浙江大学

本科实验报告

课程名称：计算机体系结构

姓　　名：郑乔尹

学　　院：计算机科学与技术学院

系：计算机科学与技术系

专　　业：计算机科学与技术

学　　号：3210104169

指导教师：姜晓红

2023　年　10　月　10　日

浙江大学实验报告

课程名称：计算机体系结构 实验类型：综合

实验项目名称：Lab1：Pipelined CPU supporting RISC-V RV32I Instructions

学生姓名：郑乔尹 专业：计算机科学与技术 学号：3210104169

同组学生姓名： 指导老师：姜晓红

实验地点：曹西 301 实验日期：2023 年 10 月 10 日

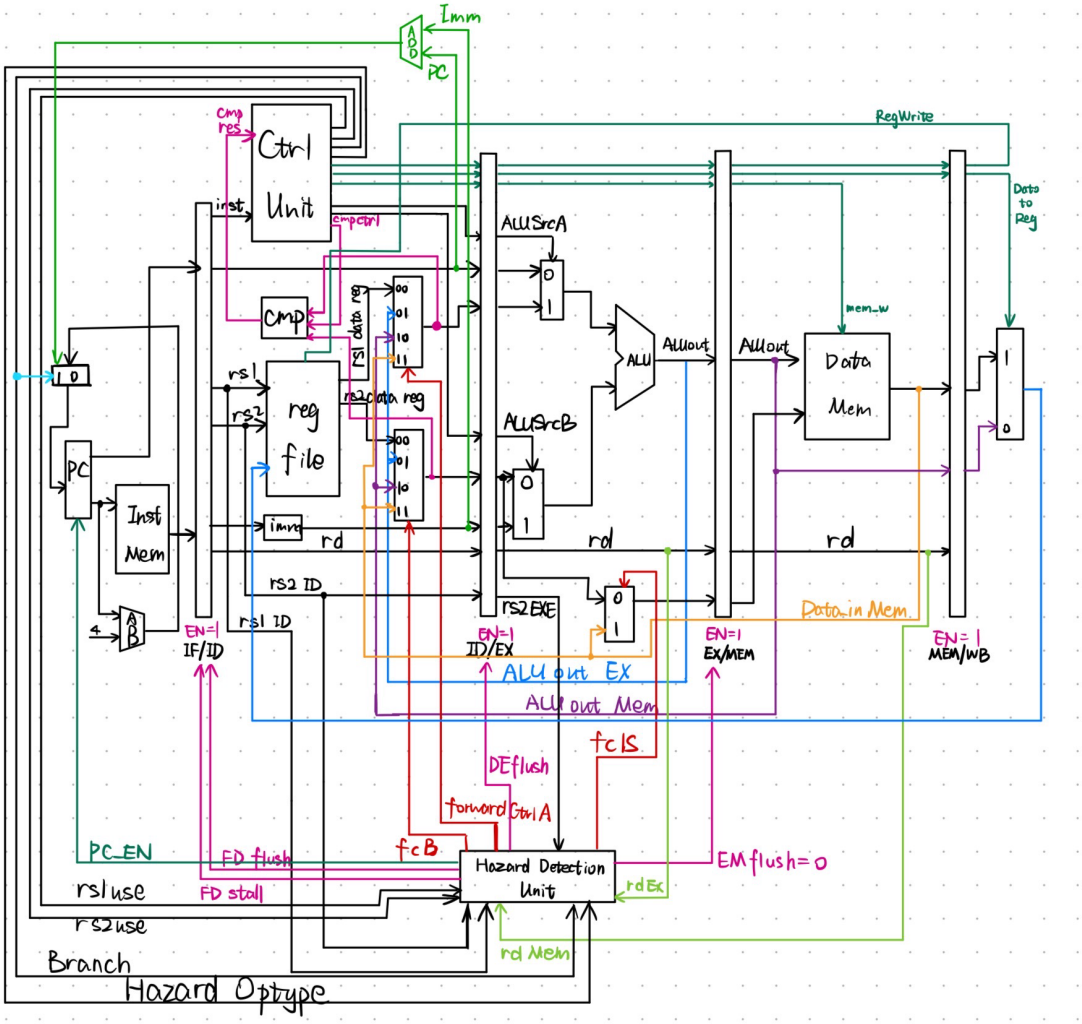
一、 实验目的和要求

Task 1: 写出本次实验的目的与要求 (5 points)

1. 理解 RISC-V RV32I 指令
2. 掌握流水线 CPU 的设计方法
3. 掌握带 forward 的流水线 CPU 的设计方法
4. 掌握 Predict-not-taken 的 branch 指令设计
5. 掌握流水线 CPU 的程序验证方法

二、 实验内容和原理

Task 2: 画出本次实验实现的电路图。（可以使用 PPT 上的线路图进行修改，但是必须和自己的实现保持一致。如果不一致，本题将不给分） (10 points)



Task 3：请给出为实现 predict not taken 各级流水线 EN/stall/flush 的实现思路及代码，给出各个信号为真的条件 (15 points)

各级流水线 EN 信号均可置 1.

Reg_FD_flush 与跳转指令相关，理由见下：

Predict-not-taken：假设不跳转，先继续为下一条指令取指，等待当前指令到达译码阶段后可以得知比较结果是否正确，假如错误，则继续执行(reg_FD_flush 为 0)，否则将已经取指的下一条指令进行 flush(此时将 reg_FD_flush 置 1 即可)，防止其被译码并错误执行，然后执行跳转指令，跳转至正确位置

```
    assign reg_FD_flush = Branch_ID; // branch correct, flush the content
in IF/ID
```

reg_FD_stall 与 L type 造成的 stall 有关，如果需要发生 stall，则将其置 1 以停止对当前已经取指的指令进行译码，以实现 stall：

```
    assign reg_FD_stall = stall; // stall
```

同时停止取指，PC 使能置 0：

```
assign PC_EN_IF = ~stall; // stall, no IF
```

同时需要将已经译码的当前指令的执行流水线中止，即将 reg_DE_flush 置 1，stall 后再让其继续执行：

```
assign reg_DE_flush = stall; // stall, flush the content in ID/EX
```

Task 4: 请给出 3 个 forward 的实现思路及代码。对于 forward_ctrl_A 和 forward_ctrl_B 只需要给出其中一个即可。forward_ctrl_A 有几种信号？每种信号对应什么情况？forward_ctrl_ls 的条件是什么？ (20 points)

Forward_ctrl_A 有 4 种信号，00，01，10，11；00 对应不用 forward，直接使用寄存器值，01 对应从上一指令 EX 阶段 forward ALU 计算结果到当前指令，10 对应从上上条指令 MEM 阶段 forward ALU 计算结果到当前指令，11 对应从上上条指令 MEM 阶段 forward 内存读取值到当前指令，用于 Load 指令与非 Store 指令的 forward。

```
wire rs1_forward_ED = (hazard_optype_EX == 2'b01) && //EXE hazard data
       ((rd_EXE == rs1_ID) && rd_EXE) && //EXE write to rs1
       (rs1use_ID); //ID read from rs1
 wire rs1_forward_MD = (hazard_optype_MEM == 2'b01) && //MEM hazard data
       ((rd_MEM == rs1_ID) && rd_MEM) && //MEM write to rs1
       (rs1use_ID); //ID read from rs1
wire rs1_forward_LD = (hazard_optype_MEM == 2'b10) && //MEM hazard load
       ((rd_MEM == rs1_ID) && rd_MEM) && //MEM write to rs1
       (rs1use_ID); //ID read from rs1
```

注意 forward_ctrl_A/B 信号中的优先级，应当优先接受 EX 阶段的 forward，因为这是最新的值，一开始的时候直接用了按位与，导致优先级出错，像 RRR 这种情况就没办法正确 forward，后来采用三元运算符，解决了优先级问题，代码如下：

```
assign forward_ctrl_A = rs1_forward_ED ? 2'b01 :
                        (rs1_forward_MD ? 2'b10 :
                        (rs1_forward_LD ? 2'b11 : 2'b00));
```

Forward_ctrl_ls 有 2 种信号：0 代表不发生 forward，1 代表将上一指令 MEM 阶段的内存读取值 forward 到当前指令的 MEM 阶段，解决 Load 与 Store 指令相连时发生的 data hazard，信号由以下方式得出，store 指令执行到 Ex 阶段时，判断其 rs2 是否需要使用 load 指令中的 rd，同时判断两指令是否符合上一条为 L 下一条为 S：

```
    assign forward_ctrl_ls = (rs2_EXE == rd_MEM) && rd_MEM &&
(hazard_optype_EX == 2'b11) && (hazard_optype_MEM == 2'b10);
```

Task 5: 请简要解释顶层 RV32core 的连线 (5 points)

根据自己定义的 forward 信号意义进行连线：forward_ctrl_A: 00 使用 rs1_data_reg, 01 使用 ALUout_EXE 获取 EX 阶段 ALU 计算值进行 forward, 10 使用 ALUout_MEM，获取 MEM 阶段 ALU 计算值进行 forward，11 使用 Datain_MEM，获取 MEM 阶段内存读取值进行 forward。

```
MUX4T1_32
mux_forward_A(.I0(rs1_data_reg),.I1(ALUout_EXE),.I2(ALUout_MEM),.I3(Dat
ain_MEM),        //to fill sth. in ()
        .s(forward_ctrl_A),.o(rs1_data_ID));
```

Forward_ctrl_B 同理：

```
 MUX4T1_32
mux_forward_B(.I0(rs2_data_reg),.I1(ALUout_EXE),.I2(ALUout_MEM),.I3(Dat
ain_MEM),        //to fill sth. in ()
        .s(forward_ctrl_B),.o(rs2_data_ID));
```

Forward_ctrl_ls:

1 代表从 Mem 阶段前递内存读取值，0 代表正常使用 rs2_data

```
    MUX2T1_32
mux_forward_EXE(.I0(rs2_data_EXE),.I1(Datain_MEM),.s(forward_ctrl_ls),.
o(Dataout_EXE));        //to fill sth. in ()
```

ALUSrcA:

1 代表 JAL，JALR，AUIPC 这些需要 PC 值的指令，获取 PC 值，0 代表使用 rs1_data。

```
    MUX2T1_32
mux_A_EXE(.I0(rs1_data_EXE),.I1(PC_EXE),.s(ALUSrc_A_EXE),.o(ALUA_EXE));
    //to fill sth. in ()
```

ALUSrcB:

1 代表 L, I, LUI, AUIPC, S 这些需要立即数的指令获取立即数作为输入 B，0 则代表使用 rs2_data.

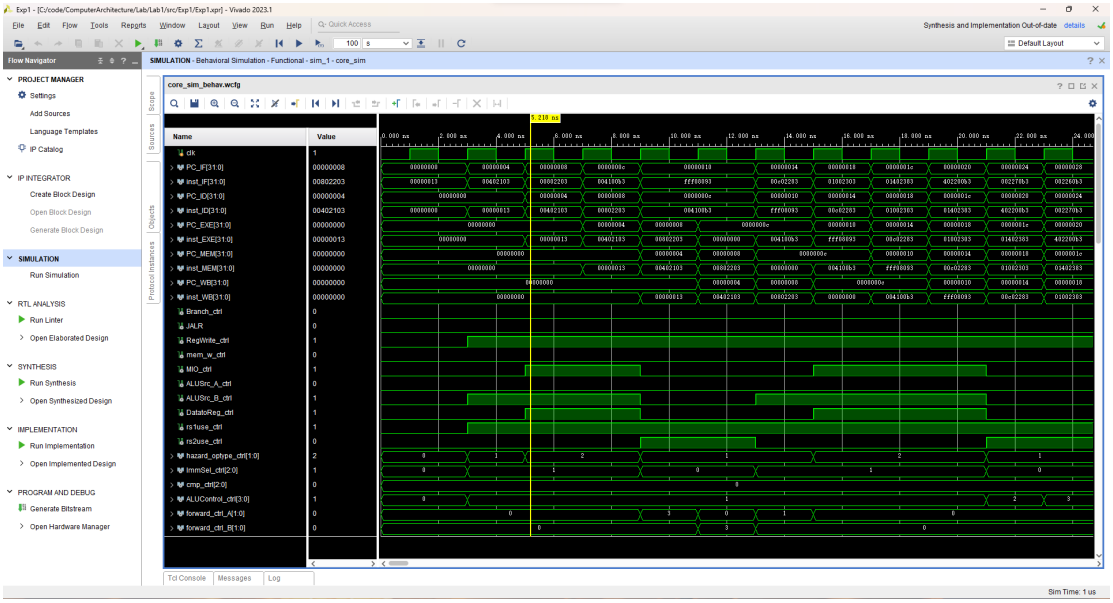
```
    MUX2T1_32
mux_B_EXE(.I0(rs2_data_EXE),.I1(Imm_EXE),.s(ALUSrc_B_EXE),.o(ALUB_EXE))
;       //to fill sth. in ()
```

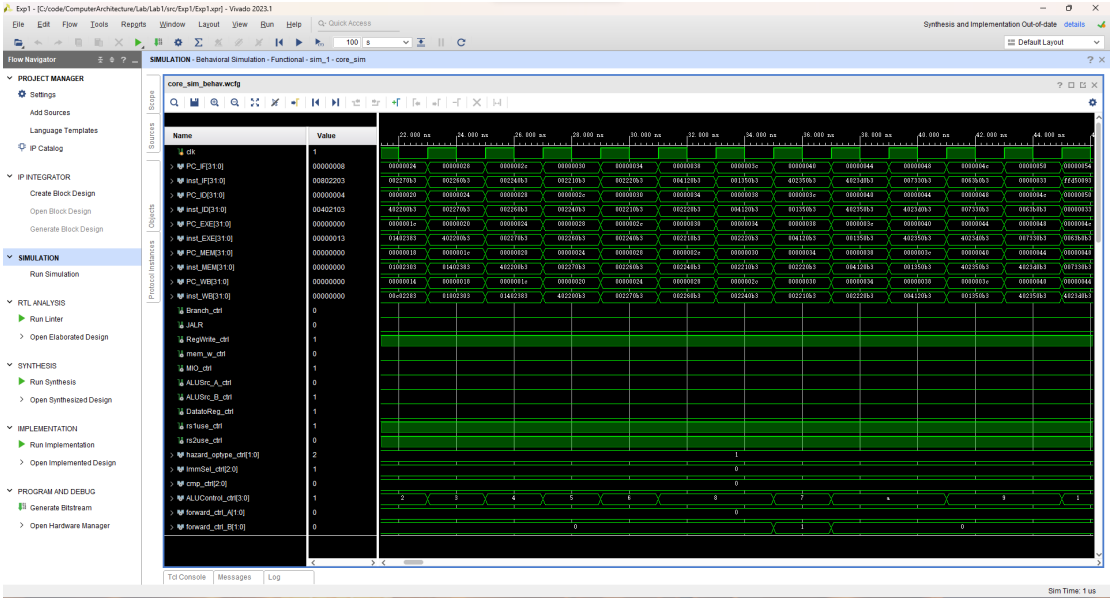
三、 实验过程和数据记录及结果分析

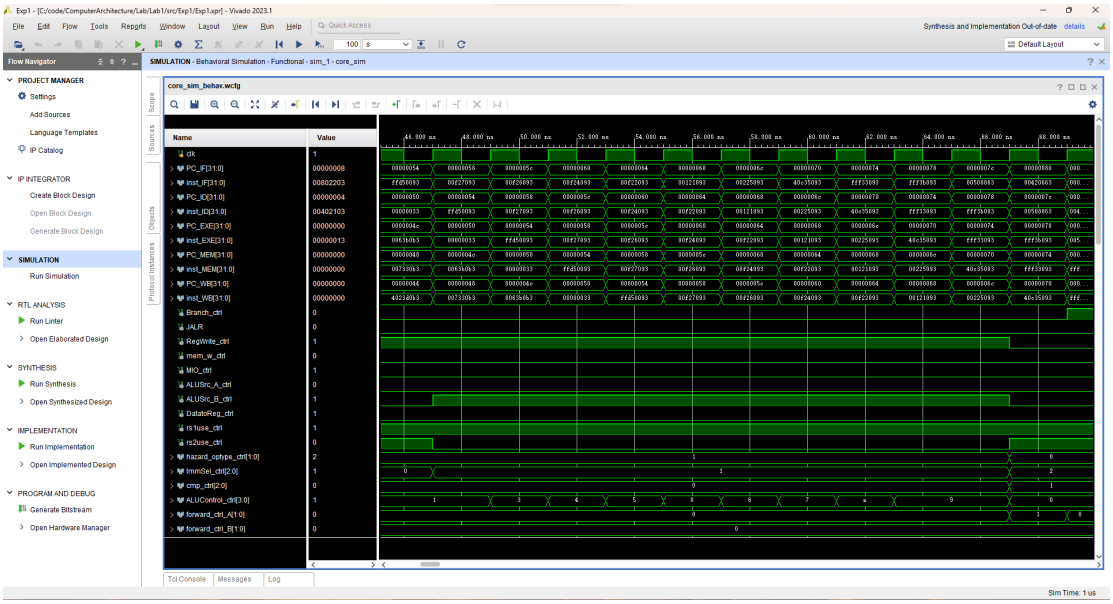
仿真图片应完整包含时间信息和信号名称。

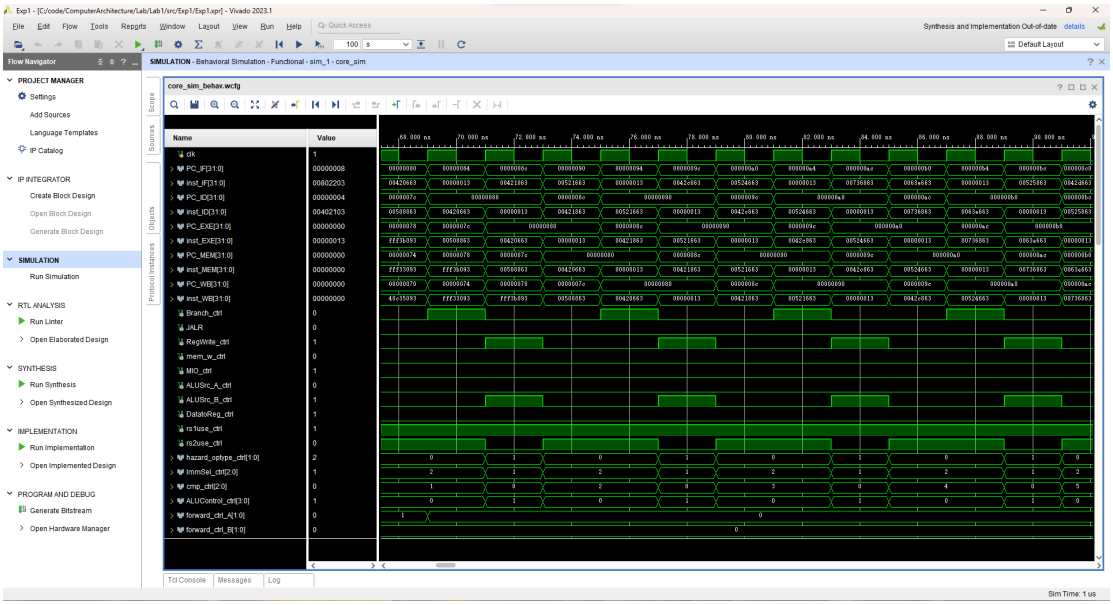
对仿真的解释示例：XXXns，X 信号变为 X，由于 XXX，导致 X 信号变为 XXX，……，我们发现 X 被 forward 到了 X。

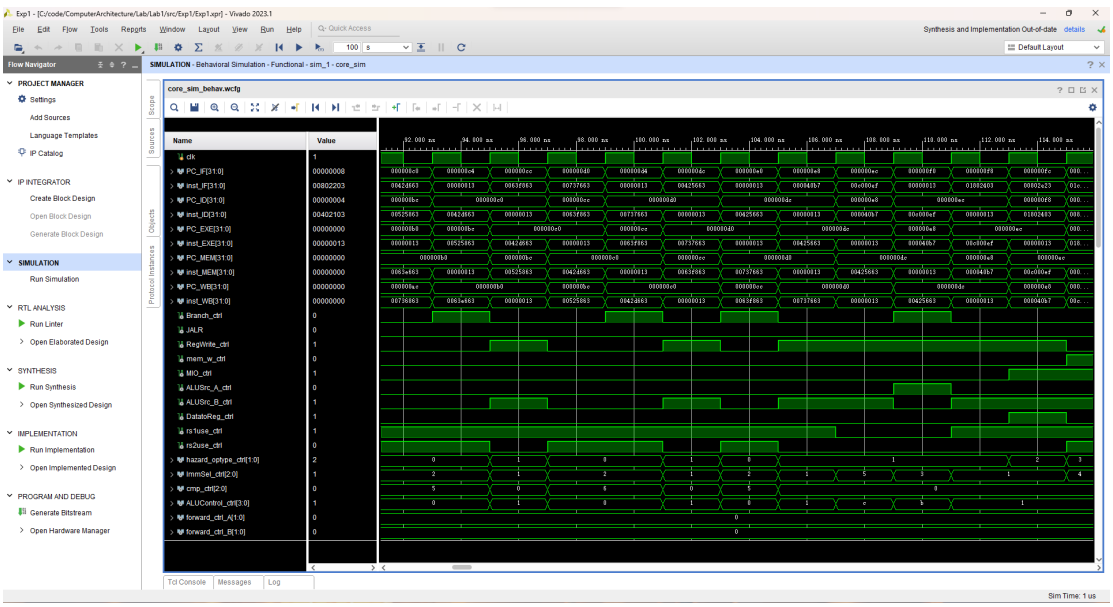
Task 5: 请给出本次实验仿真的完整截图 (5 points)

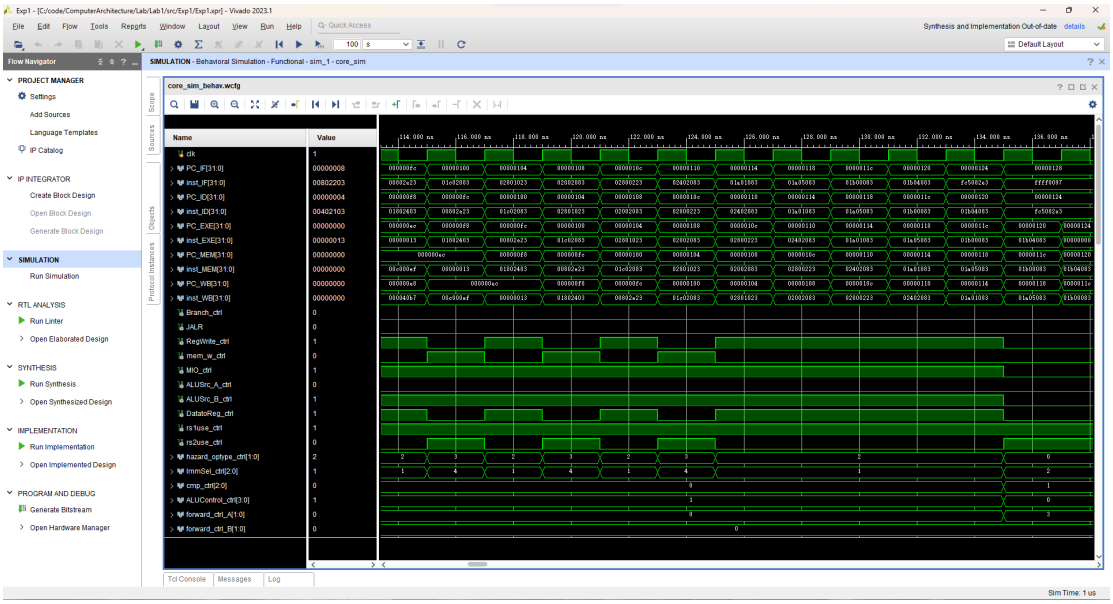


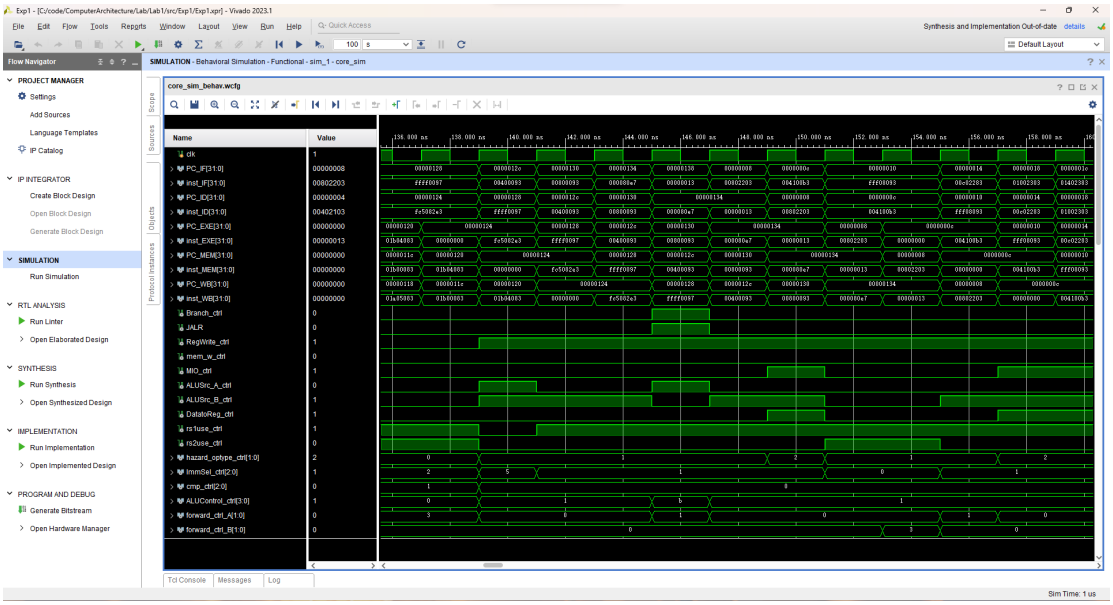



Exp1 - [C:/code/ComputerArchitecture/Lab/Lab1/src/Exp1/Exp1.xpr] - Vivado 2023.1
File Edit Flow Tools Reports Window Layout View Run Help
Quick Access
Synthesis and Implementation Out-of-date details
Default Layout
Flow Navigator
SIMULATION - Behavioral Simulation - Functional - sim_1 - core_sim
core_sim_behav.wcfg
Name
Value
clk
PC_IF[31:0]
inst_IF[31:0]
PC_ID[31:0]
inst_ID[31:0]
PC_EXE[31:0]
inst_EXE[31:0]
PC_MEM[31:0]
inst_MEM[31:0]
PC_WB[31:0]
inst_WB[31:0]
Branch_ctrl
JALR
RegWrite_ctrl
mem_w_ctrl
MIO_ctrl
ALUSrc_A_ctrl
ALUSrc_B_ctrl
DatatoReg_ctrl
rs1use_ctrl
rs2use_ctrl
hazard_optype_ctrl[1:0]
ImmSel_ctrl[2:0]
cmp_ctrl[2:0]
ALUControl_ctrl[3:0]
forward_ctrl_A[1:0]
forward_ctrl_B[1:0]
PROJECT MANAGER
Settings
Add Sources
Language Templates
IP Catalog
IP INTEGRATOR
Create Block Design
Open Block Design
Generate Block Design
SIMULATION
Run Simulation
RTL ANALYSIS
Run Linter
Open Elaborated Design
SYNTHESIS
Run Synthesis
Open Synthesized Design
IMPLEMENTATION
Run Implementation
Open Implemented Design
PROGRAM AND DEBUG
Generate Bitstream
Open Hardware Manager
Tcl Console
Messages
Log
Sim Time: 1 us

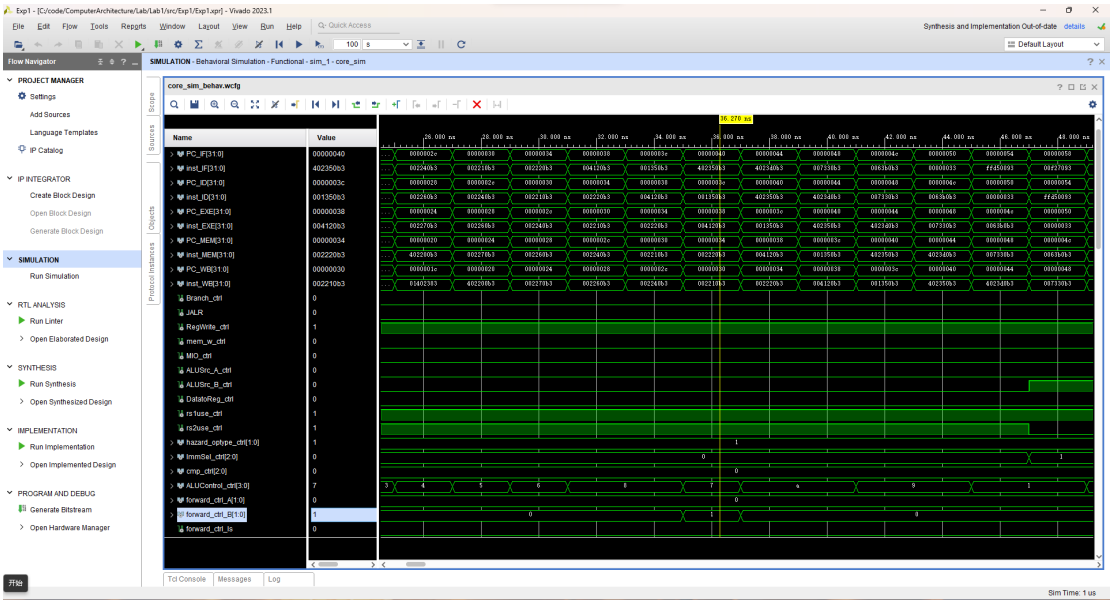

Exp1 - [C:/code/ComputerArchitecture/Lab/Lab1/src/Exp1/Exp1.xpr] - Vivado 2023.1
File Edit Flow Tools Reports Window Layout View Run Help
Quick Access
Synthesis and Implementation Out-of-date details
Default Layout
Flow Navigator
SIMULATION - Behavioral Simulation - Functional - sim_1 - core_sim
core_sim_behav.wcfg
Name
Value
clk
PC_IF[31:0]
inst_IF[31:0]
PC_ID[31:0]
inst_ID[31:0]
PC_EXE[31:0]
inst_EXE[31:0]
PC_MEM[31:0]
inst_MEM[31:0]
PC_WB[31:0]
inst_WB[31:0]
Branch_ctrl
JALR
RegWrite_ctrl
mem_w_ctrl
MIO_ctrl
ALUSrc_A_ctrl
ALUSrc_B_ctrl
DatatoReg_ctrl
rs1use_ctrl
rs2use_ctrl
hazard_optype_ctrl[1:0]
ImmSel_ctrl[2:0]
cmp_ctrl[2:0]
ALUControl_ctrl[3:0]
forward_ctrl_A[1:0]
forward_ctrl_B[1:0]
PROJECT MANAGER
Settings
Add Sources
Language Templates
IP Catalog
IP INTEGRATOR
Create Block Design
Open Block Design
Generate Block Design
SIMULATION
Run Simulation
RTL ANALYSIS
Run Linter
Open Elaborated Design
SYNTHESIS
Run Synthesis
Open Synthesized Design
IMPLEMENTATION
Run Implementation
Open Implemented Design
PROGRAM AND DEBUG
Generate Bitstream
Open Hardware Manager
Tcl Console
Messages
Log
Sim Time: 1 us


Exp1 - [C:/code/ComputerArchitecture/Lab/Lab1/src/Exp1/Exp1.xpr] - Vivado 2023.1
File Edit Flow Tools Reports Window Layout View Run Help
Quick Access
Synthesis and Implementation Out-of-date details
Default Layout
Flow Navigator
SIMULATION - Behavioral Simulation - Functional - sim_1 - core_sim
core_sim_behav.wcfg
Name
Value
clk
PC_IF[31:0]
inst_IF[31:0]
PC_ID[31:0]
inst_ID[31:0]
PC_EXE[31:0]
inst_EXE[31:0]
PC_MEM[31:0]
inst_MEM[31:0]
PC_WB[31:0]
inst_WB[31:0]
Branch_ctrl
JALR
RegWrite_ctrl
mem_w_ctrl
MIO_ctrl
ALUSrc_A_ctrl
ALUSrc_B_ctrl
DatatoReg_ctrl
rs1use_ctrl
rs2use_ctrl
hazard_optype_ctrl[1:0]
ImmSel_ctrl[2:0]
cmp_ctrl[2:0]
ALUControl_ctrl[3:0]
forward_ctrl_A[1:0]
forward_ctrl_B[1:0]
PROJECT MANAGER
Settings
Add Sources
Language Templates
IP Catalog
IP INTEGRATOR
Create Block Design
Open Block Design
Generate Block Design
SIMULATION
Run Simulation
RTL ANALYSIS
Run Linter
Open Elaborated Design
SYNTHESIS
Run Synthesis
Open Synthesized Design
IMPLEMENTATION
Run Implementation
Open Implemented Design
PROGRAM AND DEBUG
Generate Bitstream
Open Hardware Manager
Tcl Console
Messages
Log
Sim Time: 1 us



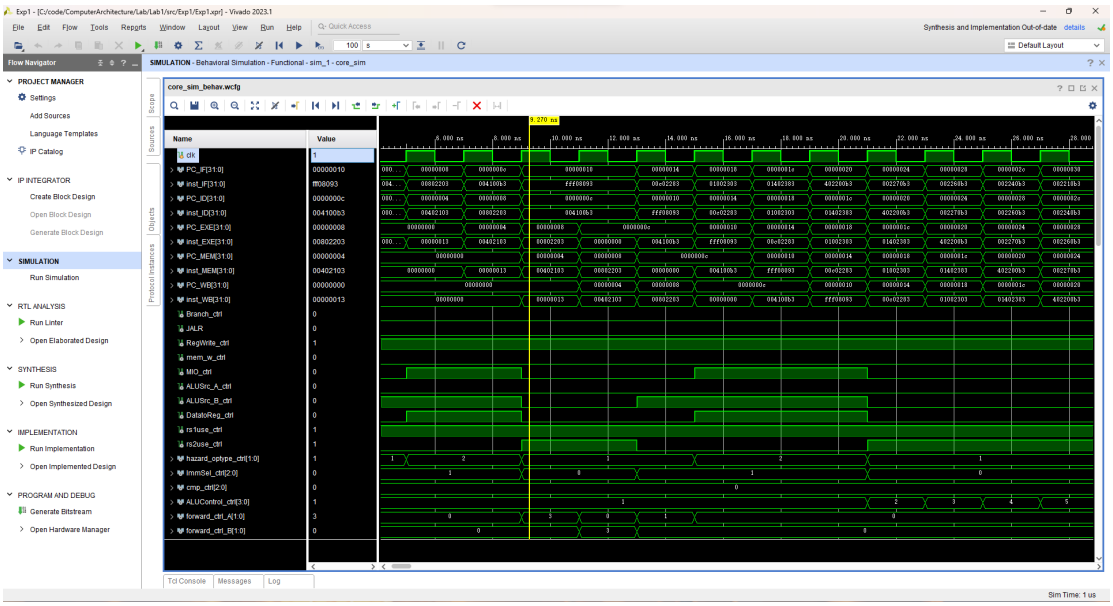


Task 6: 请给出一个 RR forward 部分仿真的高清图片，并对涉及到的信号加以详细解释 (10 points)



对应指令 `slt x1,x4,x2　slt x1,x2,x4　srl x1, x6, x1;` `srl x1, x6, x1` 对 `slt x1,x2,x4` 中的 x1 有着依赖关系，可以看到在 `PC=0x40` 即 `srl x1,x6,x1` 进入译码阶段时，由于发生冲突的 x1 为 rs2，需要上一条指令 EX 阶段的 ALU 计算值，故 hazard_optype 被置 01，代表需要 ALU 计算值前递，在 HazardDetectionUnit 中将 `forward_ctrl_B` 变为 01，代表 rs2 使用上一条指令(EX 阶段)的 ALU 计算结果前递。

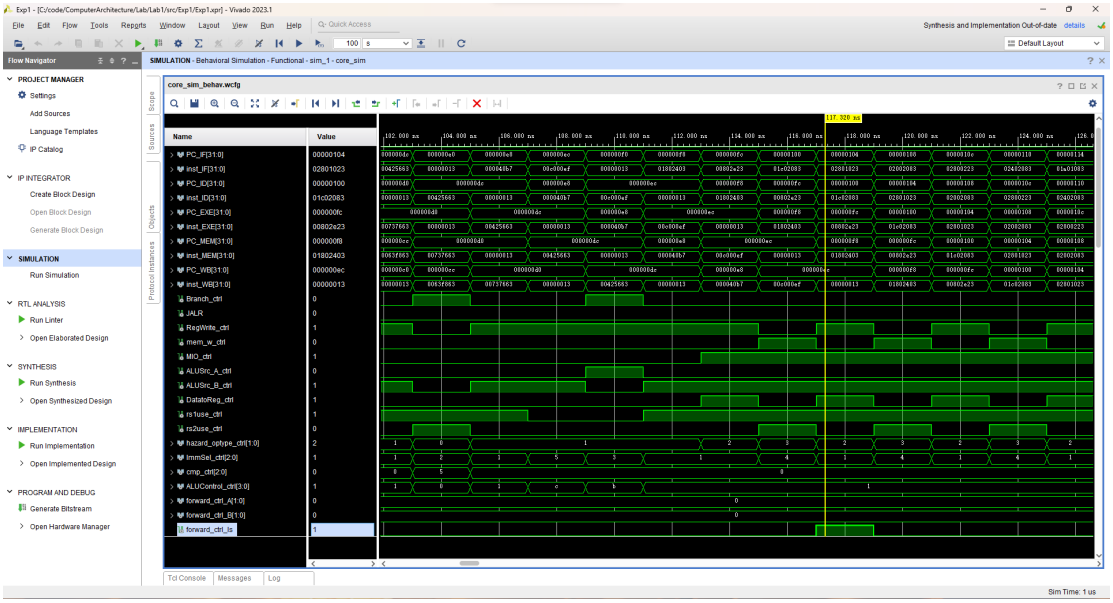
Task 7: 请给出一个 LR forward 部分仿真的高清图片，并对涉及到的信号加以详细解释 (10 points)



可以看到图中 PC=0x10 时发生了 stall。需要 LR forward 的指令为 lw x4, 8(x0) 与 add x1, x2, x4，add 指令中使用的 rs2 为 x4，而上一条 lw 指令还未将内存中读取的值写入 x4,这个值需要在 Mem 阶段获取，进而前递，因此需要进行一个周期的 stall。

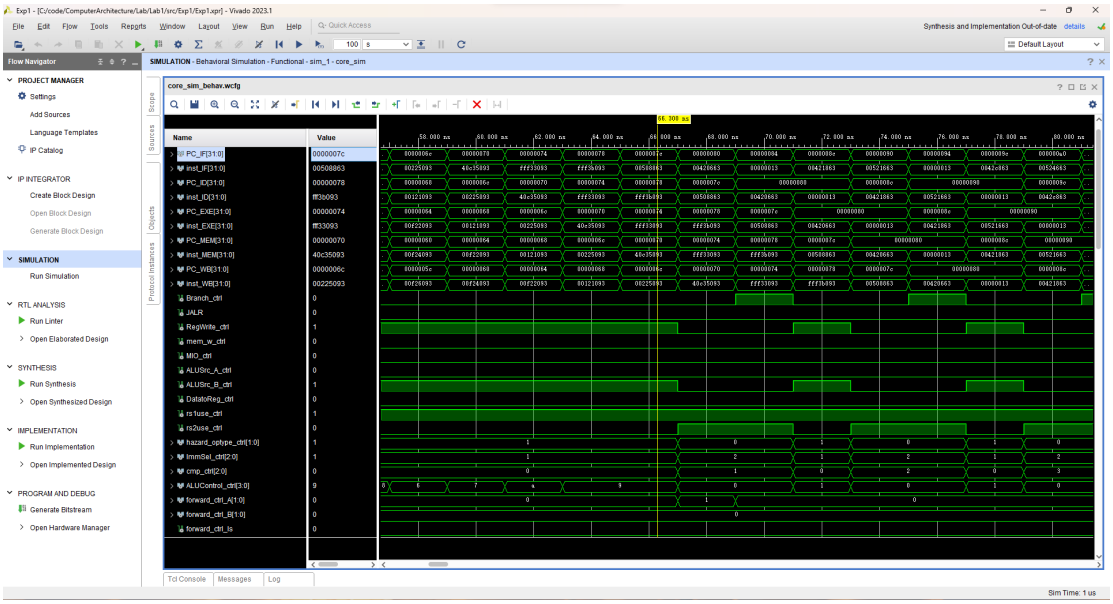
可以看到在 PC=0x0c 时，也即 lw 指令译码时，hazard_optype 被置为 10，代表当前指令为 L type；PC=0x10 时，也就是 add 指令译码阶段，hazard_optype 被置为 01，代表当前为计算指令，HazardDetectionUnit 通过上一指令的 hazard_optype 与当前指令的 hazard_optype 判断出现在需要 LR_forward，从而将 PC_en_IF 置 0，reg_FD_stall 置 1，停止取指，以对流水线进行 stall，同时要将 reg_DE_flush 置 1，防止 add 指令被继续执行。Stall 一个周期之后，lw 指令进入 Mem 阶段，这时符合 rs2_forward_LD 的条件，forward_ctrl_B 被置 11，代表从 Mem 阶段前递内存读取值到 rs2。

Task 8: 请给出一个 LS forward 部分仿真的高清图片，并对涉及到的信号加以详细解释 (10 points)



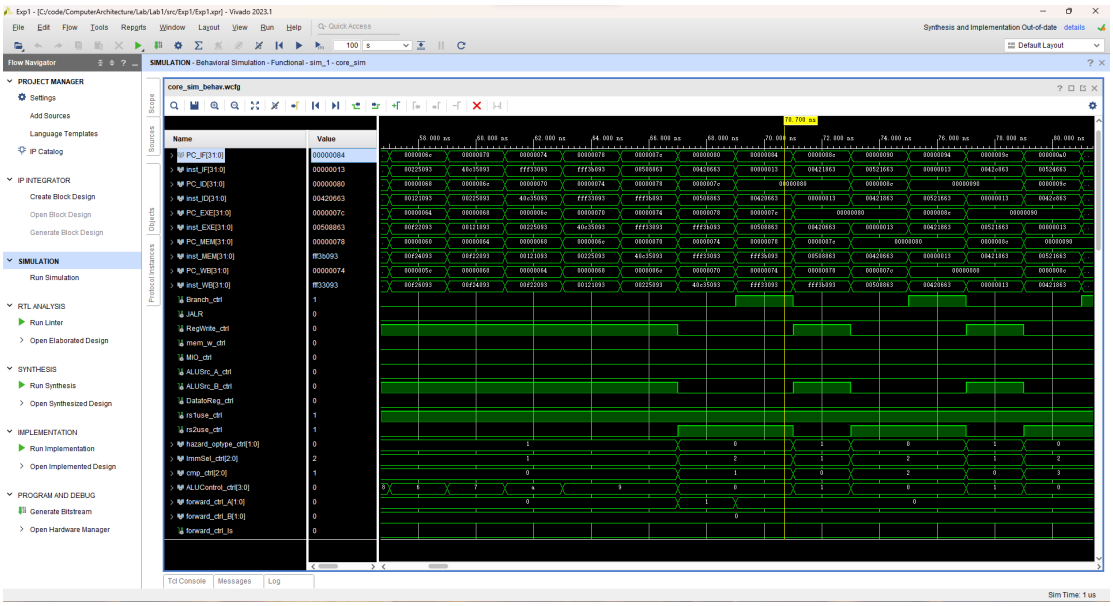
这是 lw x8, 24(x0) 与 sw x8, 28(x0)两条指令发生的 LS forward，PC=0xfc 时，lw 指令被译码，获取到 hazard_optype=10，代表 L Type 指令，PC=0x100 时，sw 指令被译码，获取到 hazard_optype=11，代表 S Type 指令，等到 lw 指令执行到 Mem 阶段后，发现此时在 Ex 阶段的 sw 需要的 rs2 与 lw 指令中的 rd 出现冲突，于是 HazardDetectionUnit 检测到需要 LS forward，将 forward_ctrl_ls 置 1，代表将 Mem 阶段的内存读取值前递到下一指令，供 sw 指令在即将到来的 Mem 阶段使用。

Task 9: 请给出一个 predict 成功部分仿真的高清图片，并对涉及到的信号加以详细解释 (5 points)



该指令为(PC=0x7c 取指)beq x1,x5,label0，实际上它还与上一条指令有着 data hazard，但是在其译码阶段(PC=0x80)已经正确获取了前递值。通过获取到的 x1，x5 数据以及 cmp_ctrl(=0，代表当前指令需要判断相等)，cmp_32 模块可以判断分支结果是否正确，这里 cmp_res=0，代表比较结果不符，传入 HazardDetectionUnit 的 Branch_ID 为 0，代表分支条件错误，由于执行的为 predict-not-taken 策略，继续执行即可，不用对其他信号进行更改。

Task 10: 请给出一个 predict 失败部分仿真的高清图片，并对涉及到的信号加以详细解释 (5 points)



该指令为(PC=0x80 取指) beq x4,x4,label0。通过获取到的 x4，x4 数据以及 cmp_ctrl(=0，代表当前指令需要判断相等)，cmp_32 模块可以判断分支结果是否正确，这里 cmp_res=1，代表比较结果符合，传入 HazardDetectionUnit 的 Branch_ID 为 1，代表分支条

件正确，由于执行的为 predict-not-taken 策略，下一条指令已经进入取指阶段，需要停止其执行并对 PC 进行跳转，故将 reg_FD_flush 置 1，防止已经取指的下一条继续执行。然后正常进行分支指令跳转即可。

四、 讨论与心得

Task 11: 请写出对本次实验内容的深入讨论，或者本次实验的心得体会。例如遇到的难题等等。请认真填写本模块，若不填写或胡乱填写将酌情扣分，写明白真实情况即可。 (+10 points)

在提交 PTA 后发现 cycle21 处出现错误，检查发现是 RRR 情况的 Hazard 出现错误，通过对代码进行检查，发现原来 forward_ctrl_A 使用的都是位运算，存在严重的优先级问题，比如当同时检测到 EX 与 MEM 阶段需要 forward 时，forward_ctrl_A/B 将会出错，不会输出 01，而是 11，导致结果错误，最后改为三元运算符，成功通过了仿真。同时提交时忘记给输出信号 reg_EM_flush 赋值，导致高阻态，吸取了教训，以后实验会将所有输出信号赋值，即使自己没有使用。