1. 差異在於 pipeline 更能有效利用各個 processor 區間(IF, ID ,EX, MEM, WB)
   Multicycle 為分成多步驟完成任務
   Pipeline 一級完成一部分, 便交給下一級處理, 達到管線化效果
   理想的 Pipeline CPI = 1, cycle time 等於 Multicycle

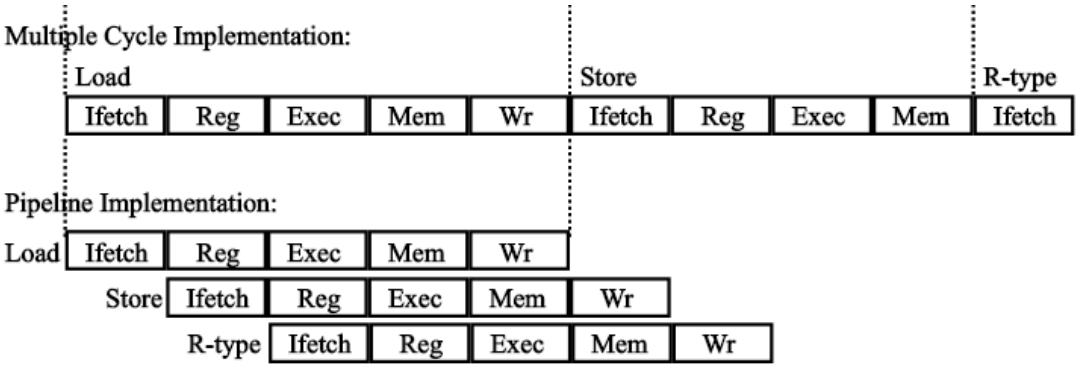


   Image from http://slideplayer.com/slide/9178251/

2. Structural hazard
   資源需求大於資源數量時發生, 須等資源釋放才能解除

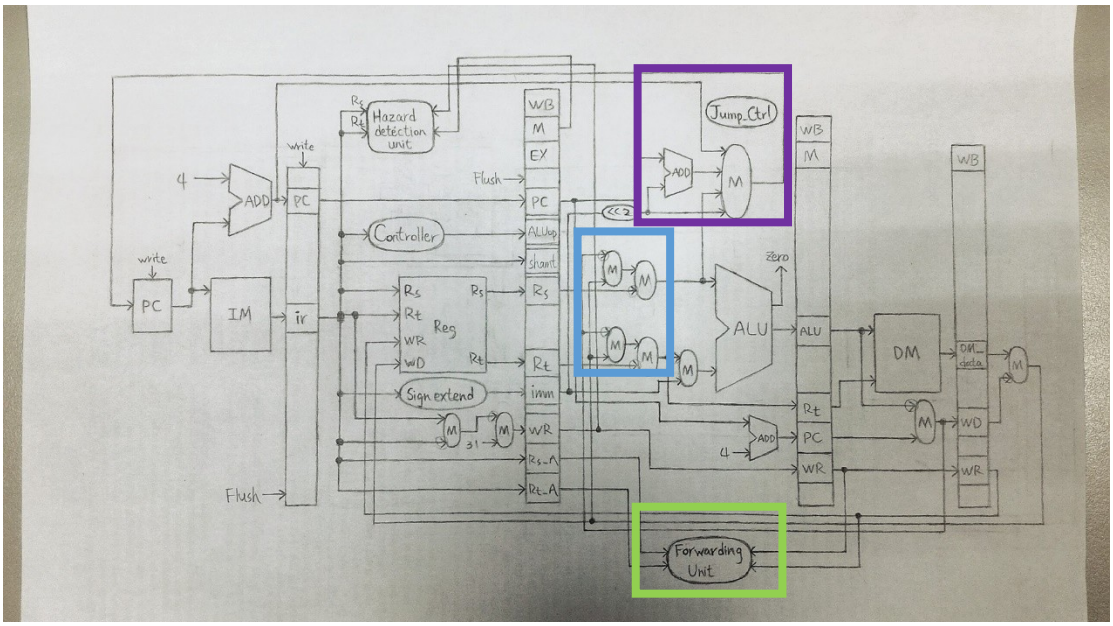
   Data hazard
   常見於某些指令需要的資料還在 pipe 裡, 但尚未存回 register
   可透過 forwarding 方式解決

   control hazard
   branch 發生時, 將清除部分已讀取指令, 造成效能下降
   可透過 branch 判斷提早, 減輕效能下降問題 或
   使用 branch prediction

3. 假設 stage 和處理指令複雜度有正相關
   Pros：越高的 stage 意味能用更少程式碼來完成程式(一條指令能做更多事)

   Cons：電路複雜度上升, 需要更多機制去維持其效能性
   (ex forwarding 更複雜)
   Branch 可能清除更多已讀取指令 -> control hazard

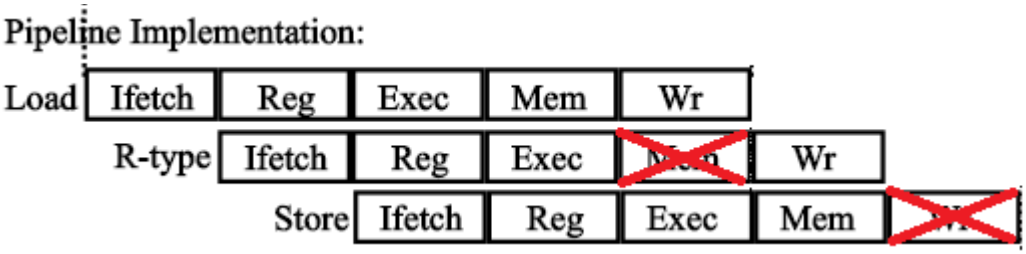
4. 這題要用程式來描述如何處理 data 和 control hazard (課本就有 所以我偷懶)



Data hazard 使用 forwarding 將資料拉回並使用 **藍色為多工器 綠色為控制電路**

Control hazard 我將 branch 判斷移到 **EX 端**, ALU 可運算判斷結果, forwarding 可將值拉回供 ALU 判斷, branch 動作將犧牲 2 指令, 不過架構比 branch prediction 簡單許多

5.



Pipeline 只花 6 cycles 完成 multicycle 要花 13 cycles
可看出 Pipeline 只要一半不到的時間就能完成