### 第五次作业

How Tomasulo’s Algorithm technique solves WAR, WAR and RAW hazards?

* WAW，WAR: Tomasulo在issue阶段解决WAW，WAR问题，在issue阶段从指令队列的头部获取下一条指令。如果有一个空的保留站，就把它放在那里。如果不是，则存在结构性风险并且指令会停止，直到释放站或缓冲区。此步骤重命名寄存器，消除了RAW和WAR问题
* RAW: Tomasulo在EXE阶段解决RAW问题，在EXE阶段，观察 CDB，直到所有需要的操作数都在保留站中。当所有的操作数都准备好后，才可以执行操作了。通过延迟指令执行直到操作数可用，避免了RAW问题。

How Scoreboard technique solves WAR, WAR and RAW hazards?

* WAW: 记分牌在issue阶段解决WAW问题，issue阶段开始的条件是功能单位没有被占用，且没有其他操作在写当前的目标寄存器，这样直接避免写WAW问题。
* RAW: 记分牌在read operands阶段解决RAW问题，read operands的开始条件是Rj和Rk为true即没有操作在写源寄存器，如果有，Rj和Rk已经在issue阶段被赋值为false，只有当写回操作完成后，才会将Rj和Rk赋值为0，即在issue阶段赋值为true，动态避免了RAW问题。
* WAR: 记分牌在write阶段解决WAR问题，write result阶段的开始条件是对于任意其他功能单元来说，没有任何功能单元正在读该指令的目标寄存器，或者任何功能单元Rj和Rk为false，直到读操作完成后才能完成写操作，避免了WAR问题