哈爾濱工業大學（深圳）
HARBIN INSTITUTE OF TECHNOLOGY

实验报告

开课学期：2022 夏季

课程名称：计算机设计与实践

实验名称：CPU 设计

实验性质：综合设计型

实验学时：52 地点：T2###

学生班级：计算机 7 班

学生学号：200110711

学生姓名：黄飞宇

评阅教师：

报告成绩：

实验与创新实践教育中心制

2022 年 7 月

设计的功能描述（含所有实现的指令描述，以及单周期/流水线 CPU 频率）

支持 24 条必做指令的单周期/流水线 miniRV-1 CPU，单周期 CPU 频率为 25MHz，流水线 CPU 频率为 100MHz，支持下板验证。

各指令描述如下：

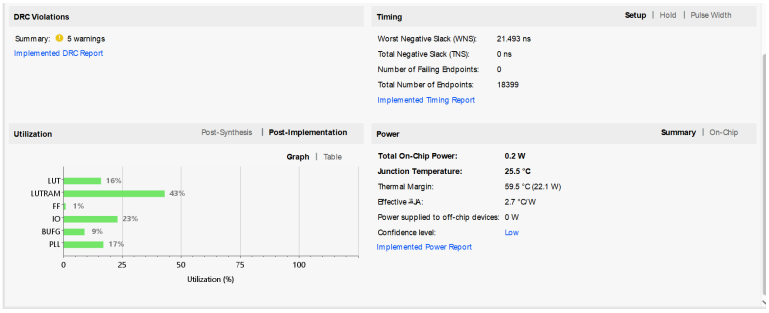
助记符 BIT#	使用语法	操作及解释 (r)表示寄存器r的值; Mem[addr]表示地址为addr的存储单元的值; sext表示有符号扩展
R-类型		
add	add rd, rs1, rs2	(rd) ← (rs1) + (rs2)
sub	sub rd, rs1, rs2	(rd) ← (rs1) - (rs2)
and	and rd, rs1, rs2	(rd) ← (rs1) & (rs2)
or	or rd, rs1, rs2	(rd) ← (rs1) | (rs2)
xor	xor rd, rs1, rs2	(rd) ← (rs1) ^ (rs2)
sll	sll rd, rs1, rs2	(rd) ← (rs1) << (rs2)
srl	srl rd, rs1, rs2	(rd) ← (rs1) >> (rs2), 逻辑右移
sra	sra rd, rs1, rs2	(rd) ← (rs1) >> (rs2), 算术右移
I-类型		
addi	addi rd, rs1, imm	(rd) ← (rs1) + sext(imm)
andi	andi rd, rs1, imm	(rd) ← (rs1) & sext(imm)
ori	ori rd, rs1, imm	(rd) ← (rs1) | sext(imm)
xori	xori rd, rs1, imm	(rd) ← (rs1) ^ sext(imm)
slli	slli rd, rs1, shamt	(rd) ← (rs1) << shamt
srli	srli rd, rs1, shamt	(rd) ← (rs1) >> shamt, 逻辑右移
srai	srai rd, rs1, shamt	(rd) ← (rs1) >> shamt, 算术右移
lw	lw rd, offset(rs1)	(rd) ← sext(Mem[(rs1) + sext(offset)][31:0])
jalr	jalr rd, offset(rs1)	t ← pc + 4; pc ← ((rs1) + sext(offset)) & ~1; (rd) ← t
S-类型		
sw	sw rs2, offset(rs1)	Mem[(rs1) + sext(offset)] ← (rs2)[31:0]
B-类型		
beq	beq rs1, rs2, offset	if (rs1 = rs2) pc ← pc + sext(offset)
bne	bne rs1, rs2, offset	if (rs1 ≠ rs2) pc ← pc + sext(offset)
blt	blt rs1, rs2, offset	if (rs1 < rs2) pc ← pc + sext(offset), 有符号比较
bge	bge rs1, rs2, offset	if (rs1 ≥ rs2) pc ← pc + sext(offset), 有符号比较
U-类型		
lui	lui rd, imm	(rd) ← sext(imm[31:12] << 12)
J-类型		
jal	jal rd, offset	(rd) ← pc + 4; pc ← pc + sext(offset)

设计的主要特色（除基本要求以外的设计）

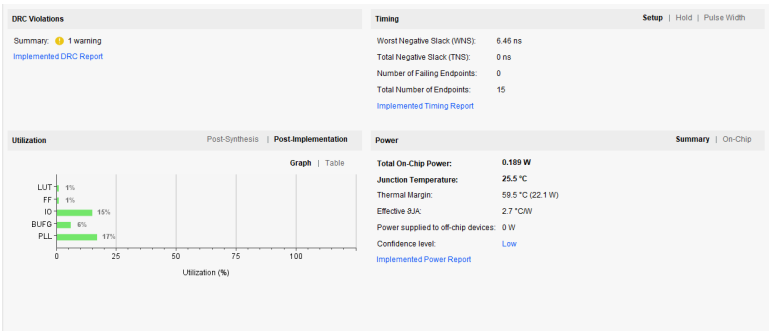
无。

资源使用情况、功耗数据截图（实现后）

- 单周期



- 流水线

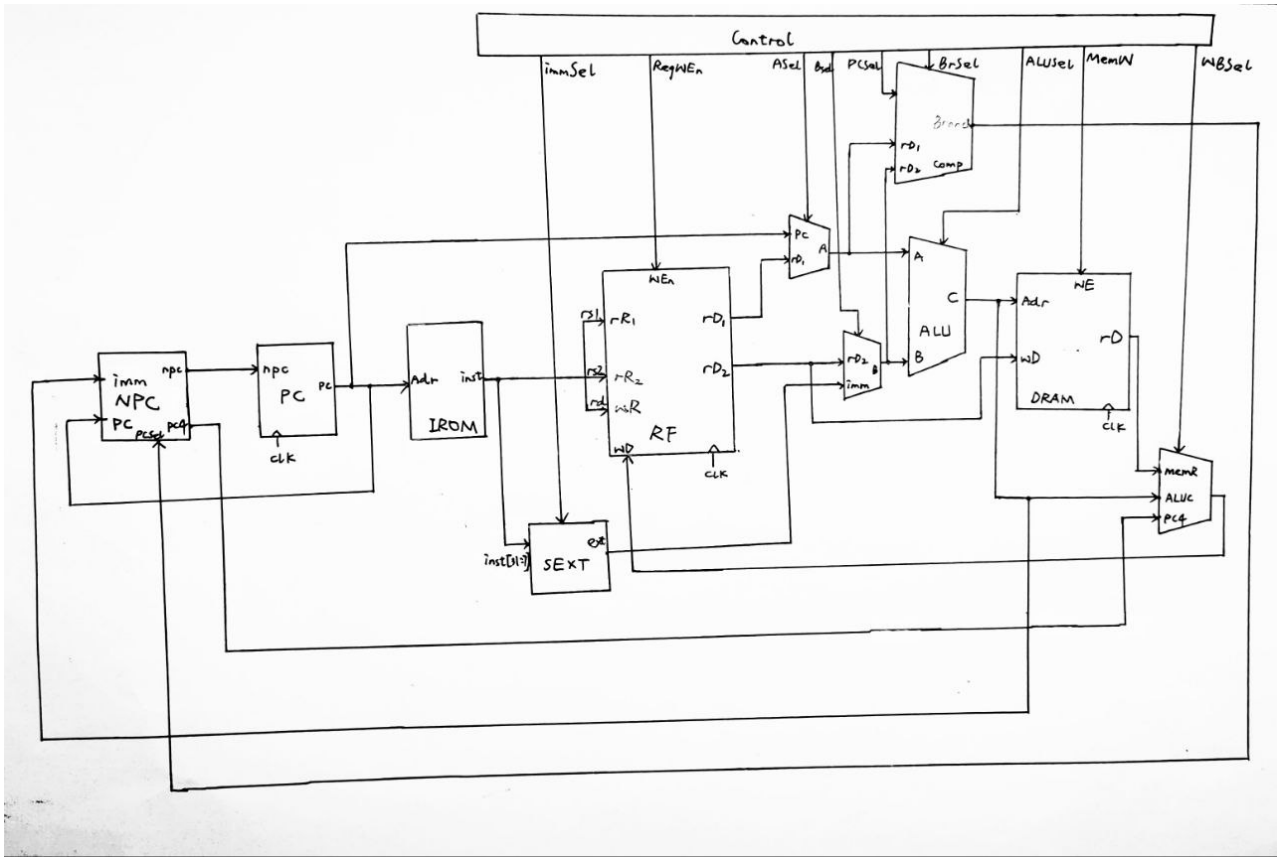


1 单周期 CPU 设计与实现

1.1 单周期 CPU 整体框图

（要求：无需画出模块内的逻辑，但要标出模块之间信号线的信号名和位宽，以及说明每个模块的功能含义）

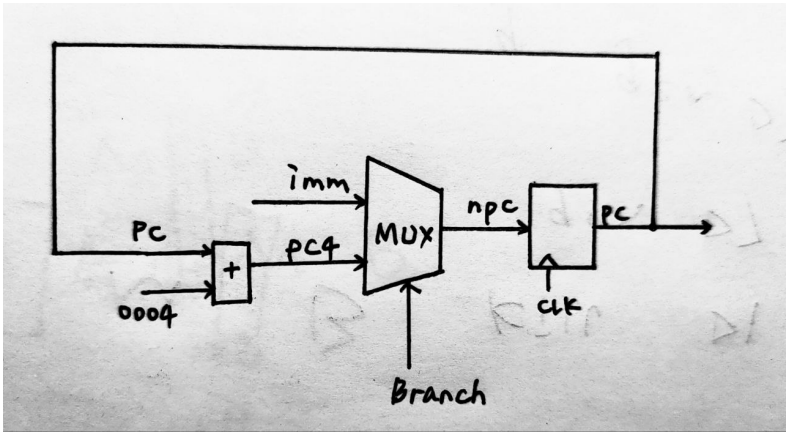
单周期整体框图



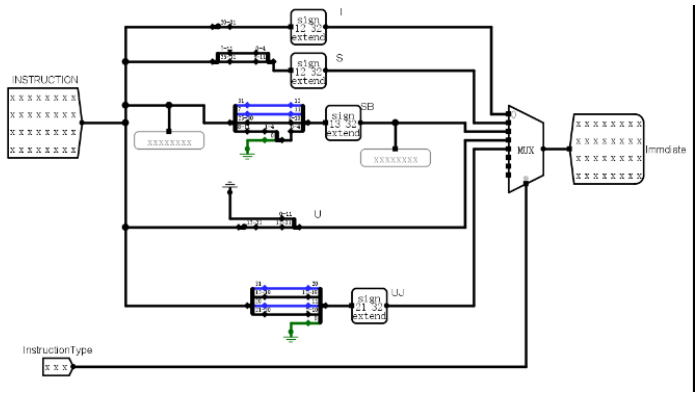
1.2 单周期 CPU 模块详细设计

（要求：各个模块的详细设计图，要包含内部的子模块，以及关键性逻辑，标出信号名和位宽，并有详细说明）

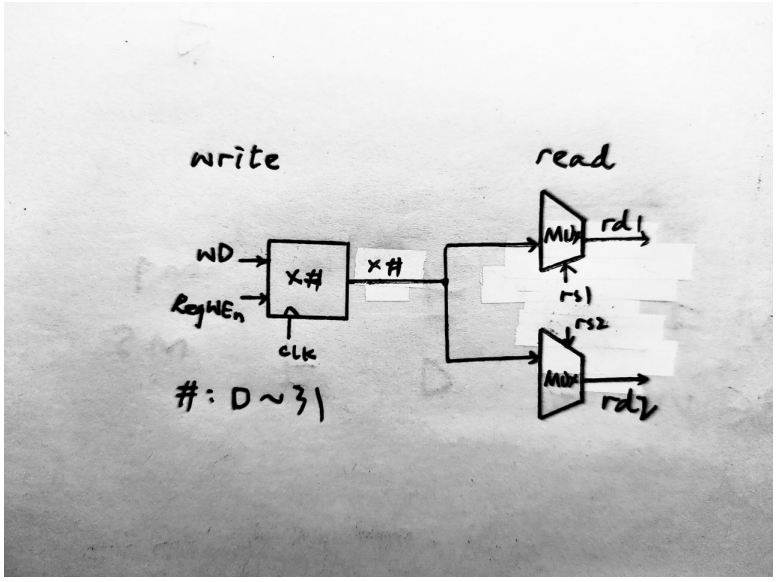
- NPC & PC



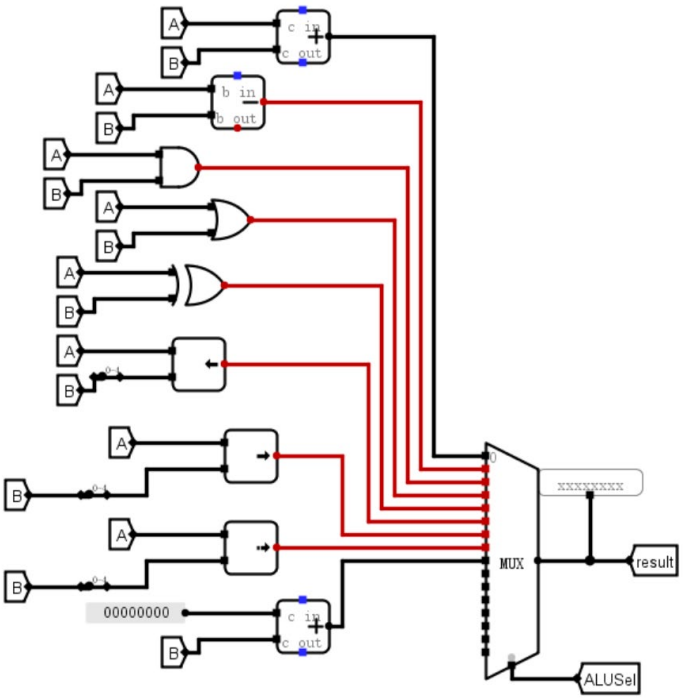
- SEXT



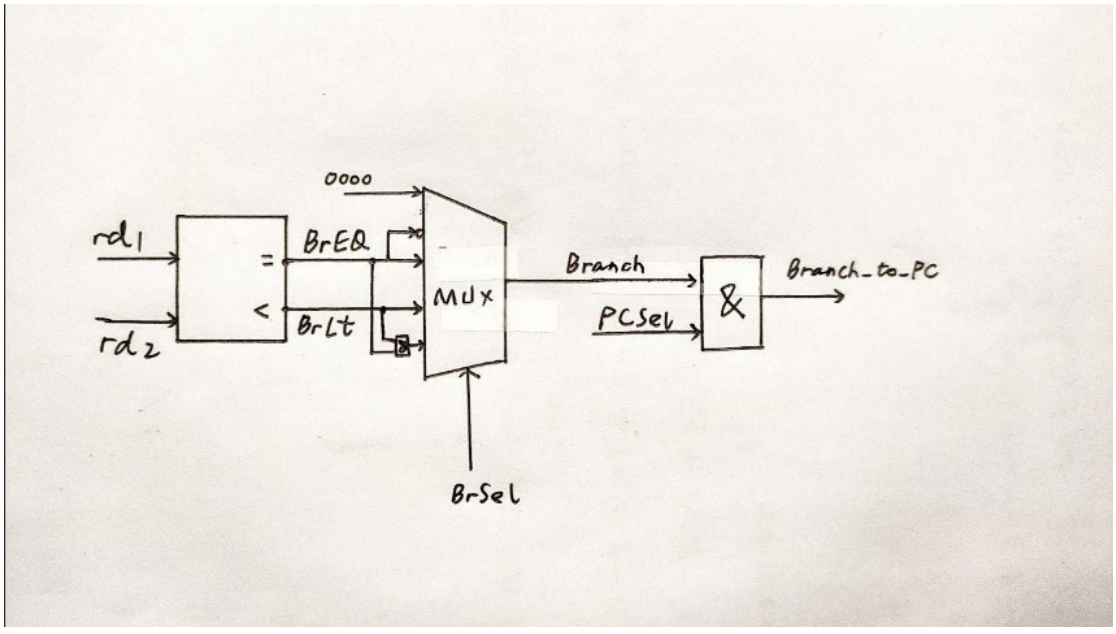
- RF



- ALU


A
B
c in
c out
b in
b out
00000000
MUX
xxxxxxxx
result
ALUSel

- COMP


0000
rd1
rd2
=
<
BrEQ
BrLt
MUX
Branch
PCSel
&
Branch_to_PC
BrSel

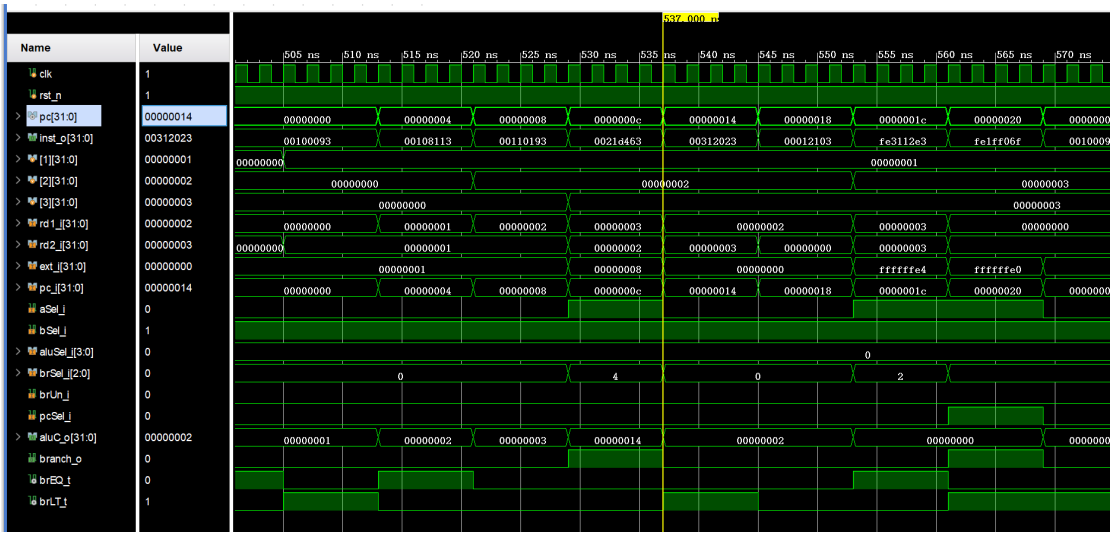
1.3 单周期 CPU 仿真及结果分析

（要求：包含逻辑运算指令、访存指令、跳转指令的仿真截图，以及结果分析）

```
12                  12  start:
11  00100093        11  addi x1, x0, 0x1
10  00108113        10  addi x2, x1, 0x1
 9  00110193         9  addi x3, x2, 0x1
 8  0021d463         8  bge x3, x2, mem_inst
 7  00108093         7  addi x1, x1, 0x1
 6                   6
 5                   5  mem_inst:
 4  00312023         4  sw x3, 0x0(x2) # store x3 in 0x2
 3  00012103         3  lw x2, 0x0(x2) # load x2 from 0x2, x2 = 0x3
 2  fe3112e3         2  bne x2, x3, start
 1  fe1ff06f         1  jal x0, start
15                  15
```

1. 逻辑运算指令

- 黄线处，指令为 bge；
- ALU 获取 rd1 与 rd2，得到判断 brEQ=0, brLT=0，满足跳转条件；
- pc 由 0x0C 跳转至 0x14

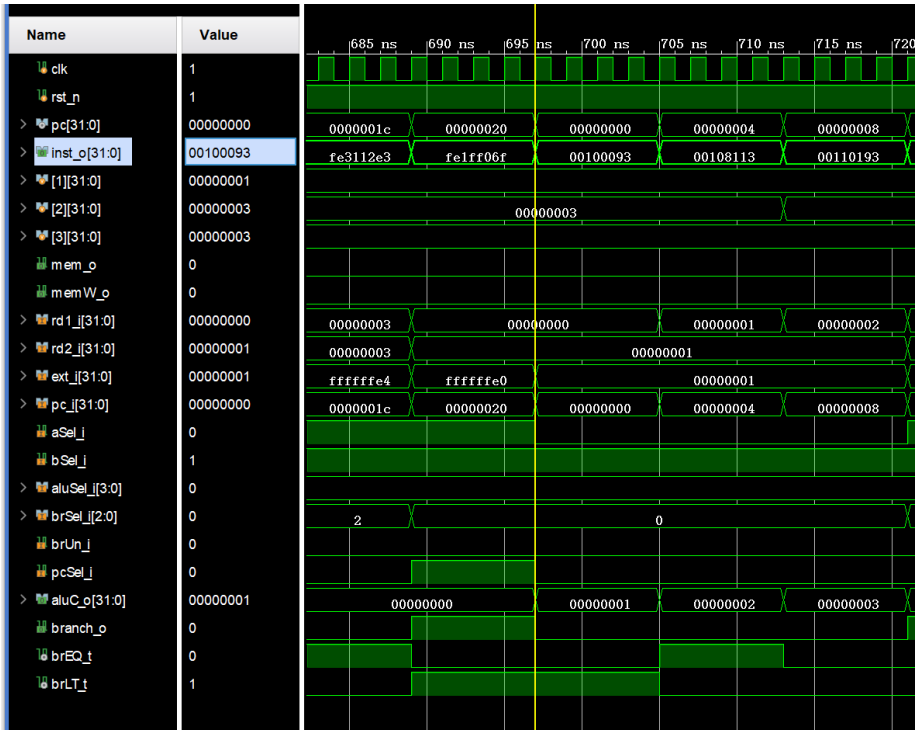


2. 访存指令

- 黄线前后为 sw 与 lw 指令，先将 x3 数值 3 存入 0x2 处，再用 x2 取出；
- 黄线前后的 mem_o 信号均为高电平，表明为访存操作；memW_o 信号前为低电平，表明为写入 DRAM；
- 可以看到 x2 的数据从 0x2 变成了 0x3，表示取出与写入成功

3. 跳转指令

- 黄线前为 jal 指令，跳转至 start 处（0x0），跳转正确



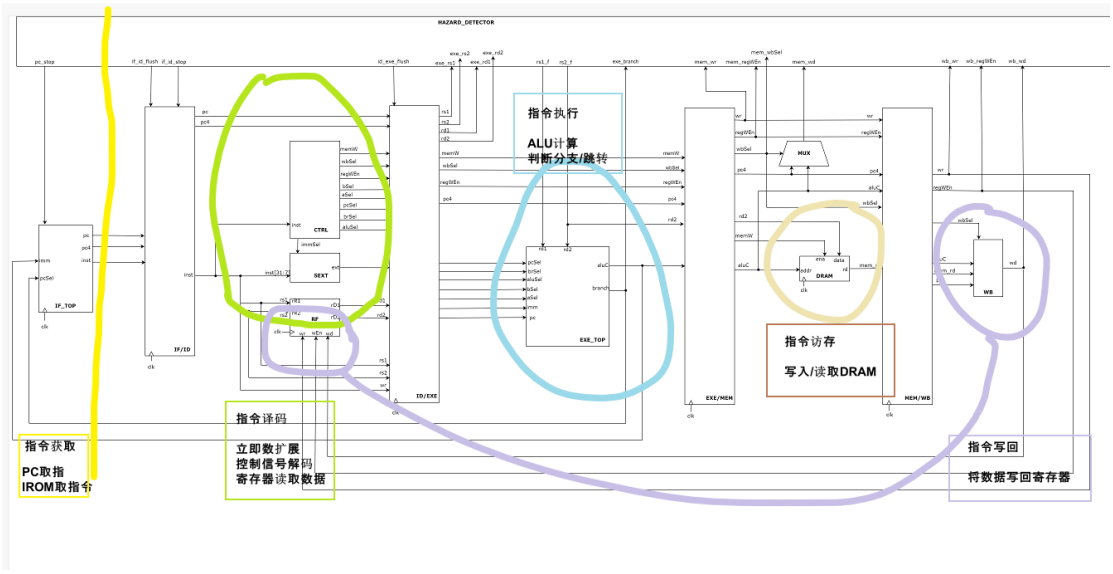
2 流水线 CPU 设计与实现

2.1 流水线的划分

（要求：画出流水线的划分，并标明每个阶段 CPU 完成的功能）

经典五级流水线：

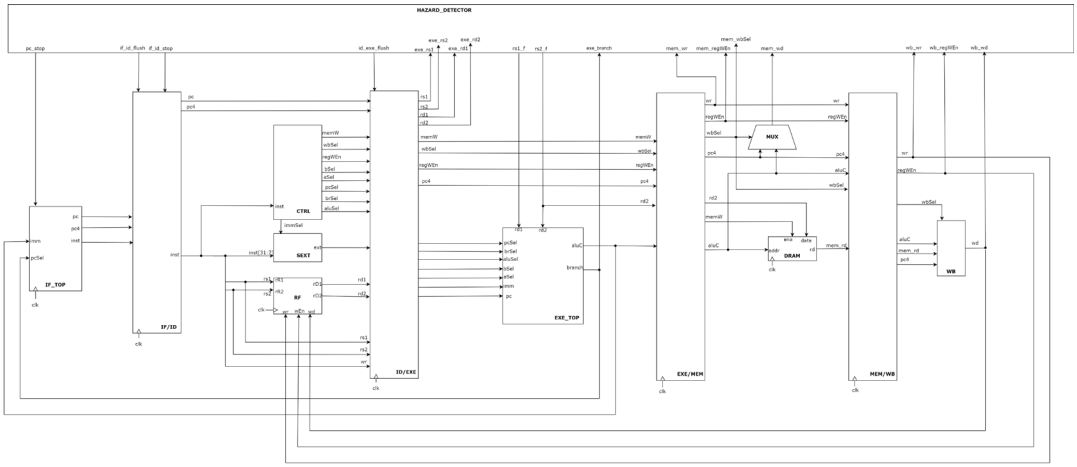
1. IF：指令获取
2. ID：指令译码
3. EXE：指令执行
4. MEM：指令访存
5. WB：指令写回



2.2 流水线 CPU 整体框图

（要求：无需画出模块内的逻辑，但要标出模块之间信号线的信号名和位宽，以及说明每个模块的功能含义）

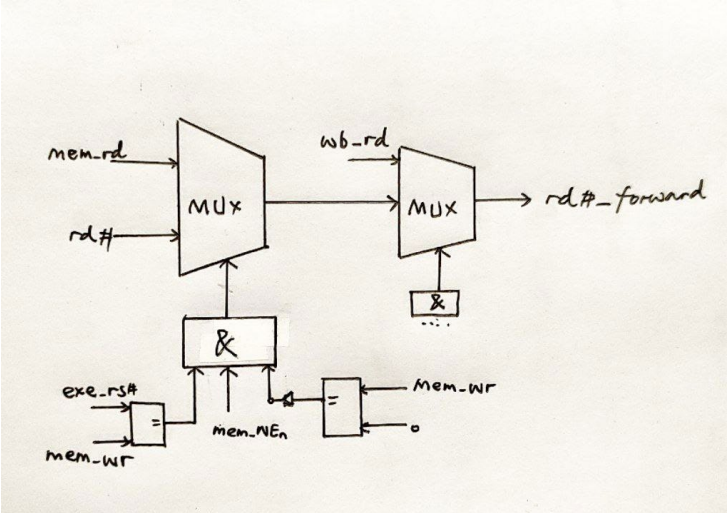
流水线整体框图



2.3 流水线 CPU 模块详细设计

（要求：各个模块的详细设计图，要包含内部的子模块，以及关键性逻辑，标出信号名和位宽，并有详细说明；数据冒险与控制冒险的解决方法必须要详细说明）

数据前递在整体框图中有详细连线，此处仅对 hazard_detector 模块进行说明：



数据冒险与控制冒险的解决方法：

- 数据冒险：在 ID/EXE 后半阶段检测冒险，将前递的数据传送到 `exe_top`/`alu` 中；也就是在 `hazard_detector` 判断并输出 ALU 的输入 `rs1_f`，`rs2_f`.
  - `rs_f` 三种选择：`rd`，`MEM.wd`，`WB.wd`
  - load 停顿控制：PC，IF/ID 停顿；ID/EXE 清空
  - 冒险类型
    - 1A：ID/EX.rs1 = EX/MEM.wR (其他省略)
    - 2A：ID/EX.rs2 = EX/MEM.wR
    - 1B：ID/EX.rs1 = MEM/WB.wR
    - 2B：ID/EX.rs2 = MEM/WB.wR
    - C：在同时写入和读取的情况下，对于读取数据进行判断 `rs==wr && regWEn && wr!=0`，是否**直接**读取写入的数据 `wd`.（另一种形式的前递）
- 控制冒险：假设分支不发生，如果发生分支跳转，则插入两个 bubble；也就是在 `hazard_detector` 中，传入 EXE 阶段的 `branch`，如果有控制冒险，则清空 IF/ID，ID/EXE 以达到插入两个 bubble 的目的.

2.4 流水线 CPU 仿真及结果分析

（要求：包含数据冒险、控制冒险的仿真截图，以及结果分析）

1. 控制冒险

```
0:  jal x0,4 <reset_vector>
4:  addi    x1,x0,0
8:  addi    x2,x0,0
c:  add x14,x1,x2
10: addi    x7,x0,0
14: addi    x3,x0,2
```

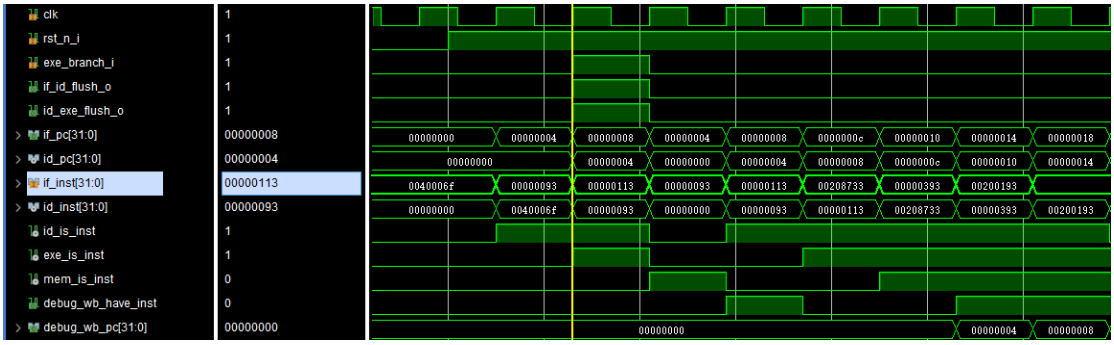
- 给出复位信号，由于根据『假设分支不发生』策略，PC 不断加 4；
- 当第一条跳转指令执行至 EXE 阶段时，立即发出 flush 信号，清空错误取出的两条指令，下一条指令 PC 为跳转指令给出的 Imm 值。

此时 EXE 传出跳转信号，npc 为指令计算结果

if_pc: 0 => 4 => 8 => 4 => 8

id_pc: 0=> 0 => 4 => 0 => 4

立即清空 IF/ID, ID/EXE 寄存器数值，标识指令无效



2. 数据冒险

```
    start:
1   addi x1, x0, 1 # x1 ← 1
2   addi x2, x1, 1 # x2 ← 2
3   addi x3, x1, 2 # x3 ← 3
4   addi x4, x1, 3 # x4 ← 4
5   addi x5, x4, 1 # x5 ← 5
6   add x6, x6, x1 # x6 ← x6 + 1
7   jal x0, start
```

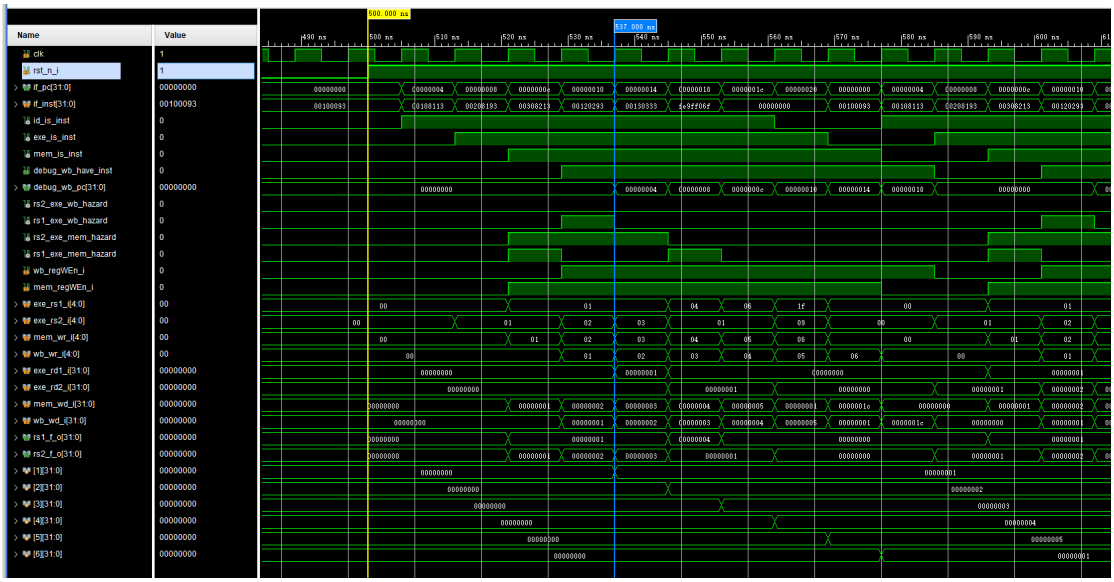
- 黄线处指令开始执行，五个周期后（蓝线处），数值才写入 x1 中
- 第二条指令与第一条指令的数据冒险（蓝线前两个周期）根据下图判断，将 mem_wd_i 前递至 rs2_f_o

```
// situation A: EXE & MEM hazard
assign rs2_exe_mem_hazard = (exe_rs2_i == mem_wr_i) && mem_regWEn_i && mem_wr_i =/= 0;

// situation B: EXE & WB hazard
assign rs2_exe_wb_hazard = (exe_rs2_i == wb_wr_i) && wb_regWEn_i && wb_wr_i =/= 0;

// rs2: forward data
always @(*) begin
    if (rs2_exe_mem_hazard)     rs2_f_o = mem_wd_i ;
    else if (rs2_exe_wb_hazard) rs2_f_o = wb_wd_i  ;
    else                        rs2_f_o = exe_rd2_i;
end
```

其他两个数据冒险是同样的分析方式，不再赘述。



3 设计过程中遇到的问题及解决方法

解决方法：看波形分析，Trace 与仿真双管齐下

问题：

1. Passed Tests: beq, bne
   Failed Tests: blt, bge

   * 等于正确, 大于小于错误

```
case (a_b_sign_eq)
    0: brLT_o = rd1_i[30:0] < rd2_i[30:0] ? `BRLT_T : `BRLT_F;
< 1: brLT_o = rd1_i[31] == 0 ? `BRLT_T : `BRLT_F;
---
> 1: brLT_o = rd1_i[31] == 1 ? `BRLT_T : `BRLT_F;
endcase
```

2. Passed Tests: slli, srai, srli
   Failed Tests: sll, sra, srl

   * 立即数移位操作正确, 寄存器移位错误

```
< `ALUSEL_SRL: aluC_o = a >> b;
---
> wire [4:0] shamt = b[4:0];
> `ALUSEL_SRL: aluC_o = a >> shamt;
```

3. 将前递的数据传入寄存器

   仅 `sw` 未通过.

   * `wb_top` 中 `mem_rd_i` 恒为零, 意即: `sw` 无法读出**lw 写入的数据**

```
exe_mem_reg CPU_EXE_MEM (
    .clk_i        (clk       ),
    .rst_n_i      (rst_n_i   ),
<   .exe_rd2_i    (exe_rd2   ),
>   .exe_rd2_i    (rs2_forward),
```

4 总结

（要求：个人收获以及对课程的建议）

个人收获：

1. 学会写实验报告，学会使用 Excel，更熟练地使用 Word
2. 更加理解 CPU 的内部构造与流水线的底层实现

建议在课程前将［流水线］整体硬件框图实现。