MIPS 单周期处理器设计文档

MIPS 单周期处理器设计文档由顶层结构、各模块规格、控制器设计、测试程序组成。

一、顶层结构

从顶层视图来看，MIPS 单周期处理器模块主要分为 IFU（取指令单元）、GRF（通用寄存器组）、ALU（算术逻辑单元）、DM（数据存储器）、EXT（位扩展器）、Controller（控制器）以及 shift left 16（左移十六位）。除此之外，还有数个多路开关、逻辑门、Splitter、时钟信号、reset 信号。

IFU、GRF、DM、共用一个时钟信号。

reset 信号为顶层唯一有效驱动源。

顶层信号端口

序号	信号	位数	功能描述
1	reset	1	当 reset=1 时，PC 被设置为 0x00000000，GPR 所有寄存器置零，DM 所有数据置零
2	clk	1	IFU、GRF、DM 的时钟信号

二、各模块规格（除控制器）

1、IFU

IFU 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	if beq zero	1	I	信号为 1 时执行 beq 指令，修改 PC 值，否则 PC=PC+4
4	Instr	32	O	输出 32 位指令

IFU 模块功能

序号	功能名称	功能描述
1	复位	reset=1 时，PC 被置为 0x00000000
2	取指令	以 PC 为地址从 IM 中取出 32 位指令
3	修改 PC 值	在时钟上升沿时，当 if beq zero=1 时，按 beq 指令修改 PC，否则 PC=PC+4
4	输出	Instr 输出取出的 32 位指令

2、GRF

GRF 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	WA	5	I	写入文件的寄存器下标
4	WD	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	RD1	32	O	读寄存器文件时第一个寄存器的输出
9	RD2	32	O	读寄存器文件时第二个寄存器的输出

GRF 模块功能

序号	功能名称	功能描述
1	读寄存器	RD1 输出 RA1 所寻址的寄存器的文件数据，RD2 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 RegWrite 有效时，WD 被写入 WA 所寻址的寄存器
3	复位	GRF 所有寄存器置零

3、ALU

ALU 信号端口

序号	信号	位数	方向	描述
1	A	32	I	ALU 的第一个操作数
2	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	C	32	O	ALU 的运算结果
5	If =	1	O	判断 A 与 B 是否相等，若相等输出 1，否则输出 0

ALU 模块功能

序号	功能名称	功能描述
1	加法	输出端 C=A+B
2	减法	输出端 C=A-B
3	或	输出端 C=A|B
4	判断相等	若 A=B，if =信号输出 1，否则输出 0

4、DM

DM 信号端口

序号	信号	位数	方向	描述
1	MemAddr	5	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	MemData	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	reseet	1	I	复位信号
6	Data	32	O	从数据存储器读取的数据

DM 模块功能

序号	功能名称	功能描述
1	读数据储存器	Data 输出数据寄存器在 MemAddr 地址的数据
2	写数据储存器	当 MemWrite 有效并且时钟为上升沿时，MemData 被写入数据存储器地址为 MemAddr 的区域
3	复位	DM 所有数据置零

5、EXT

EXT 信号端口

序号	信号	位数	方向	描述
1	A	16	I	位扩展输入信号
2	0 or sign	1	I	信号为 1 时，进行符号扩展，信号为 0 时，进行 0 扩展
3	B	32	O	位扩展输出信号

EXT 模块功能

序号	功能名称	功能描述
1	符号扩展	当 0 or sign 信号为 1 时，A 符号扩展后由 B 输出
2	0 扩展	当 0 or sign 信号为 0 时，在 A 前置 16 个 0 后由 B 输出

6、shift left 16

Shift left 16 信号端口

序号	信号	位数	方向	描述

1	A	16	I	左移输入
2	B	32	O	左移后输出信号

Shift left 16 模块功能

序号	功能名称	功能名称
1	左移 16 位	将 A 左移 16 位后由 B 输出

7、shift left 2

Shift left 2 信号端口

序号	信号	位数	方向	描述
1	A	32	I	左移输入
2	B	32	O	左移后输出信号

Shift left 2 模块功能

序号	功能名称	功能名称
1	左移 2 位	将 A 左移 2 位后由 B 输出

三、控制器设计

以下为控制信号产生的真值表：

func	100001	100011	无					
opcode	000000	000000	001101	100011	101011	000100	001111	
控制信号	addu	subu	ori	lw	sw	beq	lui	nop
if beq	0	0	0	0	0	1	0	x
Ext	x	x	0	1	1	x	0	x
RegDst	0	0	1	1	x	x	1	x
RegWrite	1	1	1	1	0	0	1	0
RegWData [1:0]	00	00	00	01	x	x	10	x
ALUSrc	0	0	1	1	1	x	x	x
ALUOP[3:0]	0000	0001	0010	0000	0000	x	x	x
MemWrite	0	0	0	0	1	0	0	0

以下为各控制信号功能：

if beq 与 ALU 的 if=信号同为 1 时，if beq zero=1，PC 执行 beq 跳转。

Ext 信号为 0 时，执行 0 扩展；为 1 时，执行符号扩展。

RegDst 为 0 时，GRF 的 WA 端口输入为 rd，为 1 时，GRF 的 WA 端口输入

为 rt.

RegWrite 为 0 时，GRF 写使能端无效，为 1 时，GRF 写使能端有效。

RegWData[1:0]为 00 时，GRF 的 WD 端口输入为 ALU 的 C 端口，为 01 时，GRF 的 WD 端口输入为 DM 的 Data 端口，为 10 时 GRF 的 WD 端口输入为 shift left 16 输出端 B。

ALUSrc 为 0 时，ALU 的 B 端口输入为 GRF 的 RD2 端口，为 1 时，ALU 的 B 端口输入为 EXT 的 B 端口。

ALUOP[3:0]为 0000 时，ALU 执行加法，为 0001 时，ALU 执行减法，为 0010 时，ALU 执行或操作。

MemWrite 为 0 时，DM 写入控制信号无效，为 1 时，DM 写入控制信号有效。

四、测试程序

汇编语言：

```
nop
lui $2,0x0101
sw $2,4($0)
nop
lw $3,4($0)
addu $2,$2,$3
subu $4,$2,$4
beq $4,$2,label
lw $1,4($0)
label:
ori $5,$2,1
nop
sw $5,8($0)
```

机器码：

00000000 3c020101 ac020004 00000000 8c030004 00431021 00442023 108200001 8c010004 34450001 00000000 ac050008

测试期望：指令结束后，1 号寄存器值为 0x00000000,2、4 号寄存器值为 0x02020000,3 号寄存器值为 0x01010000,5 号寄存器值为 0x02020001,DM 的地址为 4 处所存值为 0x01010000,DM 的地址为 8 处所存值为 0x02020001。

思考题：

1、在上个学年的计组课程中，PC（程序计数器）位数被规定为 30 位，试分析其与 32 位 PC 的优劣。

PC 使用 30 位时，忽略 32 位最低的两位，32 位 PC 加 4 实际上是 30 位 PC 加 1，而 ROM 两个相邻储存单元地址相差 1，用 30 位符合 logism 的情况，实际上 ROM 两个相邻储存单元地址相差 4，使用 32 位符合实际。

2、现在我们的模块中 IM 使用 ROM， DM 使用 RAM， GRF 使用寄存器，这种做法合理吗？ 请给出分析，若有改进意见也请一并给出。

合理，指令执行中，IM 不会被写入数据，只会被读取指令，使用 ROM 合适。指令执行中，DM 有数据读写，因此用 RAM 合适。GRF 使用寄存器对应在 MIPS 指令中的 32 个通用寄存器，合理。

3、结合上文给出的样例真值表，给出 RegDst， ALUSrc， MemtoReg，RegWrite, nPC_Sel, ExtOp 与 op 和 func 有关的布尔表达式（表达式中只能使用“与、或、非”3 种基本逻辑运算。）

RegDst=~op0~op1~op2~op3~op4~op5

ALUSrc=op0~op1op2op3~op4~op5+op0op1~op2~op3~op4op5+op0op1~op2op3~op4op5

MemtoReg=op0op1~op2~op3~op4op5

RegWrite=~op0~op1~op2~op3~op4~op5+op0~op1op2op3~op4~op5+op0op1~op2~op3~op4op5

nPC_Sel=~op0~op1op2~op3~op4~op5

ExtOp=op0op1~op2~op3~op4op5+op0op1~op2op3~op4op5

令 ALUctr[2:0]:

00　Add

01　Subtract

10　Or

AlUctr0=~op0~op1~op2~op3~op4~op5~func0func1~func2~func3~func4func5+~op0~op1op2~op3~op4~op5

AlUctr1=op0~op1op2op3~op4~op5

4、充分利用真值表中的 X 可以将以上控制信号化简为最简单的表达式， 请给出化简后的形式。

RegDst=~op0~op1~op2~op3~op4~op5

ALUSrc=op0~op1op2op3~op4~op5+op0op1~op2~op3~op4op5+op0op1~op2op3~op4op5

MemtoReg=op0op1~op2~op3~op4op5

RegWrite=~op0~op1~op2~op3~op4~op5+op0~op1op2op3~op4~op5+op0op1~op2~op3~op4op5

nPC_Sel=~op0~op1op2~op3~op4~op5

ExtOp=op0op1~op2~op3~op4op5+op0op1~op2op3~op4op5

令 ALUctr[2:0]:

00　Add

01　Subtract

10　Or

AlUctr0=~op0~op1~op2~op3~op4~op5~func0func1~func2~func3~func4func5+~op0~op1op2~op3~op4~op5

AlUctr1=op0~op1op2op3~op4~op5

5、事实上，实现 nop 空指令，我们并不需要将它加入控制信号真值表，为什么？请给出你的理由。

nop 指令为 0x00000000,如果不加入控制信号真值表，当 IFU 输出 nop 时，寄存器写使能信号 RegWrite、数据储存器写使能信号 MemWrite、npc_sel 都为无效，nop 指令实际上无法改变 GRF、DM、PC 中的数据，所以不用考虑。

6、前文提到，“可能需要手工修改指令码中的数据偏移”，但实际上只需再增加一个 DM 片选信号,就可以解决这个问题。请阅读相关资料并设计一个 DM 改造方案使得无需手工修改数据偏移。

由于 DM、IM 仅仅取了 5 位地址，当数据偏移量过大，可能出现 2 个 32 位地址所取 5 位地址相同，但实际上两个地址不同的情况。可将除 5 位地址外的剩余地址作为片选信号，同时增加 IM、DM 片数。

7、除了编写程序进行测试外，还有一种验证 CPU 设计正确性的办法——形式验证。 形式验证的含义是根据某个或某些形式规范或属性，使用数学的方法证明其正确性或非正确性。请搜索“形式验证（Formal Verification)"了解相关内容后，简要阐述相比与测试，形式验证的优劣。

形式验证优点：运用数学理论方法，对所有可能进行验证，验证时间短。
形式验证缺点：不能有效的验证电路的实际性能，如电路的时延和功耗等。