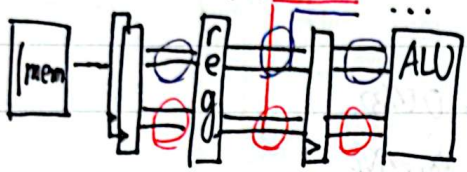
No.

Date

Multi-Issue:

在之前讨论情景中，1个 clock cycle 中最多1个 instruction 有问题
考虑如何提高并行效率：①是 加深流程，如一个环节拆成2个
②是'多几套这样的流水线'，这称为 multiple issue
在②做法下，原来 pipeline 中的 CPI ≥1，可能变为 CPI <1

fig: 

两套！比方：洗衣房中一湿一烘
⇒三湿三烘

可见，这种做法主要对 Hardware 提出了要求。
那么理想情况下，指令 scheduling 尽量 1个ALU or Branch，1个 Load or Store，这两个组一组，一起进 Multi-Issue. 可能有 Hazard，如两组间有 Read before Write，但一般 forwarding 也可解决
或者：与 nop 组一组；又或者 调换顺序

可见 scheduling 是重要的！它可能是 compile time 或 execution time 来 'schedule' 的 (又名: Packaging)
Static ↑ Dynamic (by hardware)*.

In most static issue processors: partially handled by Compiler
In dynamic issue designs: be normally dealt with at runtime by processor, although compiler often have already tried to help improve by placing instructions in a beneficial order.

而对于 dealing with data/control hazard:
hardware technique
Static: compiler handles some or all data/control hazards.
Dynamic: processors attempt to alleviate some classes of hazards

*: Also called superscalar:
示意图：IF & ID → Reservation Station → Integer ALU
→ R S → Float Unit
→ R S → Load/Store Unit
→ Commit Unit with reorder buffer

Multi-issue is not Multi-core, nor SIMD; it can be combined with pipelining, SIMD, etc