P5思考题解答

流水线冲突解决了前流水级需要的数值是还没有来得及写入的数值。本设计考虑匹配寄存器的方式解决转发，只要满足以下条件：

1. 已经产生了需要写入的数据
2. 转发的元寄存器非0
3. 寄存器写使能为1
4. 两两流水级之间使用和准备写入寄存器号匹配

即可转发。

由于流水线IF级不需要对寄存器的操作，故这里我们可能进行的转发路径共有六条；再仔细分析，1. 对于ID 🡪 EX，在ID级需要使用寄存器值的只有Branch和Jump指令，由于延迟槽的存在，两条B, J指令并不能连续出现，而在EX级可能产生新值的指令只有Link型指令，故不需要构建EX 🡪 ID的转发；2. 对于WB 🡪 MEM，可以通过GRF片内转发实现。

本设计中将W->M级写入数据汇总，方便转发

处理转发的代码如下：

reg [1:0] \_ForwardRSD, \_ForwardRTD, \_ForwardRSE, \_ForwardRTE, \_ForwardRTM, \_ForwardRTE\_ALUb;

//reg [1:0] \_ForwardRaF;

assign ForwardRSD = \_ForwardRSD;

assign ForwardRTD = \_ForwardRTD;

assign ForwardRSE = \_ForwardRSE;

assign ForwardRTE = \_ForwardRTE;

assign ForwardRTM = \_ForwardRTM;

assign ForwardRTE\_ALUb = \_ForwardRTE\_ALUb;

always @\* begin

if (ReReg1\_D == WrReg\_M && RegWr\_M && generated\_M && ReReg1\_D)

\_ForwardRSD = `WrData\_M;

else if (ReReg1\_D == WrReg\_W && RegWr\_W && generated\_W && ReReg1\_D)

\_ForwardRSD = `WrData\_W;

else

\_ForwardRSD = 2'b00;

if (ReReg2\_D == WrReg\_M && RegWr\_M && generated\_M && ReReg2\_D)

\_ForwardRTD = `WrData\_M;

else if (ReReg2\_D == WrReg\_W && RegWr\_W && generated\_W && ReReg2\_D)

\_ForwardRTD = `WrData\_W;

else

\_ForwardRTD = 2'b00;

end

always @\* begin

if (ReReg1\_E == WrReg\_M && RegWr\_M && generated\_M && ReReg1\_E)

\_ForwardRSE = `WrData\_M;

else if (ReReg1\_E == WrReg\_W && RegWr\_W && generated\_W && ReReg1\_E)

\_ForwardRSE = `WrData\_W;

else

\_ForwardRSE = 2'b00;

if (ReReg2\_E == WrReg\_M && RegWr\_M && generated\_M && ReReg2\_E)

\_ForwardRTE = `WrData\_M;

else if (ReReg2\_E == WrReg\_W && RegWr\_W && generated\_W && ReReg2\_E)

\_ForwardRTE = `WrData\_W;

else

\_ForwardRTE = 2'b00;

end

always @\* begin

if (ReReg2\_M == WrReg\_W && RegWr\_W && generated\_W && ReReg2\_M)

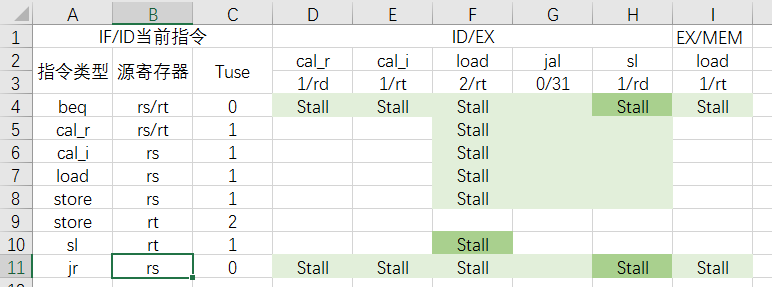
\_ForwardRTM = `WrData\_W;

else

\_ForwardRTM = 2'b00;

end

当以上的转发路径中，若在需求寄存器的值的时候新值还没有产生，便无法直接用转发解决冲突，需要通过暂停处理。通过分析Tuse,Tnew在冲突控制单元中在指令进入ID级的时候判断是否暂停。



处理暂停的代码如下：

wire stall\_b, stall\_calr, stall\_cali, stall\_ld, stall\_st, stall\_jr, stall\_sl, temp;

assign temp = (stall\_b || stall\_calr || stall\_cali || stall\_ld || stall\_st || stall\_jr || stall\_sl);

assign stall = (temp === 1'bx)? 0:temp;

assign stall\_b = beq\_D && ((Cal\_r\_E && (IR\_D[`rs] == IR\_E[`rd]||IR\_D[`rt] == IR\_E[`rd]))

||(Cal\_i\_E && (IR\_D[`rs] == IR\_E[`rt]||IR\_D[`rt] == IR\_E[`rt]))

||(ld\_E && (IR\_D[`rs] == IR\_E[`rt]||IR\_D[`rt] == IR\_E[`rt]))

||(sl\_E && (IR\_D[`rs] == IR\_E[`rd]||IR\_D[`rt] == IR\_E[`rd]))

||(ld\_M && (IR\_D[`rs] == IR\_M[`rt]||IR\_D[`rt] == IR\_M[`rt])));

assign stall\_calr = Cal\_r\_D && ld\_E && (IR\_D[`rs] == IR\_E[`rt]||IR\_D[`rt] == IR\_E[`rt]);

assign stall\_cali = Cal\_i\_D && ld\_E && (IR\_D[`rs] == IR\_E[`rt]||IR\_D[`rt] == IR\_E[`rt]);

assign stall\_ld = ld\_D && ld\_E && IR\_D[`rs] == IR\_E[`rt];

assign stall\_st = st\_D && ld\_E && IR\_D[`rs] == IR\_E[`rt];

assign stall\_sl = sl\_D && ld\_E && IR\_D[`rt] == IR\_E[`rt];

assign stall\_jr = jr\_D && ((Cal\_r\_E && IR\_D[`rs] == IR\_E[`rd])

||(Cal\_i\_E && IR\_D[`rs] == IR\_E[`rt])

||(sl\_E && IR\_D[`rs] == IR\_E[`rd])

||(ld\_E && IR\_D[`rs] == IR\_E[`rt])

||(ld\_M && IR\_D[`rs] == IR\_M[`rt]));