MIPS 流水线处理器设计文档

MIPS 流水线处理器设计文档由顶层模块、数据通路、控制器、冲突模块、测试程序组成。

一、顶层模块 mips.v

序号	信号	位数	方向	功能描述
1	clk	1	I	pc、grf、dm 的时钟信号
2	reset	1	I	当 reset=1 且时钟为上跳沿时，pc 被设置为 0x00003000，gpr 所有寄存器置零，dm 所有数据置零

顶层模块将数据通路模块 datapath.v 与控制器模块 control.v,hazard.v 相连。

二、数据通路 datapath.v

信号端口

序号	信号	位数	方向	描述
1	reset	1	I	复位信号
2	clk	1	I	时钟信号
3	PCOP	2	I	muxPCOP 模块控制信号
4	enable	1	I	PC,IF_ID 使能端
5	clear	1	I	ID_EX 清零信号
6	RegDst	2	I	muxRegDst 模块控制信号
7	ExtOP	1	I	Ext 模块控制信号
8	RegWrite	1	I	寄存器文件写使能
9	RegWData	1	I	muxRegWData 模块控制信号
10	ALUSrc	1	I	muxALUSrc 模块控制信号
11	ALUOP	4	I	选择 ALU 运算类型的信号
12	MemWrite	1	I	dm 写入控制信号
13	Instr_original	32	O	IM 输出指令
14	Instr_D	32	O	D 级指令
15	Instr_E	32	O	E 级指令
16	Instr_W	32	O	W 级指令
17	MF_RS_D_OP	2	O	MF_RS_D 模块控制信号

18	MF_RT_D_OP	2	O	MF_RT_D 模块控制信号
19	MF_RS_E_OP	2	O	MF_RS_E 模块控制信号
20	MF_RT_E_OP	2	O	MF_RT_E 模块控制信号
21	MF_RT_M_OP	2	O	MF_RT_M 模块控制信号
22	Equal_out	1	O	equal 模块输出信号
23	bgez_out	1	O	bgez_out 信号
24	bgtz_out	1	O	bgtz_out 信号
25	bltz_out	1	O	bltz_out 信号
26	blez_out	1	O	blez_out 信号
27	busy	1	O	Busy 信号

datapath.v 将 pc.v, dm.v , im.v , add4.v,npc.v,alu.v , grf.v , ext.v , muxPCOP.v , muxRegWData.v , muxRegDst.v , muxALUSrc.v 等模块连接在一起。下面将介绍这些模块。

1、grf.v

信号端口

序号	信号	位数	方向	描述
1	RA1	5	I	读寄存器文件时第一个寄存器的下标
2	RA2	5	I	读寄存器文件时第二个寄存器的下标
3	Waddr	5	I	写入文件的寄存器下标
4	WData	32	I	寄存器文件写入数据
5	clk	1	I	时钟信号
6	reset	1	I	复位信号
7	RegWrite	1	I	寄存器文件写使能
8	RData1	32	O	读寄存器文件时第一个寄存器的输出
9	RData2	32	O	读寄存器文件时第二个寄存器的输出

模块功能

序号	功能名称	功能描述
1	读寄存器	RData1 输出 RA1 所寻址的寄存器的文件数据，RData2 输出 RA2 所寻址的寄存器的文件数据
2	写寄存器	当时钟上升沿到来时，并且 reset=0 且 RegWrite 有效时，WD 被写入 WA 所寻址的寄存器
3	复位	reset=1 且 clk 为上升沿时 gpr 所有寄存器置零

2、alu.v

信号端口

序号	信号	位数	方向	描述
1	ALU_A	32	I	ALU 的第一个操作数
2	ALU_B	32	I	ALU 的第二个操作数
3	ALUOP	4	I	选择 ALU 运算类型的信号
4	ALU_C	32	O	ALU 的运算结果
5	equal	1	O	判断 ALU_A 与 ALU_B 是否相等，若相等输出 1，否则输出 0
6	s	5	I	逻辑左移、右移位数

模块功能

序号	功能名称	功能描述
1	加法	输出端 ALU_C=ALU_A+ALU_B
2	减法	输出端 ALU_C=ALU_A-ALU_B
3	或	输出端 ALU_C=ALU_A|ALU_B
4	判断相等	若 ALU_A=ALU_B，equal 信号输出 1，否则输出 0
5	置高位	输出端 ALU_C={{ALU_B[15:0]},{16{1'b0}}}
6	逻辑左移	输出端 ALU_C=ALU_A<<s
7	逻辑右移	输出端 ALU_C=ALU_A>>s
8	与	输出端 ALU_C=ALU_A&ALU_B
9	异或	输出端 ALU_C=ALU_A^ALU_B
10	nor	输出端 ALU_C=ALU_A nor ALU_B
11	算术右移	输出端 ALU_C={{32{ALU_B[31]}},ALU_B}>>s
12	算术可变右移	输出端 ALU_C={{32{ALU_B[31]}},ALU_B}>>ALU_A[4:0]
13	逻辑可变左移	输出端 ALU_C=ALU_B<<ALU_A[4:0]
14	逻辑可变右移	输出端 ALU_C=ALU_B>>ALU_A[4:0]
15	小于置一（有符号）	输出端 ALU_C=$signed(ALU_A)<$signed(ALU_B)?32'b1:32'b0
16	小于置一（无符号）	输出端 ALU_C=(ALU_A<ALU_B?32'b1:32'b0):32'b0

3、dm.v

信号端口

序号	信号	位数	方向	描述
1	addr	32	I	写入数据储存器的数据的地址/读取数据储存器的数据的地址
2	din	32	I	写入数据储存器的数据
3	MemWrite	1	I	写入控制信号
4	clk	1	I	时钟信号
5	reset	1	I	复位信号
6	dout	32	O	从数据存储器读取的数据
7	DMOP	2	I	Sb，sh，sw 信号
8	Be_out	4	I	be 模块输出

模块功能

序号	功能名称	功能描述
1	读数据储存器	dout 输出数据储存器在 addr[11:2]地址的数据
2	写数据储存器	当 MemWrite 有效,reset=0,并且时钟为上升沿时，din 被写入数据存储器地址为 addr[11:2]的区域
3	复位	reset=1 并且时钟为上升沿时,DM 所有数据置零

4、ext.v

信号端口

序号	信号	位数	方向	描述
1	ext_in	16	I	位扩展输入信号
2	EXTOP	1	I	信号为 1 时，进行符号扩展，信号为 0 时，进行 0 扩展
3	ext_out	32	O	位扩展输出信号

模块功能

序号	功能名称	功能描述
1	符号扩展	当 EXTOP 信号为 1 时，ext_in 符号扩展后由 ext_out 输出
2	0 扩展	当 EXTOP 信号为 0 时，在 ext_in 前置 16 个 0 后由 ext_out 输出

5、im.v

信号端口

序号	信号	位数	方向	描述
1	address	32	I	指令地址
2	instr	32	O	指令

模块功能

序号	功能名称	功能描述
1	取指令	取出位于(address-0x00003000)[12:2]处指令

6、pc.v

信号端口

序号	信号	位数	方向	描述
1	PC_in	32	I	下一条指令 PC 值
2	clk	1	I	时钟信号
3	reset	1	I	复位信号
4	PC_out	32	O	当前 PC 值
5	Enable	1	I	使能端

模块功能

序号	功能名称	功能描述
1	输出指令地址	时钟上升沿且 reset=0，enable=1，PC 输出指令地址
2	复位	时钟上升沿且 reset=1，将 PC 置为 0x00003000

7、muxPCOP.v

信号端口

序号	信号	位数	方向	描述
1	PCOP	2	I	muxPCOP 控制信号
2	PC4	32	I	PC+4
3	NPC	32	I	NPC 结果
4	newPC	32	O	下条指令地址
5	RD1	32	I	rs 值

模块功能

序号	功能名称	功能描述
1	输出下条指令地址	PCOP==2'b00:newPC<=PC4; PCOP==2'b01:newPC<=NPC; PCOP==2'b10:newPC<=RD1;

8、muxRegDst.v

信号端口

序号	信号	位数	方向	描述
1	RegDst	2	I	muxRegDst 控制信号
2	IR_W	32	I	W 级指令

3	WAddr	5	O	写入文件的寄存器下标

模块功能

序号	功能名称	功能描述
1	输出写入文件的寄存器下标	RegDst=2’b00，WAddr=rd RegDst=2’b01，WAddr=rt RegDst=2’b10，WAddr=5’d31

9、muxRegWData.v

信号端口

序号	信号	位数	方向	描述
1	ALUC_W	32	I	W 级 ALU 结果
2	DM_W	32	I	W 级从数据存储器读取的数据
3	RegWData	2	I	muxRegWData 的控制信号
4	WData	32	O	寄存器文件写入数据
5	IR_W	32	I	W 级指令
6	PC4_W	32	I	PC+4

模块功能

序号	功能名称	功能描述
1	输出寄存器文件写入数据	RegWData=2’b00:WData<=ALUC_W; RegWData=2’b01:WData<=DM_W; RegWData=2’b10:WData<=PC4_W+4;

10、muxALUSrc.v

信号端口

序号	信号	位数	方向	描述
1	ALUSrc	1	I	muxAlUSrc 的控制信号
2	RD2	32	I	读寄存器文件时第二个寄存器的下标
3	EXT_out	32	I	位扩展输出信号
4	ALU_B	32	O	ALU 的第二个操作数

模块功能

序号	功能名称	功能描述
1	选择 ALU 的第二个操作数	ALUSrc=0 时：ALU_B=RD2 ALUSrc=1 时：ALU_B=EXT_out

11、add4.v

信号端口

序号	信号	位数	方向	描述
1	PC	32	I	PC
3	PC4	32	O	PC+4

模块功能

序号	功能名称	功能描述
1	PC+4	PC4=PC+4

12、IF_ID.v

信号端口

序号	信号	位数	方向	描述
1	clk	1	I	时钟信号
2	reset	1	I	复位信号
3	IR_IM	32	I	IM 输出指令
4	PC4_ADD4	32	I	PC+4
5	PC4_D	32	O	PC+4
6	IR_D	32	O	D 级指令
7	enable	1	I	使能端

模块功能

序号	功能名称	功能描述
1	D 级流水线寄存器	传递 IR 与数据
2	使能端置零	暂停时 enable=0

13、ID_EX.v

信号端口

序号	信号	位数	方向	描述
1	clk	1	I	时钟信号
2	reset	1	I	复位信号
3	IR_D	32	I	D 级指令
4	RD1	32	I	读寄存器文件时第一个寄存器的输出
5	RD2	32	O	读寄存器文件时二个寄存器的输出
6	EXT_out	32	O	D 级指令
7	PC4_D	32	O	PC+4

8	IR_E	32	O	E 级指令
9	RD1_E	32	O	读寄存器文件时第一个寄存器的输出
10	RD2_E	32	O	读寄存器文件时第二个寄存器的输出
11	EXT_E	32	O	EXT 结果
12	PC4_E	32	O	PC+4
13	Clear	1	I	清零信号

模块功能

序号	功能名称	功能描述
1	E 级流水线寄存器	传递 IR 与数据
2	清零	ID_EX 清零

14、EX_MEM.v

信号端口

序号	信号	位数	方向	描述
1	clk	1	I	时钟信号
2	reset	1	I	复位信号
3	IR_E	32	I	E 级指令
4	RD2_E	32	I	读寄存器文件时二个寄存器的输出
5	ALU_out	32	I	ALU 结果
6	PC4_E	32	I	PC+4
7	IR_M	32	O	M 级指令
8	ALUC_M	32	O	ALU 结果
9	PC4_M	32	O	PC+4
10	RD2_M	32	O	读寄存器文件时二个寄存器的输出
11	muxHILO_out	32	I	muxHILO 输出值
12	muxHILO_M	32	I	muxHILO 输出值

模块功能

序号	功能名称	功能描述
1	M 级流水线寄存器	传递 IR 与数据

15、MEM_WB.v

信号端口

序号	信号	位数	方向	描述
1	clk	1	I	时钟信号

2	reset	1	I	复位信号
3	IR_M	32	I	M 级指令
4	DM_Data	32	I	从数据存储器读取的数据
5	ALU_M	32	I	ALU 结果
6	PC4_M	32	I	PC+4
7	IR_W	32	O	W 级指令
8	ALUC_W	32	O	ALU 结果
9	PC4_W	32	O	PC+4
10	DM_W	32	O	从数据存储器读取的数据

模块功能

序号	功能名称	功能描述
1	W 级流水线寄存器	传递 IR 与数据

16、MF_RS_E.v

信号端口

序号	信号	位数	方向	描述
1	MF_RS_E_OP	2	I	MF_RS_E 控制信号
2	RD1_E	32	I	读寄存器文件时第二个寄存器的值
3	ALUC_M	32	I	ALU 结果
4	WData	32	I	muxRegWData 结果
5	MF_RS_E_out	32	O	转发结果
6	PC4_M	32	I	PC+4

模块功能

序号	功能名称	功能描述
1	转发	控制信号为 01,10,11 时转发

17、MF_RT_E.v

信号端口

序号	信号	位数	方向	描述
1	MF_RT_E_OP	2	I	MF_RT_E 控制信号
2	RD2_E	32	I	读寄存器文件时第二个寄存器的值
3	ALUC_M	32	I	ALU 结果

4	WData	32	I	muxRegWData 结果
5	RD2	32	O	转发结果
6	PC4+M	32	I	PC+4

模块功能

序号	功能名称	功能描述
1	转发	控制信号为 01,10,11 时转发

18、MF_RS_D.v

信号端口

序号	信号	位数	方向	描述
1	MF_RS_D_OP	2	I	MF_RS_D 控制信号
2	RData1	32	I	读寄存器文件时第 1 个寄存器的值
3	ALUC_M	32	I	ALU 结果
4	WData	32	O	muxRegWData 结果
5	MF_RS_D_out	32	I	转发结果
6	PC4_M	32	I	PC+4

模块功能

序号	功能名称	功能描述
1	转发	控制信号为 01,10,11 时转发

19、MF_RT_D.v

信号端口

序号	信号	位数	方向	描述
1	MF_RT_D_OP	2	I	MF_RS_D 控制信号
2	RData2	32	I	读寄存器文件时第二个寄存器的值
3	ALUC_M	32	I	ALU 结果
4	WData	32	O	muxRegWData 结果
5	MF_RT_D_out	32	I	转发结果
6	PC4_M	32	I	PC+4

模块功能

序号	功能名称	功能描述
1	转发	控制信号为 01,10,11 时转发

20、MF_RT_M.v

信号端口

序号	信号	位数	方向	描述
1	MF_RT_M_OP	2	I	MF_RT_M 控制信号
2	RD2_M	32	I	读寄存器文件时第二个寄存器的值
3	WData	32	O	muxRegWData 结果
4	MF_RT_M_out	32	I	转发结果

模块功能

序号	功能名称	功能描述
1	转发	控制信号为 01 时转发

21、compare.v

信号端口

序号	信号	位数	方向	描述
1	RD1	32	I	读寄存器文件时第一个寄存器的值
2	RD2	32	I	读寄存器文件时第二个寄存器的值
3	Equal_out	1	O	RD1=RD2，则为 1，否则为 0
4	bgez_out	1	O	$signed(RD1)>=0?1'b1:1'b0;
5	bgtz_out	1	O	$signed(RD1)>0?1'b1:1'b0;
6	bltz_out	1	O	$signed(RD1)<0?1'b1:1'b0;
7	blez_out	1	O	$signed(RD1)<=0?1'b1:1'b0;

22、Be.v

信号端口

序号	信号	位数	方向	描述
1	Addr1_0	2	I	地址低两位
2	DMOP	2	I	DMOP 控制信号
3	be_out	1	O	字节使能

23、Expand.v

信号端口

序号	信号	位数	方向	描述
1	expandOP	3	I	expand 控制信号
2	Addr1_0	2	I	地址低两位
3	DM_W	32	I	原始数据
4	Expand_out	32	O	数据扩展结果

24、Multdiv.v

信号端口

序号	信号	位数	方向	描述
1	clk	1	I	时钟信号
2	reset	1	I	复位信号
3	start	2	I	乘除法开始信号
4	multdivOP	2	I	Multdiv 控制信号
5	multdiv_A	32	I	Rs 值
6	multdiv_B	32	I	Rt 值
7	HIWrite	1	I	HI 写使能
8	LOWrite	1	I	LO 写使能
9	HI	32	O	HI 寄存器输出
10	LO	32	O	LO 寄存器输出
11	Busy	1	O	Busy 信号

25、muxHILO.v

信号端口

序号	信号	位数	方向	描述
1	HILOOP	1	I	muxHILO 控制信号
2	HI	32	I	HI 寄存器输出
3	LO	32	I	LO 寄存器输出
4	muxHILO_out	32	O	muxHILO 结果

26、muxSelect.v

信号端口

序号	信号	位数	方向	描述
1	Select	1	I	MuxSelect 控制信号
2	ALUC_M	32	I	ALU 结果
3	muxHILO_M	32	I	muxHILO 结果
4	muxSelect_out	32	O	MuxSelect 结果

三、控制器 control.v

control.v 信号端口

序号	信号	位数	方向	描述
1	instr	32	I	指令
3	PCOP	2	O	muxPCOP 控制信号
4	RegDst	2	O	muxRegDst 控制信号
5	ExtOP	1	O	ext 控制信号
6	RegWrite	1	O	gpr 写入控制信号
7	RegWData	2	O	muxRegWData 的控制信号
8	ALUSrc	1	O	muxAlUSrc 的控制信号
9	ALUOP	4	O	alu 控制信号
10	MemWrite	1	O	dm 写入控制信号
11	Equal	1	I	Equal 信号
12	bgez_out	1	I	bgez_out 信号
13	bgtz_out	1	I	bgtz_out 信号
14	bltz_out	1	I	bltz_out 信号
15	blez_out	1	I	blez_out 信号
16	start	2	O	乘除法开始信号
17	multdivOP	2	O	乘除控制信号
18	HIWrite	1	O	HI 写使能
19	LOWrite	1	O	LO 写使能
20	Select	1	O	MuxSelect 控制信号
21	DMOP	2	O	Sb，sh，sw 信号
22	expandOP	2	O	expand 控制信号
23	HILOOP	1	O	MuxHILO 控制信号

四、冲突模块

信号端口

序号	信号	位数	方向	描述
1	IR_D	32	I	D 级指令

2	IR_E	32	I	E 级指令
3	IR_M	32	I	M 级指令
4	IR_W	32	I	W 级指令
5	MF_RT_E_OP	2	O	MF_RT_E 控制信号
6	MF_RS_E_OP	2	O	MF_RS_E 控制信号
7	MF_RS_D_OP	2	O	MF_RS_D 控制信号
8	MF_RT_D_OP	2	O	MF_RT_D 控制信号
9	MF_RT_M_OP	2	O	MF_RT_M 控制信号
10	enable	1	O	PC,IF_ID 使能信号
11	clear	1	O	ID_EX 清零信号

五、测试代码

```
lui $1,0x1234
ori $1,$1,0x6789
sw $1,0($0)
sh $1,6($0)
sb $1,5($0)
lb $2,6($0)
lbu $3,6($0)
add $4,$3,$2
sb $4,8($0)
lh $5,8($0)
lhu $6,4($0)
sub $1,$0,$1
sh $1,10($0)
lui $7,0x1010
lhu $8,10($0)
sra $9,$8,2
sllv $10,$9,$5
nor $11,$10,$9
slt $12,$10,$11
sltu $12,$10,$11
srlv $13,$11,$12
srav $14,$13,$12
andi $15,$14,0x9234
xori $16,$15,0x5678
slti $17,$16,0x7000
sltiu $18,$16,0x4562
lh $13,2($0)
lw $14,4($0)
divu $14,$13
jal label
mult $13,$14
mfhi $15
mflo $16
mthi $2
mtlo $2
srlv $16,$16,$4
div $13,$16
j end
mfhi $17
mflo $18
label:
jalr $1,$31
nop
end:
nop
```

```
$ 1 <= 12340000
$ 1 <= 12346789
*00000000 <= 12346789
*00000006 <= 6789
*00000005 <= 89
$ 2 <= ffffff89
$ 3 <= 00000089
$ 4 <= 00000012
*00000008 <= 12
$ 5 <= 00000012
$ 6 <= 00008900
$ 1 <= edcb9877
*0000000a <= 9877
$ 7 <= 10100000
$ 8 <= 00009877
$ 9 <= 0000261d
$10 <= 98740000
$11 <= 678bd9e2
$12 <= 00000001
$12 <= 00000000
$13 <= 678bd9e2
$14 <= 678bd9e2
$15 <= 00009020
$16 <= 0000c658
$17 <= 00000000
$18 <= 00000000
$13 <= 00001234
$14 <= 67898900
$31 <= 0000307c
$ 1 <= 000030a8
$15 <= 0000075c
$16 <= b391d400
$16 <= 00002ce4
$17 <= 00001234
```

六、思考题

1、为什么需要有单独的乘除法部件而不是整合进 ALU？为何需要有独立的 HI，LO 寄存器？

整合进 ALU 会导致正在进行乘除法时，后续指令不能使用 ALU，暂停的时钟周期变多，同时导致 E 级流水耗时过长，流水线时钟周期变大。

2、参照你对延迟槽的理解，试解释“乘除槽”。

乘除槽指乘法指令后五个周期，除法指令后十个周期。在乘除槽中的与乘除、HI、LO 相关的指令都应该被阻塞，其他指令可以运行。

3、为何上文文末提到的 lb 等指令使用的数据扩展模块应在 MEM/WB 之后,而不能在 DM 之后?

DM 本身耗时大，如果在 DM 后，会导致 M 级流水耗时更大，导致流水线时钟周期因此变大，降低性能。

4、举例说明并分析何时按字节访问内存相对于按字访问内存性能上更有优势。（Hint： 考虑 C 语言中字符串的情况）

访问 char 类型变量时按字节访问内存相对于按字访问内存性能上更有优势，因为 char 类型占一个字节。

5、如何概括你所设计的 CPU 的设计风格？为了对抗复杂性你采取了哪些抽象和规范手段？

规划者。将指令归类为 CAL_R,CAL_I,B,md,mf,mt 等类型，对同类型指令进行统一处理。

6、你对流水线 CPU 设计风格有何见解？

规划者和侦测者各有优劣，规划者一目了然但是代码量太大。侦测者代码简洁，偏向于基础原理。