RV32I Pipeline CPU Design Report

Author: 王嵘晟

ID: PB17111614

设计：

1.

> NPC Generator: 对于输入的PC+4、Jal Target、Br Targent、Jalr Target，设置优先级Br>Jalr>Jal>PC+4完成对NPC Generator的更新

2.

> Immediate Extend模块应该引进控制信号，达到对于立即数扩展方式的控制的目的。从Controller Decoder引出的控制信号直接通过数据通路送至Immediate Extend中。为了满足在ID阶段一个总的时钟周期内完成相关操作，需要让这两个模块的内部时钟频率高于总线时钟周期

3.

> 对于CSR指令的处理详见下面问题4，在现有结构的基础上增加CSRs寄存器以及数个选择器、段间寄存器以完成设计

4.

> 对于分支跳转指令，在分支预测时使用连续两次预测均失败则更改预测算法

问题：

1.描述执行一条ADDI 指令的过程（数据通路、控制信号等）。

> IF阶段：首先NPC将当前指令地址传送给PC，PC将当前PC存储的地址分为两部分，一部分送入PC自增ALU里做+4处理作为下一条指令的地址，写回NPC。另一部分存入Instruction Cache中，进而进入IR，完成流水线的IF阶段。
> ID阶段：对于寄存器堆，Inst[19: 15]寻址到RS1，作为源寄存器。Inst[11: 7]通过Addr寄存器作为Rd。Inst[31: 7]传入立即数扩展，立即数扩展根据指令译码得到的控制信号，来确认需要扩展的是Inst[31: 20]这12位，得到立即数源操作数。Inst[31: 0]传入控制信号译码来实现，其中通过Inst[6: 0]这7位opcode和Inst[14: 12]这3位func 3来产生相应的控制信号。由于ADDI是立即数加法，所以两个操作数一个被翻译成寄存器另一个完成了立即数扩展。对于控制信号的译码决定了传给ALU的控制信号为加法操作，OP2 src信号选择Imm，另外有一部分控制信号传入立即数扩展模块中。OP1和OP2两个段间寄存器分别把ALU的源操作数通过Reg1和Reg2OrImm端口送入。
> EX阶段：将ALU的两个源操作数输入，然后根据控制信号ALU Func判断要执行加操作，运算结束后得到的结果从ALU out传出，进入二选一选择器，根据控制信号选出ALU out，存入Result寄存器。
> M阶段：由于是立即数加法指令I-type，所以没有访问内存阶段，不执行任何操作
> WB阶段：首先由WB set控制将Rd的地址传入寄存器堆，然后将ALU out写入寄存器堆中对应Rd中。

2.描述执行一条JALR 指令的过程·（数据通路、控制信号等）。

> IF阶段：首先NPC将当前指令地址传送给PC，PC将当前PC存储的地址分为两部分，一部分送入PC自增ALU里做+4处理作为下一条指令的地址，另一部分存入Instruction Cache中，进而进入IR，完成流水线的IF阶段。
> ID阶段：对于寄存器堆，Inst[19: 15]寻址到RS1，作为源寄存器。Inst[31: 0]做指令译码生成控制信号，其中过Inst[6: 0]这7位opcode和Inst[14: 12]这3位func 3来产生相应的控制信号。Inst[11: 7]通过Addr寄存器作为Rd。Inst[31: 7]传入立即数扩展，并得到这里需要使用的立即数~1.立即数扩展根据指令译码得到的控制信号，来确认需要扩展的是Inst[31: 20]这12位，得到立即数源操作数。对于译码得到的源寄存器和源操作数，根据控制信号：OP2 src信号选择Imm，将源寄存器和立即数存入段间寄存器，并送入Reg1和Reg2OrImm这两个端口。
> EX阶段：ALU读入两个源操作数，根据ALU func控制信号得出执行加操作，将RS1中的数据和扩展后的imm相加得到ALU out输出.ALU out经由NPC load二选一选择器和result寄存器后传回到ALU src1寄存器，而ALU src2寄存器引入立即数~1，ALU func决定ALU执行逻辑与运算，然后ALU out沿着数据通路Jalr Target直接传回NPC Generator，通过控制信号Jalr写入。IF阶段得到的PC+4做拓展成32位，从PCE传入二选一选择器，根据控制信号Load NPC决定将PCE选出存入Result寄存器。
> M阶段：由于JALR是I-type不需要访问内存，所以M阶段不执行任何操作。
> WB阶段：首先Wb set控制将Rd地址传入寄存器堆，然后将PC+4的值写入寄存器堆中Rd对应的寄存器中。

3.描述执行一条LW 指令的过程（数据通路、控制信号等）

> IF阶段：首先NPC将当前指令地址传送给PC，PC将当前PC存储的地址分为两部分，一部分送入PC自增ALU里做+4处理作为下一条指令的地址，写回NPC。另一部分存入Instruction Cache中，进而进入IR，完成流水线的IF阶段。
> ID阶段：对于寄存器堆，Inst[19: 15]寻址到RS1，作为源寄存器。Inst[31: 0]做指令译码生成控制信号，其中过Inst[6: 0]这7位opcode和Inst[14: 12]这3位func 3来产生相应的控制信号。Inst[11: 7]通过Addr寄存器作为Rd。Inst[31: 7]传入立即数扩展，立即数扩展根据指令译码得到的控制信号，来确认需要扩展的是Inst[31: 20]这12位，得到立即数源操作数，即寻址偏移量。对于译码得到的源寄存器和源操作数，根据控制信号：OP2 src信号选择Imm，将源寄存器和立即数存入段间寄存器，并送入Reg1和Reg2OrImm这两个端口。
> EX阶段：ALU读入两个源操作数，根据ALU func控制信号得出执行加操作，将RS1中的数据和扩展后的imm相加得到ALU out输出，即为LW寻址的相对地址。将ALU out写入二选一选择器，根据Load NPC控制信号，决定将ALU out写入Result寄存器。
> M阶段：访问内存，找到相对地址为ALU out的那32位数据并取出，然后根据Load TypeM不需要做位扩展，所以访存结果直接写入到由WB set控制的二选一选择器中。
> WB阶段：首先WB set控制Rd通过，将Rd即目标寄存器地址写入到寄存器堆中，锁定目标寄存器。然后WB set控制访存结果通过，向寄存器堆中的Rd寄存器中写入访存结果，完成LW。

4.如果要实现CSR 指令（csrrw，csrrs，csrrc，csrrwi，csrrsi，csrrci），设计图中还需要增加什么部件和数据通路？给出详细说明。

> 数据通路部分：首先增加寄存器：CSRs，从 IR 中出来的 Inst[19: 15]、Result 寄存器返回的 ALU out 经过由 CSR src 控制信号决定的二选一选择器传入 CSRs 寄存器。CSRs寄存器传出的数据分为三部分，其中两个分别引入两个二选一选择器，分别与 Reg1 数据和从 Reg2 与 Imm 选择出的数据（这里只能是 Imm ）做二选一操作，然后通入 OP1 与 OP2 寄存器。另一个先存入一个新的段间寄存器，并在 Result 寄存器前新加一个二选一选择器，二选一选择器的输入分别为 CSRs 的数据和前面 ALU out 的二选一选

> 择器输出的值。等待一个时钟周期后，这个二选一选择器的输出值再直接传入 Result 寄存器。Result 寄存器的数据在原来的二个通路基础上增加新的通路，直接传回到 CSRs 寄存器前的二选一选择器，经由控制信号选择传回 CSRs 寄存器。
> 控制信号部分：做控制信号译码时需要额外多做更多的控制信号通路，并增加更多的可以在段间存储控制信号的寄存器。

5.哪些指令分别采用了五类立即数（I-type，S-type，B-type，U-type，J-type 至少各举一例）？Verilog 如何将这些立即数拓展成 32 位的?

> I-type: ADDI 将12位立即数符号扩展成32位，如果最高位为1则从13到32位都是1
> S-type: SB Inst[31: 25]表示偏移量[11: 5]，Inst[11: 7]表示偏移量[4: 0]，先将这12位偏移量做好合并，然后符号扩展到32位，方法同I-type
> B-type: BEQ 首先Inst[31: 25]表示偏移量[12, 10: 5]而Inst[11: 7]表示偏移量[11, 4:1]做好合并,然后将12位偏移量做符号扩展成32位，方法同I-type
> U-type: LUI 将20位立即数算数左移12位，得到32位立即数
> J-type: JAL 首先Inst[31: 12]这20位立即数表示偏移量[20, 10: 1, 11, 19: 12]，先还原成正常的20位偏移量，再做符号扩展城32位，方法同I-type

6.如何实现Data Cache 的非字对齐的Load 和Store?

> 对于非字对齐的 Load，首先指令译码产生控制信号时，产生Load TypeM，控制Data Extension模块做Data Cache中传出的数据的位数扩展，对于LH，将16位的值直接符号扩展成32位。对于LHU，做0扩展到32位，LB指令是将8位数据做符号扩展成32位，而LBU是将8位数据做0扩展到32位。根据四种处理非字对齐Load的方式，控制信号Load TypeM需要两位。
> 对于非字对齐的 Store，同样首先根据指令译码产生相应的控制信号，这里命名为Store TypeM，处理方式为：对于ALU out出来的待存储的地址数据，通过Reg2寄存器传入，表示存储在Data Cache里的地址，为x[rs1]+sign-extend(offset)。这里SB由于是存储8位数据，所以直接将x[rs1][7: 0]存入对应的Data Cache单元中。而对于SH是存储16位数据，所以直接将x[rs1][15: 0]存入对应的Data Cache单元中。

7.ALU 模块中，默认wire 变量是有符号数还是无符号数?

> 有符号数

8.哪条指令执行过程中会使得Load Npc == 1?

> 设计中认为对于这个二选一选择器来说，Load Npc = 1表示选择的是PCE，即对PC做了拓展后的结果。所以可以产生控制信号Load Npc = 1的指令有：AUIPC JAL JALR 这些最后WB阶段会把PC+4做拓展后存回寄存器文件的指令

9.NPC Generator 中对于不同跳转target 的选择有没有优先级?

> 优先级BR>JALR>JAL

10.Harzard 模块中，有哪几类冲突需要插入气泡?

> 数据相关冲突： RAW WAR WAW
> 资源相关冲突： 如前后两条指令都需要使用ALU时后面一条应该加入气泡等待

11.Harzard 模块中采用默认不跳转的策略，遇到branch 指令时，如何控制flush 和stall 信号?

> 由于默认不跳转，所以遇到branch时，不需要做flush处理，所有的flush信号都为0，而branch指令中例如BR等不需要5个周期就可以执行完，但为了解决可能的冲突需要加入stallF、stallE、stallW相应为1的信号来保证流水线可以正常没有冲突地运行

12. 0 号寄存器值始终为0，是否会对forward 的处理产生影响?

> 不会，0号寄存器只是为了当经常会遇到大量的跟0比较/赋值成0的场景时，设置一个寄存器，永远都是0，这样就不需要每次给一个寄存器赋值成0再跟另外一个寄存器比较了，提高效率，而且0号寄存器不允许写入，属于只读寄存器，所以不影响forward处理。