流水线CPU

设计文档（P5）

贾博驿

初稿：2023年11月12日

编辑：2024年8月27日

设计实现指令

**共42条**

**实际指令39条**

R型指令：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 6 | 5 | 5 | 5 | 5 | 6 |
| opcode | rs | rt | rd | shamt | funct |

I型指令：

|  |  |  |  |
| --- | --- | --- | --- |
| 6 | 5 | 5 | 16 |
| opcode | rs (base) | rt | imm16 (immediate, offset) |

J型指令：

|  |  |
| --- | --- |
| 6 | 26 |
| opcode | instr\_index |

注：以下的RTL除分支与跳转类指令均省略PC←PC+4。

**算数与逻辑类**

1. 算数类
2. R型

|  |  |  |
| --- | --- | --- |
| 名称 | funct | RTL |
| add | 100000 | GPR[rd] ← GPR[rs] + GPR[rt] |
| addu | 100001 | GPR[rd] ← GPR[rs] + GPR[rt] |
| sub | 100010 | GPR[rd] ← GPR[rs] - GPR[rt] |
| subu | 100011 | GPR[rd] ← GPR[rs] - GPR[rt] |

注：实现的add实际上为addu、sub实际上为subu。二者均按照无符号指令处理。

1. I型

|  |  |  |
| --- | --- | --- |
| 名称 | opcode | RTL |
| addi | 001000 | GPR[rt] ← GPR[rs] + sign\_extend(immediate) |
| addiu | 001001 | GPR[rt] ← GPR[rs] + sign\_extend(immediate) |

注：实现的addi实际上为addiu。按照无符号指令处理。

1. 逻辑类
2. R型

|  |  |  |
| --- | --- | --- |
| 名称 | funct | RTL |
| and | 100100 | GPR[rd] ← GPR[rs] AND(bitwise) GPR[rt] |
| or | 100101 | GPR[rd] ← GPR[rs] OR(bitwise) GPR[rt] |
| xor | 100110 | GPR[rd] ← GPR[rs] XOR(bitwise) GPR[rt] |
| nor | 100111 | GPR[rd] ← GPR[rs] NOR(bitwise) GPR[rt] |
| sll | 000000 | GPR[rd] ← GPR[rt](31-shamt)..0 || 0shamt |
| srl | 000010 | GPR[rd] ← 0shamt || GPR[rt]31..shamt |
| sra | 000011 | GPR[rd] ← (GPR[rt]31)shamt || GPR[rt]31..shamt |
| sllv | 000100 | s ← GPR[rs]4..0 GPR[rd] ← GPR[rt](31-s)..0 || 0s |
| srlv | 000110 | s ← GPR[rs]4..0 GPR[rd] ← 0s || GPR[rt]31..s |
| srav | 000111 | s ← GPR[rs]4..0 GPR[rd] ← (GPR[rt]31)s || GPR[rt]31..s |

注：根据寄存器值移位指令只依照寄存器值的低五位移位，不受高位值影响。

1. I型

|  |  |  |
| --- | --- | --- |
| 名称 | opcode | RTL |
| andi | 001100 | GPR[rt] ← GPR[rs] AND(bitwise) zero\_extend(immediate) |
| ori | 001101 | GPR[rt] ← GPR[rs] OR(bitwise) zero\_extend(immediate) |
| xori | 001110 | GPR[rt] ← GPR[rs] XOR(bitwise) zero\_extend(immediate) |

**寄存器操作类**

1. R型

|  |  |  |
| --- | --- | --- |
| 名称 | funct | RTL |
| slt | 101010 | GPR[rd] ← 0GPRLEN-1 || GPR[rs] < GPR[rt] |
| sltu | 101011 | GPR[rd] ← 0GPRLEN-1 || (0 || GPR[rs]) < (0 || GPR[rt]) |

注：slt类指令成立时置1，**不成立时置0**。

1. I型

|  |  |  |
| --- | --- | --- |
| 名称 | opcode | RTL |
| slti | 001010 | GPR[rt] ← 0GPRLEN-1 || GPR[rs] < sign\_extend(immediate) |
| sltiu | 001011 | GPR[rt] ← 0GPRLEN-1 || (0 || GPR[rs]) < (0 || **sign\_extend**(immediate)) |
| lui | 001111 | GPR[rt] ← immediate || 016 |

注：sltiu对立即数的处理是先**有符号扩展**再进行**无符号比较**。

**分支与跳转类**

1. 分支类（I型）

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | opcode | rt | RTL |
| (all) |  |  | **I+1:** if condition then  PC ← PC + sign\_extend(offset || 02)  endif |
| bltz | 000001 | 00000 | **I:** condition ← GPR[rs] < 032 |
| bgez | 000001 | 00001 | **I:** condition ← GPR[rs] ≥ 032 |
| beq | 000100 | rt | **I:** condition ← (GPR[rs] = GPR[rt]) |
| bne | 000101 | rt | **I:** condition ← (GPR[rs] ≠ GPR[rt]) |
| blez | 000110 | 00000 | **I:** condition ← GPR[rs] ≤ 032 |
| bgtz | 000111 | 00000 | **I:** condition ← GPR[rs] > 032 |

注：流水线CPU分支指令不需要PC + 4后再加上offset，直接在延迟槽的地址上加。

1. 跳转类
2. R型

|  |  |  |
| --- | --- | --- |
| 名称 | funct | RTL |
| jr | 001000 | **I:**  **I+1:** PC ← GPR[rs] |
| jalr | 001001 | **I:** GPR[rd] ← PC + 8  **I+1:** PC ← GPR[rs] |

1. J型

|  |  |  |
| --- | --- | --- |
| 名称 | opcode | RTL |
| j | 000010 | **I:**  **I+1:** PC ← PC31..28 || instr\_index || 02 |
| jal | 000011 | **I:** GPR[31] ← PC + 8  **I+1:** PC ← PC31..28 || instr\_index || 02 |

注：流水线CPU跳转指令由于延迟槽存在应该加上PC + 8，而非单周期的PC + 4。

**内存操作类（I型）**

|  |  |  |
| --- | --- | --- |
| 名称 | opcode | RTL |
| (all) |  | addr ← sign\_extend(offset) + GPR[base] |
| lb | 100000 | byte ← addr1..0  GPR[rt] ← sign\_extend(memory[addr]7+8\*byte..8\*byte) |
| lh | 100001 | half ← addr1  GPR[rt] ← sign\_extend(memory[addr]15+8\*half..8\*half) |
| lw | 100011 | GPR[rt] ← memory[addr] |
| lbu | 100100 | byte ← addr1..0  GPR[rt] ← zero\_extend(memory[addr]7+8\*byte..8\*byte) |
| lhu | 100101 | half ← addr1  GPR[rt] ← zero\_extend(memory[addr]15+8\*half..8\*half) |
| sb | 101000 | memory[addr]7..0 ← GPR[rt]7..0 |
| sh | 101001 | memory[addr]15..0 ← GPR[rt]15..0 |
| sw | 101011 | memory[addr]31..0 ← GPR[rt] |

注：

1.针对半字操作需要addr为2的非负整数倍，针对字节操作需要addr为4的非负整数倍，本次实现时实际只涉及针对字节操作，且保证地址是合法的。

2.针对字节和半字操作时只修改对应字节和半字的值，**同一字的其他部分不变**。

转发与暂停设计

**指令分类**

|  |  |  |  |
| --- | --- | --- | --- |
| 类型名 | 信号名 | 数量 | 指令 |
| 算数逻辑寄存器 | ALREG | 16 | \_arith（算数）：add, addu, sub, subu  \_logic（逻辑）：and, or, xor, nor  \_shift（移位）：sll, srl, sra, sllv, srlv, srav  \_comp（比较）： slt, sltu |
| 算数逻辑立即数 | ALIMM | 7 | \_sign（有符号拓展）：addi, addiu, slti, sltiu  \_zero（无符号拓展）：andi, ori, xori |
| 高位加载 | EXTLUI | 1 | lui |
| 两数比较分支 | BRANCHE | 2 | beq, bne |
| 与零比较分支 | BRANCHZ | 4 | bltz, bgez, blez, bgtz |
| 仅读取跳转 | JUMPR | 1 | jr |
| 仅写入跳转 | JUMPW | 1 | jal |
| 读取写入跳转 | JUMPRW | 1 | jalr |
| 读取内存 | LOAD | 5 | lb, lh, lw, lbu, lhu |
| 写入内存 | STORE | 3 | sb, sh, sw |

注：不包括j指令，因为j指令不读取通用寄存器也不写入通用寄存器。

**Tuse与Tnew**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 信号名 | Tuse\_rs | Tuse\_rt |  | GRF\_WD\_W\_Sel | Tnew\_E | Tnew\_M | Tnew\_W |
| ALREG | 1 | 1 |  | 00 | 1 | 0 | 0 |
| ALIMM | 1 | (3) |  | 00 | 1 | 0 | 0 |
| EXTLUI | (3) | (3) |  | 10 | 0 | 0 | 0 |
| BRANCHE | 0 | 0 |  | (00) | (0) | (0) | (0) |
| BRANCHZ | 0 | (3) |  | (00) | (0) | (0) | (0) |
| JUMPR | 0 | (3) |  | (00) | (0) | (0) | (0) |
| JUMPW | (3) | (3) |  | 11 | 0 | 0 | 0 |
| JUMPRW | 0 | (3) |  | 11 | 0 | 0 | 0 |
| LOAD | 1 | (3) |  | 01 | 2 | 1 | 0 |
| STORE | 1 | 2 |  | (00) | (0) | (0) | (0) |

注：

1.(3)表示该类指令不会读取rs或rt域翻译为寄存器编号后对应的寄存器，除GRF内部转发外不产生任何转发和暂停。

2.(00)和(0)表示表示该类指令GRFWE为0，虽然使用ALU数据但无意义。

**转发与暂停**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| SPL\_rs != 5’d0 && GRFWE\_X && SPL\_rs == GRF\_A3\_X | | | | | | | | | |
| 流水段(X) | E | | | | M (~ rs\_E\_premise) | | | | W |
| GRF\_WD\_W\_Sel\_X(Tnew\_X) | 00(1) | 01(2) | 10(0) | 11(0) | 00(0) | 01(1) | 10(0) | 11(0) | All |
| Tuse\_rs == 0 | S1 | S2 | F43 | F44 | F21 | S3 | F23 | F24 | F1 |
| Tuse\_rs == 1 | F51 | S4 | F32 |
| SPL\_rt != 5’d0 && GRFWE\_X && SPL\_rt == GRF\_A3\_X | | | | | | | | | |
| 流水段(X) | E | | | | M (~ rs\_E\_premise) | | | | W |
| GRF\_WD\_W\_Sel\_X(Tnew\_X) | 00(1) | 01(2) | 10(0) | 11(0) | 00(0) | 01(1) | 10(0) | 11(0) | All |
| Tuse\_rt == 0 | S1 | S2 | F43 | F44 | F21 | S3 | F23 | F24 | F1 |
| Tuse\_rt == 1 | F51 | S4 | F32 |
| Tuse\_rt == 2 | F62 |

注：

1.优先级**S > F4、F5、F6 > F2、F3。**

2.F2和F3之间、F4、F5和F6之间不会发生冲突，因为转发的是同一流水级的不同数据。

**转发信号表**

|  |  |  |  |
| --- | --- | --- | --- |
| SPL\_rs != 5’d0 && GRFWE\_X && SPL\_rs == GRF\_A3\_X | | | |
| 类型 | 描述 | FMUX\_V1\_D\_Sel | FMUX\_V1\_E\_Sel |
| S | 暂停 |  |  |
| F51 | V1\_E转发OP\_M |  | 11 |
| F43 | V1\_D转发ext32\_E | 110 |  |
| F44 | V1\_D转发pc8\_E | 111 |  |
| F32 | V1\_E转发DM\_Q\_W |  | 10 |
| F21 | V1\_D转发OP\_M | 011 |  |
| F23 | V1\_D转发ext32\_M | 100 |  |
| F24 | V1\_D转发pc8\_M | 101 |  |
| F1 | GRF内部转发 |  |  |
| F0 | 不转发 | 000 | 00 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SPL\_rt != 5’d0 && GRFWE\_X && SPL\_rt == GRF\_A3\_X | | | | |
| 类型 | 描述 | FMUX\_V2\_D\_Sel | FMUX\_V2\_E\_Sel | FMUX\_OP\_M\_Sel |
| S | 暂停 |  |  |  |
| F62 | DM\_D\_M转发DM\_Q\_W |  |  | 1 |
| F51 | V2\_E转发OP\_M |  | 11 |  |
| F43 | V2\_D转发ext32\_E | 110 |  |  |
| F44 | V2\_D转发pc8\_E | 111 |  |  |
| F32 | V2\_E转发DM\_Q\_W |  | 10 |  |
| F21 | V2\_D转发OP\_M | 011 |  |  |
| F23 | V2\_D转发ext32\_M | 100 |  |  |
| F24 | V2\_D转发pc8\_M | 101 |  |  |
| F1 | GRF内部转发 |  |  |  |
| 0 | 不转发 | 000 | 00 | 0 |

注：未标注的信号取值与不转发相同。

**暂停变化表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| I | I+1 | I+2 |  | I | I+1 | I+2 |
| S1 | F21 |  |  | S3 | F1 |  |
| S2 | S3 | F1 |  | S4 | F32 |  |

模块设计

**IFU**

1. 描述

取指令模块。内有PC子模块、NPC子模块和ROM。接收分支和跳转指令的控制信号和数据并实现跳转。输出指令及其地址。

1. 接口

|  |  |  |
| --- | --- | --- |
| 端口名 | 方向 | 描述 |
| RESET | I | 同步复位信号 |
| clk | I | 时钟信号 |
| STALL\_EN\_N | I | 暂停使能信号 |
| NPCSel[1:0] | I | NPC子模块控制信号 |
| BranchComp | I | NPC子模块分支指令控制信号 |
| offset[15:0] | I | 分支指令跳转偏移 |
| instr\_index[25:0] | I | j、jal指令跳转地址 |
| instr\_register[31:0] | I | jr、jalr指令跳转寄存器值 |
| Instr[31:0] | O | 即将进入流水线的指令 |
| InstrAddr[31:0] | O | 即将进入流水线的指令地址 |

1. 控制信号

BranchComp：接COMP的BranchComp。

NPCSel：

|  |  |  |
| --- | --- | --- |
| 指令（信号）名 | 取值 | 描述 |
| 其他 | 00 | PC←PC+4 |
| BRANCHE、BRANCHZ | 01 | 使用offset，结合BranchComp判断 |
| j、jal | 10 | 使用instr\_index |
| jr、jalr | 11 | 使用instr\_register |

**IFU\_PC**

1. 描述

程序计数器子模块。在复位信号有效时复位为32'h00003000。

1. 接口

|  |  |  |
| --- | --- | --- |
| 端口名 | 方向 | 描述 |
| RESET | I | 同步复位信号 |
| clk | I | 时钟信号 |
| STALL\_EN\_N | I | 暂停使能信号 |
| D[31:0] | O | 下一指令地址 |
| Q[31:0] | O | 即将进入流水线的指令地址 |

**IFU\_NPC**

1. 描述

下一条指令地址计算子模块。输入IFU接收的分支跳转指令控制信号和数据，计算出下一条指令的地址。

1. 接口

|  |  |  |
| --- | --- | --- |
| 端口名 | 方向 | 描述 |
| PC[31:0] | 1 | 即将进入流水线的指令地址 |
| NPCSel[1:0] | I | 同IFU |
| BranchComp | I | 同IFU |
| offset[15:0] | I | 同IFU |
| instr\_index[25:0] | I | 同IFU |
| instr\_register[31:0] | I | 同IFU |
| NPC[31:0] | O | 下一指令地址 |

**SPL**

1. 描述

指令分线器模块。按照三种指令的格式将一条指令分为不同的部分，供其他模块使用。

1. 接口

|  |  |  |
| --- | --- | --- |
| 端口名 | 方向 | 描述 |
| Instr[31:0] | 1 | 当前指令 |
| opcode[5:0] | O | 指令的操作数 |
| rs[4:0] | O | R、I型指令的rs |
| rt[4:0] | O | R、I型指令的rt |
| rd[4:0] | O | R型指令的rd |
| shamt[4:0] | O | R型指令的shamt |
| funct[5:0] | O | R型指令的funct |
| imm16[15:0] | O | I 型指令的立即数 |
| instr\_index[25:0] | O | J型指令的跳转地址 |

**GRF**

**(GRF\_R, GRF\_W)**

1. 描述

寄存器堆模块，共有31个寄存器（0号寄存器直接接地）。通过A1、A2输入地址，可以取出指定寄存器的完成内部转发的数据。当WE使能时，通过A3输入地址，D输入数据，可以修改指定寄存器存储的数据并进行内部转发。在流水线设计时分为两个部分体现。

1. 接口

|  |  |  |
| --- | --- | --- |
| 端口名 | 方向 | 描述 |
| RESET | I | 同步复位信号 |
| clk | I | 时钟信号 |
| WE | I | 写入使能信号 |
| A1[4:0] | I | 读取的第一个寄存器的编号 |
| A2[4:0] | I | 读取的第二个寄存器的编号 |
| A3[4:0] | I | 写入的寄存器的编号 |
| D[31:0] | I | 写入寄存器的数据 |
| V1[31:0] | O | 读取的第一个寄存器的完成内部转发的值 |
| V2[31:0] | O | 读取的第二个寄存器的完成内部转发的值 |

1. 控制信号

GRFWE：

|  |  |  |
| --- | --- | --- |
| 指令信号名 | 取值 | 描述 |
| 其他 | 0 | 不涉及寄存器写入的指令 |
| ALREG | 1 |  |
| ALIMM、EXTLUI | 1 |  |
| LOAD | 1 |  |
| JUMPW、JUMPRW | 1 |  |

1. 关联MUX

MUX\_GRF\_A3\_D，控制信号CTRL\_D\_GRF\_A3\_D\_Sel。

|  |  |  |
| --- | --- | --- |
| 指令信号名 | 取值 | 描述 |
| 其他 | 00 | 接入SPL\_rd |
| ALIMM、EXTLUI | 10 | 接入SPL\_rt |
| LOAD | 10 | 接入SPL\_rt |
| JUMPW | 11 | 固定32’d31 |

MUX\_GRF\_WD\_W，控制信号GRF\_WD\_W\_Sel\_W。

|  |  |  |
| --- | --- | --- |
| 指令信号名 | 取值 | 描述 |
| 其他 | 00 | 接入OP\_W |
| LOAD | 01 | 接入DM\_Q\_W |
| EXTLUI | 10 | 接入ext32\_W |
| JUMPW、JUMPRW | 11 | 接入pc8\_W |

**COMP**

1. 描述

比较模块，完成两数相等、不等，一数有符号小于、小于等于、大于、大于等于零的比较运算。比较结果用于判断分支指令是否执行。

1. 接口

|  |  |  |
| --- | --- | --- |
| 端口名 | 方向 | 描述 |
| CompSel[2:0] | I | 比较方式控制信号 |
| A[31:0] | I | 第一个操作数 |
| B[31:0] | I | 第二个操作数 |
| BranchComp | O | 比较结果 |

1. 控制信号

CompSel：

|  |  |  |
| --- | --- | --- |
| 指令名 | 取值 | 描述 |
| bltz(00000) | 000 | 有符号小于0 |
| bgez(00001) | 001 | 有符号大于等于0 |
| beq(000100) | 100 | 两数相等 |
| bne(000101) | 101 | 两数不等 |
| blez(000110) | 110 | 有符号小于等于0 |
| bgtz(000111) | 111 | 有符号大于0 |

**EXT**

1. 描述

位扩展模块，将16位立即数无符号扩展、有符号扩展、右补16位0为32位。

1. 接口

|  |  |  |
| --- | --- | --- |
| 端口名 | 方向 | 描述 |
| EXTSel[1:0] | I | 扩展方式控制信号 |
| imm16[15:0] | I | 16位立即数 |
| ext32[31:0] | O | 32位扩展结果 |

1. 控制信号

EXTSel：

|  |  |  |
| --- | --- | --- |
| 指令名 | 取值 | 描述 |
| 其他 | 00 | 有符号拓展 |
| andi、ori、xori | 01 | 无符号拓展 |
| lui | 10 | 右补16位0 |

**ALU**

1. 描述

算数逻辑模块，完成算数运算（加、减）、逻辑运算（与、或、异或、或非）、移位运算（左移、逻辑右移、算数右移）、比较运算（两数有符号小于，无符号小于）。计算结果用于保存到寄存器。

1. 接口

|  |  |  |
| --- | --- | --- |
| 端口名 | 方向 | 描述 |
| OPSel[1:0] | I | OP输出控制信号 |
| FuncSel[1:0] | I | 计算方式控制信号 |
| A[31:0] | I | 第一个操作数 |
| B[31:0] | I | 第二个操作数 |
| shamt[4:0] | I | 移位立即数 |
| OP[31:0] | O | 计算结果 |

1. 控制信号

OPSel：

|  |  |  |
| --- | --- | --- |
| 指令（信号）名 | 取值 | 描述 |
| add、addu、sub、subu、addi、addiu | 00 | 算数运算 |
| LOAD、STORE | 00 | 算数运算 |
| and、or、xor、nor、andi、ori、xori | 01 | 逻辑运算 |
| sll、srl、sra、sllv、srlv、srav | 10 | 移位运算 |
| slt、sltu、slti、sltiu | 11 | 比较运算 |

FuncSel：根据OPSel选择。

OPSel == 00时。

|  |  |  |
| --- | --- | --- |
| 指令名 | 取值 | 描述 |
| add、addu、addi、addiu | 00 | 不考虑算数溢出加法 |
| LOAD、STORE | 00 | 不考虑算数溢出加法 |
| sub、subu | 01 | 不考虑算数溢出减法 |

OPSel == 01时，根据指令是否为I型选择。R型接funct[1:0]、I型接opcode[1:0]。

|  |  |  |
| --- | --- | --- |
| 指令名 | 取值 | 描述 |
| and(100100)、andi(001100) | 00 | 与 |
| or(100101)、ori(001101) | 01 | 或 |
| xor(100110)、xori(001110) | 10 | 异或 |
| nor(100111) | 11 | 或非 |

OPSel == 10时，接funct[1:0]。

|  |  |  |
| --- | --- | --- |
| 指令名 | 取值 | 描述 |
| sll(000000)、sllv(000100) | 00 | 逻辑左移 |
| srl(000010)、srlv(000110) | 10 | 逻辑右移 |
| sra(000011)、srav(000111) | 11 | 算数右移 |

OPSel == 11时。

|  |  |  |
| --- | --- | --- |
| 指令名 | 取值 | 描述 |
| slt、slti | 00 | 有符号两数小于 |
| sltu、sltiu | 01 | 无符号两数小于 |

1. 关联MUX

MUX\_ALU\_B\_E，控制信号ALU\_B\_E\_Sel\_E。

|  |  |  |
| --- | --- | --- |
| 指令信号名 | 取值 | 描述 |
| 其他 | 0 | 接入FMUX\_V2\_E |
| ALIMM、LOAD、STORE | 1 | 接入ext32\_E |

**DM**

1. 描述

内存模块。包括RAM和位处理两个部分。位处理部分用于处理针对半字和字节的读取和写入。

1. 接口

|  |  |  |
| --- | --- | --- |
| 端口名 | 方向 | 描述 |
| RESET | I | 同步复位信号 |
| clk | I | 时钟信号 |
| WE | I | 写入使能信号 |
| DMSel[2:0] | I | 读取写入模式控制信号 |
| A[31:0] | I | 读取或写入的地址 |
| D[31:0] | I | 写入的数据 |
| Q[31:0] | O | 读取的数据 |

1. 控制信号

DMWE：

|  |  |  |
| --- | --- | --- |
| 指令信号名 | 取值 | 描述 |
| 其他 | 0 |  |
| STORE | 1 |  |

DMSel：接opcode[2:0]。

|  |  |  |
| --- | --- | --- |
| 指令名 | 取值 | 描述 |
| lb(100000)、sb(101000) | 000 | 针对byte操作，**读取时**有符号拓展 |
| lh(100001)、sh(101001) | 001 | 针对halfword操作，**读取时**有符号拓展 |
| lw(100011)、sw(101011) | 011 | 针对word操作 |
| lbu(100100) | 100 | 针对byte操作，读取时无符号拓展 |
| lhu(100101) | 101 | 针对halfword操作，读取时无符号拓展 |

**FR\_D, FR\_E, FR\_M, FR\_W**

1. 描述

D、E、M、W段流水线寄存器。

1. 接口

通用：

|  |  |  |
| --- | --- | --- |
| 端口名 | 方向 | 描述 |
| RESET | I | 同步复位信号 |
| clk | I | 时钟信号 |

以下端口均有I和O两个方向，I方向信号名以D\_开头，O方向信号名以Q\_开头。

FR\_D：

|  |  |
| --- | --- |
| 端口名 | 接入 |
| STALL\_EN\_N | CTRL\_S\_FR\_D\_EN |
| Instr[31:0] | IFU\_Instr |
| InstrAddr[31:0] | IFU\_InstrAddr |

FR\_E：

|  |  |
| --- | --- |
| 端口名 | 接入 |
| STALL\_RESET | CTRL\_S\_FR\_E\_RESET |
| DMWE | CTRL\_D\_DMWE |
| GRFWE | CTRL\_D\_GRFWE |
| DMSel[2:0] | CTRL\_D\_DMSel |
| FuncSel[1:0] | CTRL\_D\_FuncSel |
| OPSel[1:0] | CTRL\_D\_OPSel |
| GRF\_WD\_W\_Sel[1:0] | CTRL\_D\_GRF\_WD\_W\_Sel |
| ALU\_B\_E\_Sel | CTRL\_D\_ALU\_B\_E\_Sel |
| V1[31:0] | FMUX\_V1\_D |
| V2[31:0] | FMUX\_V2\_D |
| shamt[4:0] | SPL\_shamt |
| GRF\_A3[4:0] | MUX\_GRF\_A3\_D |
| ext32[31:0] | EXT\_ext32 |
| pc8[31:0] | pc8\_D |
| FMUX\_V1\_E\_Sel[1:0] | CTRL\_F\_ FMUX\_V1\_E\_Sel |
| FMUX\_V2\_E\_Sel[1:0] | CTRL\_F\_ FMUX\_V2\_E\_Sel |
| FMUX\_DM\_D\_M\_Sel | CTRL\_F\_ FMUX\_DM\_D\_M\_Sel |

FR\_M：

|  |  |
| --- | --- |
| 端口名 | 接入 |
| DMWE | DMWE\_E |
| GRFWE | GRFWE\_E |
| DMSel[2:0] | DMSel\_E |
| GRF\_WD\_W\_Sel[1:0] | GRF\_WD\_W\_Sel\_E |
| V2[31:0] | FMUX\_V2\_E |
| OP[31:0] | ALU\_OP |
| GRF\_A3[4:0] | GRF\_A3\_E |
| ext32[31:0] | ext32\_E |
| pc8[31:0] | pc8\_E |
| FMUX\_DM\_D\_M\_Sel | FMUX\_DM\_D\_M\_Sel\_E |

FR\_W：

|  |  |
| --- | --- |
| 端口名 | 接入 |
| GRFWE | GRFWE\_M |
| GRF\_WD\_W\_Sel[1:0] | GRF\_WD\_W\_Sel\_M |
| OP[31:0] | OP\_M |
| DM\_Q[31:0] | DM\_Q |
| GRF\_A3[4:0] | GRF\_A3\_ M |
| ext32[31:0] | ext32\_ M |
| pc8[31:0] | pc8\_ M |

**CTRL\_Decoder**

1. 描述

译码器模块。通过opcode、funct和rt判断指令的类型，输出各个模块的控制信号。

1. 接口

|  |  |  |  |
| --- | --- | --- | --- |
| 端口名 | 方向 | 端口名 | 方向 |
| opcode[5:0] | I | DMSel[2:0] | O |
| funct[5:0] | I | DMWE | O |
| rt[4:0] | I | GRFWE | O |
| CompSel[2:0] | O | GRF\_A3\_D\_Sel[1:0] | O |
| EXTSel[1:0] | O | GRF\_WD\_W\_Sel[1:0] | O |
| NPCSel[1:0] | O | ALU\_B\_E\_Sel | O |
| OPSel[1:0] | O | Tuse\_rs[1:0] | O |
| FuncSel[1:0] | O | Tuse\_rt[1:0] | O |

**CTRL\_Stall**

1. 描述

暂停控制器模块。判断是否需要暂停，若需要暂停则产生暂停信号。

1. 接口

|  |  |  |  |
| --- | --- | --- | --- |
| 端口名 | 方向 | 端口名 | 方向 |
| Tuse\_rs[1:0] | I | GRF\_WD\_W\_Sel\_E[1:0] | I |
| Tuse\_rt[1:0] | I | GRF\_WD\_W\_Sel\_M[1:0] | I |
| SPL\_rs[4:0] | I | GRF\_A3\_E[4:0] | I |
| SPL\_rt[4:0] | I | GRF\_A3\_M[4:0] | I |
| GRFWE\_E | I | IFU\_EN\_N | O |
| GRFWE\_M | I | FR\_D\_EN\_N | O |
|  |  | FR\_E\_RESET | O |

**CTRL\_Forward**

1. 描述

转发控制器模块。判断是否需要转发，若需要转发转发什么数据。

1. 接口

|  |  |  |  |
| --- | --- | --- | --- |
| 端口名 | 方向 | 端口名 | 方向 |
| Tuse\_rs[1:0] | I | GRF\_A3\_E[4:0] | I |
| Tuse\_rt[1:0] | I | GRF\_A3\_M[4:0] | I |
| SPL\_rs[4:0] | I | FMUX\_V1\_D\_Sel[2:0] | O |
| SPL\_rt[4:0] | I | FMUX\_V2\_D\_Sel[2:0] | O |
| GRFWE\_E | I | FMUX\_V1\_E\_Sel[1:0] | O |
| GRFWE\_M | I | FMUX\_V2\_E\_Sel[1:0] | O |
| GRF\_WD\_W\_Sel\_E[1:0] | I | FMUX\_DM\_D\_M\_Sel | O |
| GRF\_WD\_W\_Sel\_M[1:0] | I |  |  |