P3 单周期CPU设计

一、设计草稿

1.数据通路模块建模

（1）IFU（取指令单元）

（a）PC（程序计数器）

信号	方向	描述
CLK	I	时钟信号
PCReset	I	异步复位信号
NPC[31:0]	I	下轮指令地址32位输入
PC[31:0]	O	本轮指令地址32位输出

实现方法：32位寄存器

（b）IM（指令储存器）

信号	方向	描述
A[31:0]	I	指令地址32位输入
RD[31:0]	O	指令32位输出

实现方法：将32位地址取出2-6位，再访问ROM储存的地址，ROM规格5*32

（c）NPC

信号	方向	描述
PC[31:0]	I	本轮指令地址32位输入
imm[15:0]	I	分支偏移量16位输入
Br	I	是否为分支指令控制
Zero	I	rt与rs是否相等控制
NPC[32:0]	O	下轮指令地址32位输出

实现方法：若Br=1且Zero=1，则NPC=PC+4+sign_ext(imm||00)，否则NPC=PC+4

（2）GRF（通用寄存器文件）

（a）GRF

信号	方向	描述
CLK	I	时钟信号
RFReset	I	异步复位信号
WE	I	写使能信号
A1[4:0]	I	读取数据一地址5位输入
A2[4:0]	I	读取数据二地址5位输入
A3[4:0]	I	写入数据地址5位输入
WD[31:0]	I	写入数据32位输入
RD1[31:0]	O	读取数据一32位输出
RD2[31:0]	O	读取数据二32位输出

实现方法：使用DMX解析地址访问REG写入数据，使用MUX选择寄存器读取数据并输出

（b）REG（寄存器管理文件）

便于管理寄存器设置的子电路，与GRF密切联系，描述略

（3）ALU（算数逻辑单元）

信号	方向	描述
SA[31:0]	I	数据一32位输入
SB[31:0]	I	数据二32位输入
ALUOp[2:0]	I	计算控制信号
Zero	O	是否相等输出
ALUResult[31:0]	O	计算结果32位输出

实现方法：利用MUX、Adder、Subtractor和逻辑门电路实现

（4）DM（数据储存器）

信号	方向	描述
CLK	I	时钟信号
A[31:0]	I	读取/写入地址32位输入
WD[31:0]	I	写入数据32位输入
Wr	I	写使能信号
DMReset	I	异步复位信号
RD[31:0]	O	读取数据32位输出

实现方法：利用RAM

（5）EXT（扩展单元）

信号	方向	描述
imm[15:0]	I	待扩展的立即数16位输入
EXTOp[1:0]	I	扩展方式控制信号
EXTOut[31:0]	O	扩展结果32位输出

实现方法：利用MUX和Bit Extender

2.数据通路组装与指令建模

经指令建模发现GRF.A3、GRF.WD、ALU.SB需要MUX

3.控制模块建模

（1）指令真值表

指令	opcode	funct	Br	WRSel	WDSel	EXTOp	RFWE	BSel	ALUOp	DMWR
add	000000	100000	0	1	0	xx	1	0	000	0
sub	000000	100010	0	1	0	xx	1	0	001	0
ori	001101	xxxxxx	0	0	0	00	1	1	010	0
lw	100011	xxxxxx	0	0	1	01	1	1	011	0
sw	101011	xxxxxx	0	x	x	01	0	1	100	1
lui	001111	xxxxxx	0	0	0	10	1	1	101	0
nop	000000	000000	0	x	x	xx	0	0	xxx	0
beq	000100	xxxxxx	1	x	x	xx	0	0	xxx	0

（2）CTRL（控制器）模块建模

信号	方向	描述
opcode[5:0]	I	指令opcode6位输入
funct[5:0]	I	指令funct6位输入
Br	O	是否为分支指令信号
WRSel	O	GRF.A3选择rt或rd信号
WDSel	O	GRF.WD选择ALUResult或DMOut信号
EXTOp[1:0]	O	EXT选择零扩展或符号扩展或低位扩展信号
RFWE	O	GRF写使能信号
BSel	O	ALU.B选择EXTOut或GRF.RD2信号
ALUOp[2:0]	O	ALU不同指令运算控制信号
DMWR	O	DM写使能信号

实现方法：与门阵列和或门阵列

二、思考题

1.上面我们介绍了通过 FSM 理解单周期 CPU 的基本方法。请大家指出单周期 CPU 所用到的模块中，哪些发挥状态存储功能，哪些发挥状态转移功能。

状态储存功能：PC、GRF、DM

状态转移功能：NPC、ALU

2.现在我们的模块中 IM 使用 ROM， DM 使用 RAM， GRF 使用 Register，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。

IM储存的是程序的代码段，在程序运行过程中不会进行改变，但是需要读取信息，因此适合使用只读不写的ROM。

DM储存的是程序的数据段，在程序运行过程中既需要读取也需要写入，同时需要的访问空间比较大，且只能同时读取一个值，正好适合使用可读可写且空间大的RAM。

GRF储存的是程序运行过程中寄存器的值，在程序运行过程中需要频繁读取和写入，同时需要支持同时三次寻址，但寄存器的数量也是确定的，因此适合使用Register。

3.在上述提示的模块之外，你是否在实际实现时设计了其他的模块？如果是的话，请给出介绍和设计的思路。

否。

不过其中指令分线的部分可以封装为一个模块，其功能是将32位指令分线为6位opcode、5位rs、5位rt、5位rd、5位shamt、6位funct，并把低16位合并为16位imm。

4.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？

因为事实上nop指令对控制器中的控制信号控制输出为0，与不加入真值表控制信号输出相同，均不对GRF和DM进行改变。

5.上文提到，MARS 不能导出 PC 与 DM 起始地址均为 0 的机器码。实际上，可以避免手工修改的麻烦。请查阅相关资料进行了解，并阐释为了解决这个问题，你最终采用的方法。

可以为PC寄存器赋初值为0x00003000。

6.阅读 Pre 的 “MIPS 指令集及汇编语言”一节中给出的测试样例，评价其强度（可从各个指令的覆盖情况，单一指令各种行为的覆盖情况等方面分析），并指出具体的不足之处。

强度较弱。

指令覆盖情况：测试不同指令间先后顺序改变的情况

指令行为分析：

数据方面：未测试绝对值较大的边界值的情况

行为方面：

立即数指令：未测试偏移量和基数负数的情况

分支跳转指令：未测试目标在此指令和此指令之前的情况