**流水线cpu设计文档**

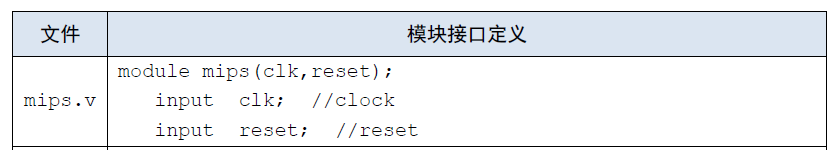
王郁含 16182672

**目录**

1. 设计与测试说明
2. 模块规格
3. 数据通路设计
4. 控制器设计
5. 测试程序
6. 思考题
   1. 设计与测试说明

* 处理器应支持指令集为：{addu, subu, ori, lw, sw, beq, lui, j, jal, jr,nop}。
* addu,subu可以不支持溢出。
* 处理器为流水线设计。
* 需要考虑延迟槽。
* 顶层文件为mips.v，接口定义如下：

表1 模块接口定义



1. 模块规格
   1. IFU（取指令单元）
   * 包括 PC （程序计数器）、IM（指令存储器），及相关逻辑。
   * IM容量为4KB（32bit×1024字）。
   * 端口定义：

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| clk | I | 时钟信号 |
| reset | I | 复位信号 |
| enable | I | 使能端 |
| Npc | I | 是否取npc |
| NPC | I | Npc的值 |
| [31:0]PC | O | 当前PC值 |
| [31:0]PC4 | O | 下一个PC值 |
| [31:0]PC8 | 0 | PC+8 |
| Instr | O | 32位指令信号 |

* 功能描述：

表3 IFU功能描述

|  |  |
| --- | --- |
| 功能 | 功能描述 |
| 复位清零 | 当时钟上升沿到来时，如果复位信号有效，则清零。 |
| 取下一条地址上的指令 | 当时钟上升沿到来且跳转控制失效时，PC + 4以获取下一地址，使IM读取下一条指令 |
| 转移至对应地址获取指令 | 当时钟上升沿到来且转移控制有效，转移至目标地址获取该地址内指令 |

2. GRF（通用寄存器组）

* + 用32个具有写使能的寄存器实现。
  + 0号寄存器保持为0。
  + 端口定义：

表4 GRF端口定义

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [4:0]RA1 | I | 读取寄存器1 |
| [4:0]RA2 | I | 读取寄存器2 |
| [4:0]WA | I | 写入寄存器地址 |
| [31:0]WD | I | 写入数据 |
| Reset | I | 复位信号 |
| clk | I | 时钟信号 |
| RWE | I | 写使能信号 |
| [31:0]PC | I | 当前指令信号 |
| [31:0]D1 | O | 读取数据1 |
| [31:0]D2 | O | 读取数据2 |

* + 功能描述：

表5 GRF 功能描述

|  |  |
| --- | --- |
| 功能 | 功能描述 |
| 复位清零 | 当时钟上升沿到来时，如果复位信号有效，则清零。 |
| 读取寄存器中的数据 | 当时钟上升沿到来且写使能WE失效时，读取RA1和RA2两个对应号的寄存器中的数据并分别输出至RD1和RD2 |
| 读取寄存器中的数据并将数据写入寄存器 | 当时钟上升沿到来且写使能WE有效时，读取RA1和RA2两个对应号的寄存器中的数据并分别输出至RD1和RD2同时将WD的数据写入WA对应号的寄存器 |

3. ALU（算术逻辑单元）

* + 提供32位加、减、或运算及大小比较功能。
  + 可以不支持溢出。
  + 端口定义：

表6 ALU 端口定义

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [31:0] A | I | 32位被运算数据 |
| [31:0] B | I | 32位被运算数据 |
| [1:0] ALUOp | I | 功能选择信号  00:加运算  01:减运算  10:按位或运算 |
| [31:0] ALUOut | O | 计算结果 |

* + 功能描述：

表7 ALU 功能描述

|  |  |
| --- | --- |
| 功能 | 功能描述 |
| 加运算 | ALUOut = A + B |
| 减运算 | ALUOut = A - B |
| 按位或运算 | ALUOut = A | B |

1. DM（数据存储器）

* DM容量为4KB（32bit×1024字）。
* 端口定义：

表8 DM 端口定义

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [4:0] Add | I | 访问的地址 |
| [31:0] WD | I | 写入地址的数据 |
| DWE | I | 写入使能端 |
| DRE | I | 读取使能端 |
| Reset | I | 复位信号 |
| Clk | I | 时钟信号 |
| [31:0] RD | O | 读取地址的数据 |

* 功能描述

表9 DM 功能描述

|  |  |
| --- | --- |
| 功能 | 功能描述 |
| 复位清零 | 当时钟上升沿时，如果信号有效，则置零 |
| 写入地址 | 当时钟上升沿且WE为1时在Addr所指地址中写入WD所代表数据 |
| 读取地址 | 当时钟上升沿且且RE为1时将Addr所指地址中的数据读出至RD |

1. EXT(位数扩展器)

* 端口定义：

表10 EXT 端口定义

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [15:0] Imm | I | 16位立即数/偏移量输入 |
| [1:0]EXTOp | I | EXT控制端：  00：无符号扩展  01：有符号扩展  10: 左移16位 |
| [31:0] EXT | O | 扩展后的32位立即数/偏移量输出 |

* 功能描述：

表11 EXT 功能描述

|  |  |
| --- | --- |
| 功能 | 功能描述 |
| 无符号扩展 | 将16位输入进行无符号扩展至32位 |
| 有符号扩展 | 当指令为sw/lw时，对16位输入进行有符号扩展至32位 |
| 左移16位 | 当指令为lui时，将16位输入扩展至32位后移至高16位 |

6. CMP(比较器)

* 用于beq指令前置比较的时候进行数据转发。
* 端口定义：

表12 CMP 端口定义

|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [31:0] A | I | 第一个取出来的数 |
| [31:0] B | I | 第二个取出来的数 |
| ZERO | O | 相等时为1，不等时为0 |

* 功能描述：

|  |  |
| --- | --- |
| 功能 | 功能描述 |
| 比较两数大小 | 相等时为1，不等时为0，有助于构造数据转发通路 |

7. NPC(地址计算器)

* 用于计算下一条指令的地址。
* 端口定义：

表13 NPC端口定义

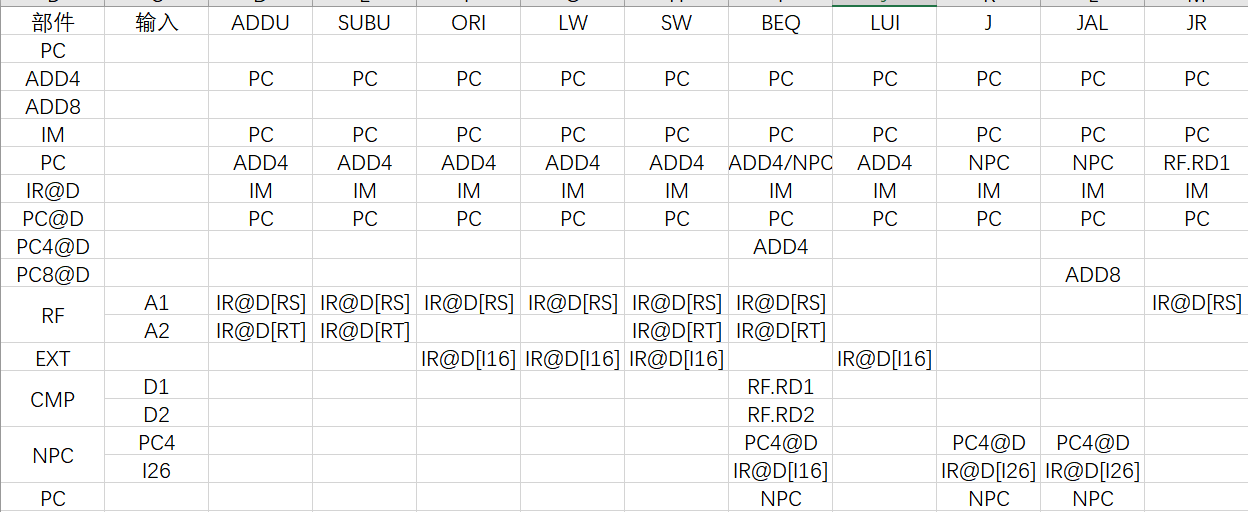
|  |  |  |
| --- | --- | --- |
| 信号名 | 方向 | 描述 |
| [31:0] PC4 | I | 下一条指令的物理地址 |
| [25:0] IMM26 | I | 可能会被用到的扩展数 |
| [31:0] JRPC | I | Jr返回的地址 |
| zero | I | Beq相等的判断 |
| [31:0]NPC | O | 输出下一条地址的pc值 |

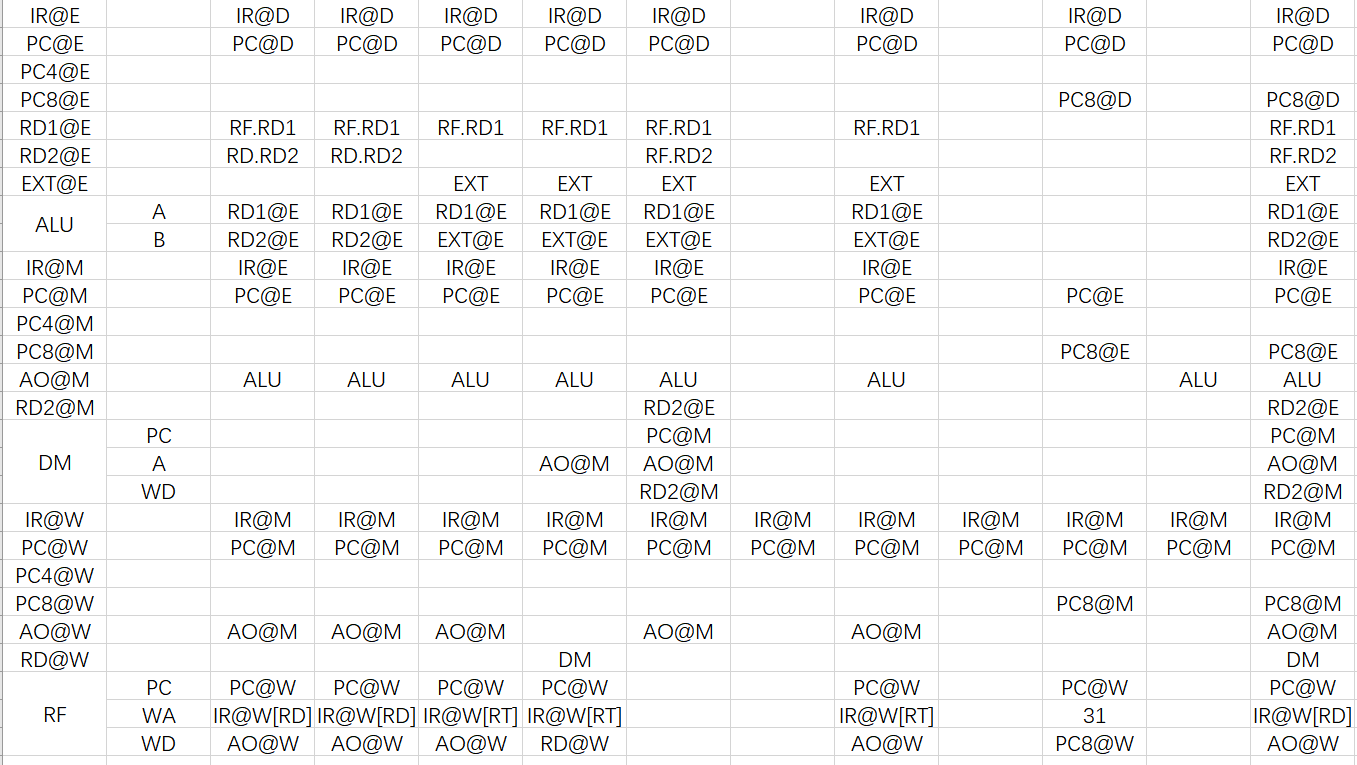
* 功能描述

表14 NPC功能描述

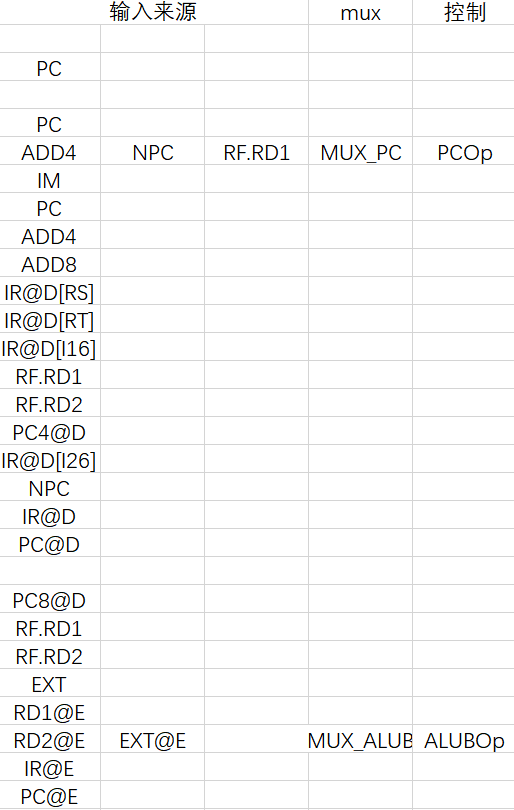
|  |  |
| --- | --- |
| 功能 | 功能描述 |
| 跳转至beq指令地址 | 根据imm26和pc4计算出beq地址，根据zero判断是否跳转 |
| 跳转至j/jal指令地址 | 根据imm26算出的跳转地址 |
| 跳转至jr指令地址 | 根据jrpc可直接得到 |

1. 数据通路设计





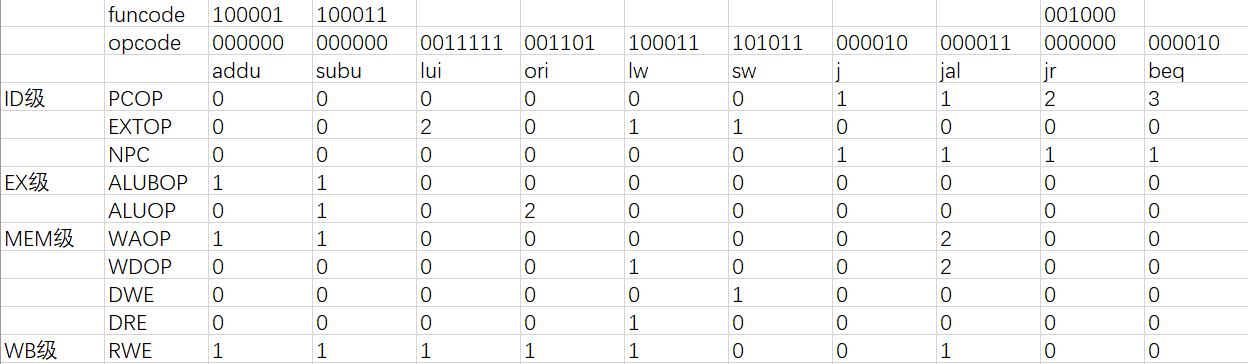
整合来源



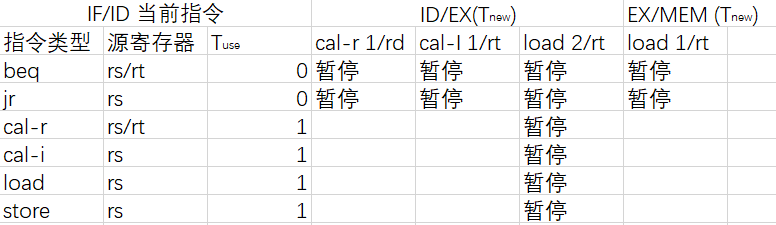


四、控制器设计

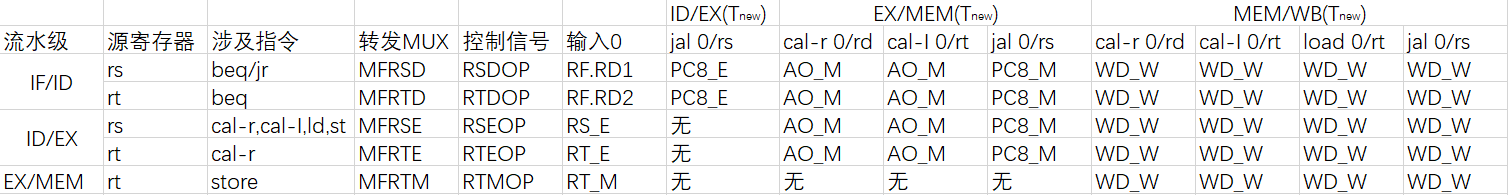
1. 功能部件控制信号



2. 暂停控制



3. 转发控制



4. 数据冒险控制

通过延迟槽解决。

五、测试程序

1. 不包含暂停/转发的测试程序

Address Code Basic Source

0x00003000 0x34210064 ori $1,$1,0x00000064 1 ori $1, $1, 100

0x00003004 0x340200c8 ori $2,$0,0x000000c8 2 ori $2, $0, 200

0x00003008 0x3480012c ori $0,$4,0x0000012c 3 ori $0, $4, 300

0x0000300c 0x3c050064 lui $5,0x00000064 4 lui $5, 100

0x00003010 0x3c0600c8 lui $6,0x000000c8 5 lui $6, 200

0x00003014 0x10000007 beq $0,$0,0x00000007 6 beq $0, $0, label1

0x00003018 0x00000000 nop 7 nop

0x0000301c 0xac270004 sw $7,0x00000004($1) 9 sw $7, 4($1)

0x00003020 0x00430823 subu $1,$2,$3 10 subu $1, $2, $3

0x00003024 0x00680023 subu $0,$3,$8 11 subu $0, $3, $8

0x00003028 0x000a4823 subu $9,$0,$10 12 subu $9, $0, $10

0x0000302c 0x03e00008 jr $31 13 jr $31

0x00003030 0x00000000 nop 14 nop

0x00003034 0x00210821 addu $1,$1,$1 16 addu $1, $1, $1

0x00003038 0x00430021 addu $0,$2,$3 17 addu $0, $2, $3

0x0000303c 0x00052021 addu $4,$0,$5 18 addu $4, $0, $5

0x00003040 0x0c000c07 jal 0x0000301c 19 jal label2

0x00003044 0x00000000 nop 20 nop

0x00003048 0x10010005 beq $0,$1,0x00000005 21 beq $0, $1, exit

0x0000304c 0x00000000 nop 22 nop

0x00003050 0x00000000 nop 23 nop

0x00003054 0x08000c17 j 0x0000305c 24 j label3

0x00003058 0x00000000 nop 25 nop

0x0000305c 0x8c610000 lw $1,0x00000000($3) 27 lw $1, 0($3)

0x00003060 0x1000ffff beq $0,$0,0xffffffff 29 beq $0, $0, exit

2. 暂停/转发测试程序

Address Code Basic Source

0x00003000 0x34210064 ori $1,$1,0x00000064 1 ori $1, $1, 100

0x00003004 0x1001001b beq $0,$1,0x0000001b 2 beq $0, $1, label

0x00003008 0x00000000 nop 3 nop

0x0000300c 0x00411021 addu $2,$2,$1 4 addu $2, $2, $1

0x00003010 0x10020018 beq $0,$2,0x00000018 5 beq $0, $2, label

0x00003014 0x00000000 nop 6 nop

0x00003018 0xac010000 sw $1,0x00000000($0) 7 sw $1, 0($0)

0x0000301c 0x00000000 nop 8 nop

0x00003020 0x00000000 nop 9 nop

0x00003024 0x8c030000 lw $3,0x00000000($0) 10 lw $3, 0($0)

0x00003028 0x10030012 beq $0,$3,0x00000012 11 beq $0, $3, label

0x0000302c 0x00000000 nop 12 nop

0x00003030 0x8c040000 lw $4,0x00000000($0) 13 lw $4, 0($0)

0x00003034 0x00802823 subu $5,$4,$0 14 subu $5, $4, $0

0x00003038 0x1080000e beq $4,$0,0x0000000e 15 beq $4, $0, label

0x0000303c 0x0c000c11 jal 0x00003044 16 jal ja

0x00003040 0x00000000 nop 17 nop

0x00003044 0x3c010000 lui $1,0x00000000 19 addu $31, $31, 20

0x00003048 0x34210014 ori $1,$1,0x00000014

0x0000304c 0x03e1f821 addu $31,$31,$1

0x00003050 0x03e00008 jr $31 20 jr $31

0x00003054 0xac1f0000 sw $31,0x00000000($0) 21 sw $31, 0($0)

0x00003058 0x8e810000 lw $1,0x00000000($20) 22 lw $1, 0($20)

0x0000305c 0x3c010000 lui $1,0x00000000 23 addu $2, $1, 12

0x00003060 0x3421000c ori $1,$1,0x0000000c

0x00003064 0x00211021 addu $2,$1,$1

0x00003068 0x00400008 jr $2 24 jr $2

0x0000306c 0x8e820000 lw $2,0x00000000($20) 26 lw $2, 0($20)

0x00003070 0xae820004 sw $2,0x00000004($20) 27 sw $2, 4($20)

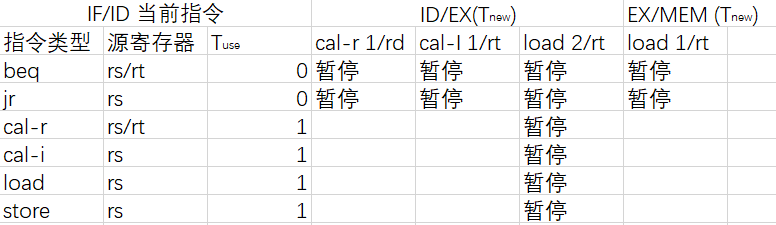
0x00003074 0x1000ffff beq $0,$0,0xffffffff 29 beq $0, $0, label

六、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。

1）暂停

下图为需要暂停的指令类型



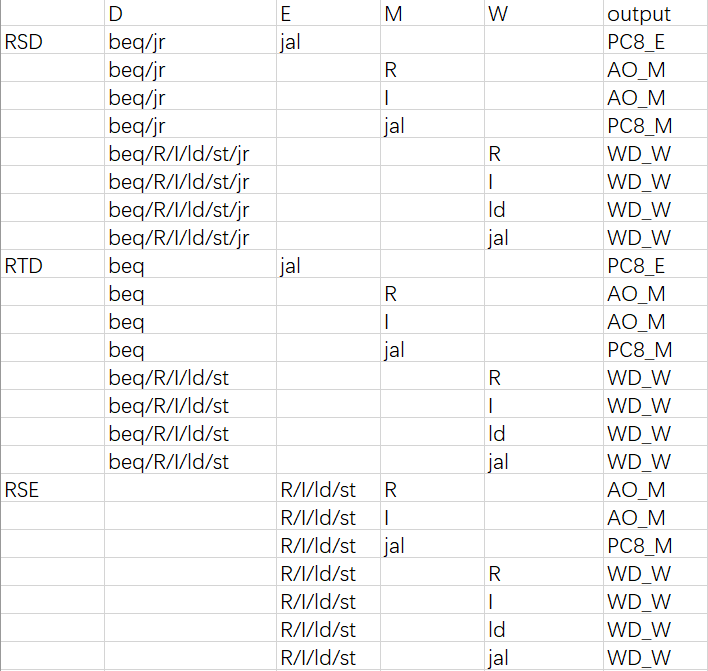
冻结IF/ID：sub继续被保存。

清除ID/EX：指令全为0，等价于插入NOP。

禁止PC：防止PC继续计数，PC应保持为PC+4。

2）转发

如下图所示，所需要的转发的数据通路。



根据这些数据通路添加多路选择器，根据表中不同的选择产生控制信号。完成数据转发。