# 计算机体系结构

Lab5 带有多周期指令和乱序执行的流水线CPU

11.4 - 11.18

# 实验目标

- 设计一款支持乱序执行的多周期流水线CPU
- 将CPU设计成IF/ID/FU/WB四阶段，且FU阶段支持多周期
- 支持指令乱序完成，并检测和解决CPU执行过程的冒险情况

# 背景知识-多周期

- 普通流水线：每个Cycle的用时取决于最长的流水级
  乘法等指令一步完成需经过非常复杂的结构，使得FU时间变得很长 ☹
- 多周期：复杂指令分解到多个周期执行
  将复杂的操作分解到每个周期，每个周期内的结构复杂度相应减小 😀
- 多个FU和乱序完成：每种指令有不同的FU，可以同时执行
  慢的指令不会阻塞后面指令的执行，加快整体的执行效率 😀

# 背景知识-多周期的问题

- **结构冲突**

执行冲突：前后两条指令都要使用同一个FU，但FU只能同时执行一条指令

此时后一条要等前一条先完成

写回冲突：不同FU的指令同时完成执行并在同一时刻需要写回，但写回总线只有一条，不能同时写两个结果

要在开始执行时就避免这种情况

- **WAW冲突**

前面的指令比后面的指令执行更慢，会导致后面指令写回的结果被前面的指令写回的结果覆盖

后面的指令需要先等待，直到这种情况不会发生，再开始执行

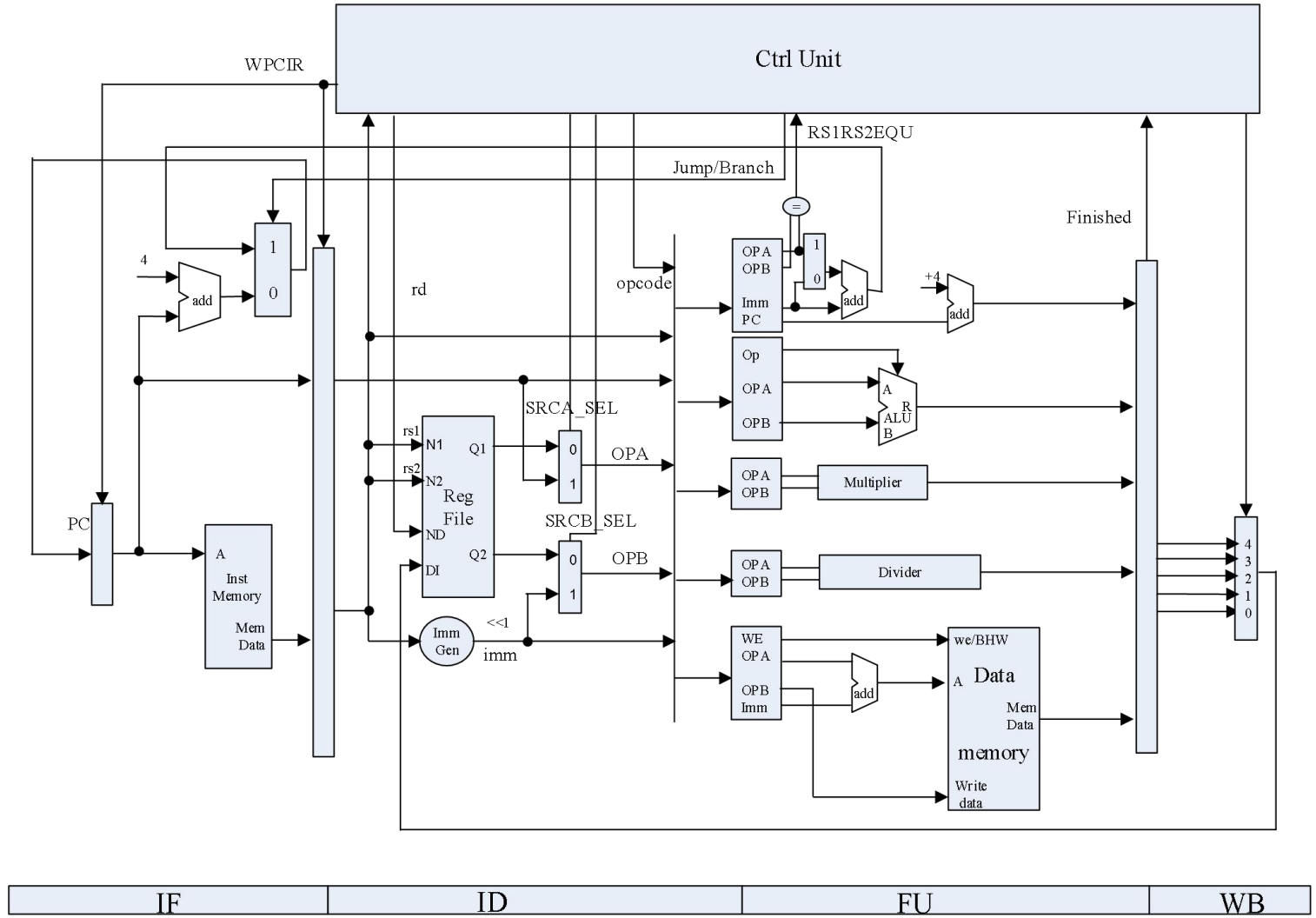
- **RAW冲突**

后面的指令需要前面指令的写回结果，但前面指令还没完成执行，需要等待

后面的指令需要先等待，直到这种情况不会发生，再开始执行

# 背景知识-多周期流水线CPU结构

- IF：取指令
- ID：**等待**直到执行当前指令不会引起冒险
- FU：执行指令
- WB：写回结果



# 背景知识-冒险和等待的判断

- 判断FU目前是否有指令正在执行

  为每个FU设置一个Flag，开始执行指令时设置为1，执行完成时设置为0，
  这样就可以借助Flag的值来判断FU中有没有指令正在执行
- 开始执行时，判断是否会存在两个指令同时写回

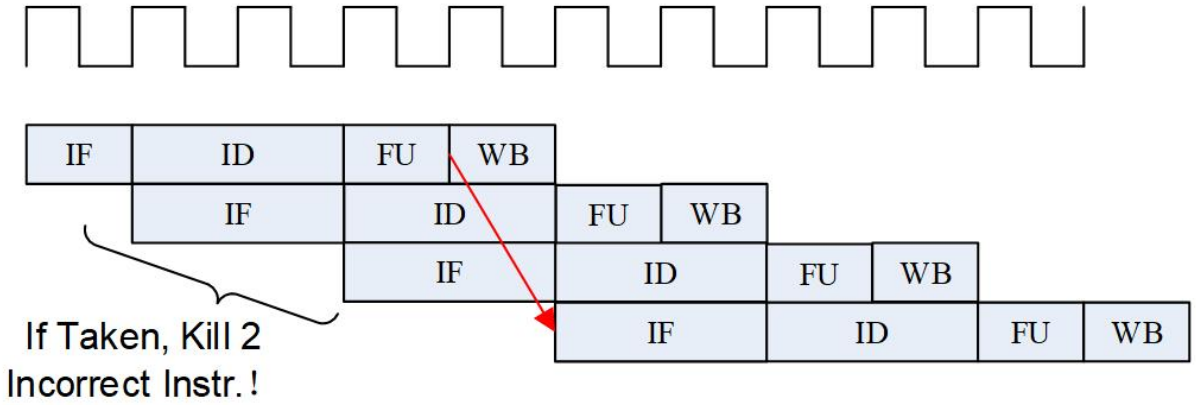
  设置一个预约寄存器，如果当前开始执行的指令要在N个周期后写回，则将预约寄存器的第N位置位，
  每个时钟周期，预约寄存器整体左移。这样就可判断当前M个周期后是否有指令需要写回
- 判断WAW

  记录每个FU将会写回到什么位置，同时利用预约寄存器判断目前FU将会在多久以后写回，
  这样只需要等待，直至当前指令在其之后写回则不会发生WAW。
- 判断RAW

  记录每个FU将会写回到什么位置，同时利用上述Flag来判断该FU是否仍然未完成执行。
  等待直到结果已经写回，即可避免RAW

# 实验细节

- 控制冒险依然存在，需要实现Prediction-Not-Taken策略




- 预约寄存器中每个位置可以是具体需要写入的FU的编号，便于进行各种判断


- 需要实现五个FU，包括ALU，MEM，MUL，DIV和JUMP

这些模块需要规定各自的latency，默认latency设置可以参考config.json中的设置，
也可以自行设置latency

# 实验得分点

- 实现FU-ALU: 10
- 实现FU-MEM: 10
- 实现FU-MUL: 10
- 实现FU-DIV: 10
- 实现FU-JUMP: 10
- 结构冒险：20
- RAW: 10
- WAW: 10
- Prediction-not-Taken: 10

# 实验步骤

- 照旧