op[2:0]	I	ALU 功能的选择信号 0：Out = A + B 1：Out = A – B 2：Out = A | B Default：Out = 0
Out[31:0]	O	ALU 的计算结果

功能定义：

序号	功能名称	功能描述
1	加运算	Out = A + B
2	减运算	Out = A - B
3	或运算	Out = A | B

5. DM（数据存储器）

端口说明：

表 9 DM 端口说明

信号名	方向	描述
clk	I	时钟信号
reset	I	复位信号，将 DM 清零 1：复位 0：无效
WE	I	写使能信号 1：可向 DM 中写入数据 0：不能向 DM 中写入数据
A[31:0]	I	32 位地址输入信号，指定读出或写入数据的地址
WD[31:0]	I	32 位数据输入信号
RD[31:0]	O	输出 A 指定的 32 位数据

功能定义：

表 10 DM 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，将 DM 中数据清零
2	读数据	读出 A 所指定的数据到 RD
3	写数据	当 WE 有效且时钟上升沿到来时，将输入数据 WD 写入 A 所指定的地址

6. EXT（位扩展单元）

端口说明：

表 11 EXT 端口说明

信号名	方向	描述
In[15:0]	I	16 位输入数据
Op[1:0]	I	位扩展方式选择信号 0：零扩展 1：符号扩展 2：加载至高位 3：beq 扩展
Out[31:0]	O	扩展后的 32 位输出数据

功能定义：

表 12 EXT 功能定义

序号	功能名称	功能描述
1	零扩展	将 16 位输入数据进行零扩展，输出 32 位数据
2	符号扩展	将 16 位输入数据进行符号扩展，输出 32 位数据
3	加载至高位	将 16 位输入数据加载至高 16 位，并将低 16 位置 0
4	B 指令扩展	左移两位后符号扩展

7. Npc（指令地址计算单元）

端口说明：

表 13 PCCAL 端口说明

信号名	方向	描述
Pc4[31:0]	I	分支指令的基地址（PC+4）
nPcsel[1:0]	I	分支指令类型
Imm32[31:0]	I	分支指令的偏移量
Imm26[25:0]	I	跳转指令的目的地址的中间 26 位
Cmp1[31:0]	I	分支指令的第一个比较数
Cmp2[31:0]	I	分支指令的第二个比较数
Jump	I	是否为 J 指令
Jr	I	是否为 Jr 指令
Gpr[31:0]	I	Jr 指令跳转地址

npc[31:0]	O	计算出的下一条指令的地址

功能定义：

表 14 PCCAL 功能定义

序号	功能名称	功能描述
1	计算分支地址	通过比较 Cmp1 和 Cmp2 是否相等计算分支地址
2	计算跳转地址	通过位拼接操作计算跳转地址

8. HZD（冲突处理单元）

端口说明：

表 15 HZD 端口说明

信号名	方向	描述
Instr_D[31:0]	I	位于 D 级的指令
Instr_E[31:0]	I	位于 E 级的指令
Instr_M[31:0]	I	位于 M 级的指令
Instr_W[31:0]	I	位于 W 级的指令
WriteAddr_E[4:0]	I	在 E 级的寄存器写入地址
WriteAddr_M[4:0]	I	在 M 级的寄存器写入地址
WriteAddr_W[4:0]	I	在 W 级的寄存器写入地址
ForwardRsD [2:0]	O	位于 D 级的寄存器 RS 需求转发多选器的选择信号
ForwardRtD [2:0]	O	位于 D 级的寄存器 RT 需求转发多选器的选择信号
ForwardRsE [2:0]	O	位于 E 级的寄存器 RS 需求转发多选器的选择信号
ForwardRtE [2:0]	O	位于 E 级的寄存器 RT 需求转发多选器的选择信号
ForwardWD	O	位于 M 级的寄存器 RT 需求转发多选器的选择信号
PCEnD	O	Pc 使能
EnD	O	D 级流水寄存器使能
FlushE	O	E 级

功能定义：

表 16 HZD 功能定义

序号	功能名称	功能描述
1	转发	根据各流水级的寄存器供需关系检测冲突并通过转发处理
2	暂停	根据各流水级的寄存器供需关系检测冲突并通过暂停处理

四、控制器设计

控制器真值表：

func	000000	000000	001101	100011	101011	000100	001111	000010
opcode	100001	100011						
op	addu	subu	ori	lw	sw	beq	lui	J
nPC_Sel	0	0	0	0	0	1	0	X
RegDst(写寄存器目标)1-rd	1	1	0	0	x	x	0	X
RegWrite（写寄存器使能端）	1	1	1	1	x	x	1	X
MemWrite（数据写入指定存储器单元）	0	0	0	0	1	0	0	0
MemtoReg（写入寄存器的数据来源）	0	0	0	1	x	x	x	X
ALUSrc（第二个 alu 操作数来源）	0	0	1	1	1	0	1	X
Jump	0	0	0	0	0	0	0	1
ALUOp[2:0]（控制 ALU 运算）	000	001	010	000	000	011	000	XXX
ExtOp[2:0] （扩展方式）	x	x	000	001	001	x	010	XXX

五、冲突单元设计

（1）暂停设计

暂停的判断方式：

Tuse：指令（需求者）在 D 级时，过多久之后，某元件就必须要用相应寄存器的值。

Tnew：位于某个流水级的某个指令（供给者），它经过多少个时钟周期可以算出结果并且存储到流水级寄存器里。

Tnew > Tuse 时，且在前指令存入的 rt/rd 不是$0，且存在一个 irD 的源寄存器（需求）与前指令的目标寄存器（供给）相等时，需要暂停。

（2）转发设计

当 Demand=1，并且需求者和提供者相同时时，由于算出的结果还没有存入 GRF 中，如果需要用到这个结果，可以通过转发的方式。

表格见 excel。

六、测试程序

.见附件

七、问答题

1. 在本实验中你遇到了哪些不同指令类型组合产生的冲突？你又是如何解决的？相应的测试样例是什么样的？

用例编号	测试类型	前序指令	测试序列
1	R-M-RS	subu	subu $1, $2, $3 addu $4, $1, $2
2	R-M-RT	subu	subu $1, $2, $3 addu $4, $2, $1

3	R-W-RS	subu	subu $1, $2, $3
			nop
			addu $4, $1, $2

4	R-M-RT	subu	subu $1, $2, $3
			nop
			addu $4, $2, $1

5	I-M-RS	ori	ori $1, $2, 1000 addu $4, $1, $2
6	I-M-RT	ori	ori $1, $2, 1000

			addu $4, $2, $1
7	I-W-RS	ori	ori $1, $2, 1000 nop addu $4, $1, $2
8	I-W-RT	ori	ori $1, $2, 1000 nop addu $4, $2, $1
9	L-M-RS	addu	addu $1, $2, $3 lw $4, 0($1)

10	L-W-RS	addu	addu $1, $2, $3
			nop
			lw $4, 0($1)

11	S-M-RS	addu	addu $1, $2, $3 sw $4, 0($1)
12	S-M-RT	addu	addu $1, $2, $3 sw $1, 0($4)

13	S-W-RS	addu	addu $1, $2, $3
			nop
			sw $4, 0($1)

14	S-W-RT	addu	addu $1, $2, $3
			nop
			sw $1, 0($4)

15	B-E-RS	subu	subu $1, $2, $3 beq $1, $4, label
16	B-E-RT	subu	subu $1, $2, $3 beq $4, $1, label
17	B-M-RS	subu	subu $1, $2, $3 nop beq $1, $4, label
18	B-M-RT	subu	subu $1, $2, $3 nop beq $4, $1, label
19	B-W-RS	subu	subu $1, $2, $3 nop nop beq $1, $4, label
20	B-W-RT	subu	subu $1, $2, $3 nop nop beq $4, $1, label
21	R-E-jal	jal	jal jemp addu $ra, $ra, 4
22	R-M-jal	jal	jal jemp

			nop addu $ra, $ra, 4
21	JR-E-RS	jr	subu $1, $2, $3 jr $1
23	JR-M-RS	jr	subu $1, $2, $3 nop jr $1
25	JR-W-RS	jr	subu $1, $2, $3 nop nop jr $1